Proceedings of the

9th International Workshop on Non-Monotonic Reasoning

NMR'2002

April 19-21, Toulouse, France

Volume Editors

Salem Benferhat
Enrico Giunchiglia

With Contribution From

Mary-Anne Williams
Lluis Godo, Thomas Lukasiewicz
Patrick Doherty, Michael Thielscher
Tony Kakas, Ilkka Niemela
Frederic Cuppens, Henry Prakken, Leon van der Torre

Preface

The papers in this collection were presented at the 9th International Workshop on Nonmonotonic Reasoning, NMR2002, held in Toulouse, France on April 19-21, 2002. The series was started by John McCarthy in 1978. The first international NMR workshop was held at Mohonk Mountain House, New Paltz, New York in June, 1984, and was organized by Ray Reiter and Bonnie Webber. The other workshops in the series were held in Grassau (Germany, 1988), South Lake Tahoe (California, 1990), Plymouth (Vermont, 1992), Schoss Dagstuhl (Germany, 1994), Timberline (Oregon, 1996), Trento (Italy, 1998), Breckenridge (Colorado, 2000).

This was the 9th workshop in the NMR series. Its aim was to bring together active researchers in the broad area of NMR, including belief revision, reasoning about actions, planning, logic programming, causality, probabilistic and possibilistic approaches to KR, and other related topics. The Workshop included invited talks, presentations of technical papers and special sessions.

The papers in these proceedings reflect some of the recent advances in the field. They are grouped into sections corresponding to the special sessions as they were held at the workshop. The sessions were

- General Track
- Changing and Integrating Information: From Theory to Practice
  Session chair: Mary-Anne Williams
- Answer Set Programming and Abductive Reasoning
  Session chairs: Tony Kakas and Ilkka Niemela
- Uncertainty Frameworks in NMR
  Session chairs: Lluis Godo and Thomas Lukasiewicz
- Representing Action and Change
  Session chairs: Patrick Doherty and Michael Thielscher
- Argument, Dialogue and Decision
  Session chairs: Frederic Cuppens, Henry Prakken and Leon van der Torre

The highlights of the workshop were three invited talks, given by Georg Gottlob (Technische Universität Wien), Vladimir Lifschitz (University of Texas at Austin), and Mary-Anne Williams (University of Newcastle).

Many people were involved in the organization of the workshop. Our special thanks go to all PC members and all organizers of special sessions. Without their effort and involvement the workshop could not have taken place. We are indebted to all the members of the organizing committee for their support, which included setting up and maintening Web pages.

Salem Benferhat, Enrico Giunchiglia
Workshop General co-Chairs

April 19^{th} 2002

NMR-2002 committees

NMR-2002 program chairs

Salem Benferhat
Enrico Giunchiglia

General track

Program committee

Eyal Amir
Gerd Brewka
James P. Delgrande
Eric Grégoire
Hector Levesque
Paolo Liberatore
Simon Parsons
David Poole
Henri Prade
Torsten Schaub
Camilla Schwind
Mirek Truszczynski
Hudson Turner

Special Session: Changing And Integrating Information From Theory To Practice

Session chair:
Mary-Anne Williams

Session program commitee:
Leopoldo Bertossi
Gerd Brewka
Samir Chopra
James Delgrande
Didier Dubois
Anthony Hunter
Sebastien Konieczny
Daniel Le Berre
Wei Liu
Shiela Mcilraith
Thomas Meyer
Leora Morgenstern
Pavlos Peppas
Simon Parsons
Ramon Pino Perez
Torsten Schaub
Stuart Shapiro

Special Session: Answer Set Programming and Abductive Reasoning

Session chairs:
Patrick Doherty
Michael Thielscher

Session program committee:
Chitta Baral
Marc Denecker
Yannis Dimopoulos
Thomas Eiter
Michael Gelfond
Katsumi Inoue
Vladimir Lifschitz
Antonis Kakas
Nicola Leone
Ilkka Niemelä
Ken Satoh
Francesca Toni
Mirek Truscszynski
Jia-Huai You

Special Session: Uncertainty Frameworks in Nonmonotonic Reasoning

Session chairs:
Lluis Godo
Thomas Lukasiewicz

Session program committee:
Alexander Dekhtyar
Didier Dubois
Nir Friedman
Hector Geffner
Angelo Gilio
Anthony Hunter
Manfred Jaeger
Gabriele Kern-Isberner
Ramón Pino Pérez
Frans Voorbraak
Emil Weydert
Ronald R. Yager

Special Session: Representing Action and Change

Session chairs:
Patrick Doherty
Michael Thielscher

Special Session: Argument, Dialogue and Decision

Session chairs:
Frederic Cuppens
Henry Prakken
Leon van der Torre

Session program committee:
Philippe Besnard
Gerd Brewka
Phan Minh Dung
Jeff Horty
Nicolas Maudet
Ron Loui
Simon Parsons
Fransesca Toni
Gerard Vreeswijk

Additional referees

Giangiacomo Gerla
Anthony Hunter
Stefan Woltran

Local organizers

Salem Benferhat
Jean-François Bonnefon
Frédéric Cuppens
Rania El Baida
Souhila Kaci
Sylvain Lagrue
Rui Da Silva Neves
Vincent Vidal

Sponsors

CNRS
Conseil Régional de Midi-Pyrénées
Feria
IRIT
Mairie de Toulouse
ONERA, centre de Toulouse
Université de Toulouse le Mirail
Université Paul Sabatier (Toulouse III)

Table of Contents

General Track

Collaborative belief revision (invited speaker) 3
Mary-Anne Williams

Visual Data Extraction Through Visual Logic Programming (invited speaker) 4
Georg Gottlob

Why Sam doesn't know calculus (invited speaker) 5
Vladimir Lifschitz

Compilation of propositional weighted bases 6
Adnan Darwiche, Pierre Marquis

Linking Makinson and Kraus-Lehmann-Magidor preferential entailments 15
Yves Moinard

Some logics of belief and disbelief 25
Samir Chopra, Johannes Heidema, Thomas Meyer

What is a joint goal? Games with beliefs and defeasible desires 33
Mehdi Dastani, Leendert van der Torre

Interpolation theorems for nonmonotonic reasoning systems 41
Eyal Amir

Compositionality results for stratified nonmonotone operators 51
David Gilis, Marc Denecker

A polynomial translation of logic programs with nested expressions into disjunctive logic programs: preliminary report 57
David Pearce, Vladimir Sarsakov, Torsten Schaub, Hans Tompits, Stefan Woltran

Logic programming with ordered disjunction 67
Gerhard Brewka

Optimality theory through default logic 77
Philippe Besnard, Robert E. Mercer, Torsten Schaub

Defeasible logic with dynamic priorities 84
Grigoris Antoniou

Intuitions and the modelling of defeasible reasoning: some case studies 91
Henry Prakken

Changing and Integrating Information: From Theory to Practice

Possibility theory in information fusion (invited talk) 103
Didier Dubois, Henri Prade

Iterated revision and the axiom of recovery: a unified treatment via epistemic states 117
Samir Chopra, Aditya Ghose, Thomas Meyer

Explanatory diagnosing and meaningful perception 124
Alberto Finzi, Fiora Pirri

Two representations for iterative non-prioritized change 135
Alexander Bochman

Revising partially ordered beliefs 142
Salem Benferhat, Sylvain Lagrue, Odile Papini

Considerations on compositional update operators 150
James Delgrande, Matthew Suderman

Repairing databases with annotated predicate logic 160
Pablo Barceló, Leopoldo Bertossi

Dynamical revision operators with memory 171
Sébastien Konieczny, Rámon Pino Pérez

Answer Set Programming and Abductive Reasoning

Unifying abductive logic programming and stable logic programming (invited talk) 183
Marc Denecker

Abduction, ASP and open logic programs 184
Piero A. Bonatti

Speculative computation and abduction for an autonomous agent 191
Ken Satoh

Pruning operators for answer set programming systems 200
Francesco Calimeri, Wolfgang Faber, Nicola Leone, Gerald Pfeifer

More on noMoRe 210
Thomas Linke, Christian Anger, Kathrin Konczak

On logic programs with cardinality constraints 219
Victor W. Marek, Jeffrey B. Remmel

Alternative characterizations for strong equivalence of logic programs 229
Pedro Cabalar

Preferred well-founded semantics for logic programming by alternating fixpoints: preliminary report . 238
Torsten Schaub, Kewen Wang

Answer sets generated by selector functions 247
Sibylle Schwarz

Uncertainty Frameworks in NMR

Evaluating defaults 257
Henry E. Kyburg Jr., Choh Man Teng

Nonmonotonic probabilistic logics between model-theoretic probabilistic logic and probabilistic logic under coherence 265
Thomas Lukasiewicz

Default reasoning in a coherent setting 275
Giulianella Coletti, Romano Scozzafava, Barbara Vantaggi

Default epistemology rankings, reasoning and revision (a very preliminary account) 283
Emil Weydert

Decision making by intelligent agents: logical argument, probabilistic inference and the maintenance of beliefs and acts 293
John Fox, Peter McBurney

Embedding default logic in propositional argumentation systems 302
Dritan Berzati, Bernhard Anrig, Jürg Kohlas

Fuzzy belief revision 311
René Witte

Nonmonotonic inference operators for fuzzy logic 321
Eva Richter

Representing Action and Change

Towards a conditional logic of actions and causation 333
Laura Giordano, Camilla Schwind

It depends on the context! A decidable logic of actions and plans based on a ternary dependence relation 343
Marcos A. Castilho, Andreas Herzig, Ivan J. Varzinczak

Learning non-monotonic causal theories from narratives of actions 349
David Lorenzo

Planning with complex actions 356
Sheila McIlraith, Ronald Fadel

The mutual exclusion problem in reasoning about action and change 365
Jan Broersen, John-Jules Meyer, Roel Wieringa

Intensionality and circumscription 372
Graham White

Modeling complex domains of actions and change 380
Antonis Kakas, Loizos Michael

Argument, Dialogue and Decision

Skepticism and floating conclusions (invited paper) 391
John F. Horty

On concise encodings of preferred extensions 392
Paul E. Dunne

Strategical considerations for argumentative agents (preliminary report) 399
Leila Amgoud, Nicolas Maudet

"Minimal defence": a refinement of the preferred semantics for argumentation frameworks 408
Claudette Cayrol, Sylvie Doutre, Marie-Christine Lagasquie-Schiex, Jérôme Mengin

On the existence and multiplicity of extensions in dialectical argumentation 416
Bart Verheij

Well-founded argumentation semantics for extended logic programming 426
Ralf Schweimeier, Michael Schroeder

Collective argumentation 436
Alexander Bochman

Value-based argumentation frameworks 444
Trevor Bench-Capon

Author Index 455

General Track

Collaborative Belief Revision

Mary-Anne Williams
Centre of eCommerce for Global Business
Faculty of Business and Law
The University of Newcastle, NSW 2308, Australia
email: maryanne@ebusiness.newcastle.edu.au

Abstract

Belief Revision lurks behind many forms of intelligent behaviour. In this talk I plan to give a brief overview of the theoretical underpinnings of Belief Revision and to describe a number of practical techniques which are employed in applications. There are many applications for belief revision in ebusiness. For example, one of the main problems that arises when building models for consumer profiles is the need to represent and revise profiles as more consumer behaviour information is acquired. Ad serving companies like sabela.com and doubleclick.com use primitive forms of belief revision to determine the banner ad to display based on a simple dynamic consumer profile.
I will also describe how the process of information integration for merging knowledge bases can be viewed and implemented as a generalised form of belief revision. Instead of incorporating a new piece of information into a knowledge base by computing the "most important" information from the knowledge base that is compatible with the new information, we build a new knowledge base by fusing the "most important" compatible information from each knowledge base.

Finally using the belief revision techniques for change and integration I will describe an agent-oriented collaborative belief revision system.

Visual Data Extraction Through Visual Logic Programming

Georg Gottlob
Technische Universität Wien
Institut für Informationssysteme 184/2
Abteilung für Datenbanken und Artificial Intelligence
Favoritenstraße 9
A-1040 Wien, Austria
e-mail: gottlob@dbai.tuwien.ac.at

Abstract

This talk starts with a short demo of the Lixto Web extraction tool. With this commercial tool a user can visually specify and annotate relevant information to be extracted from Web pages. While the user works on example Web pages, the program translates the user actions into a logic program, more precisely, a datalog program, whose rules describe the patterns to be extracted. The program (aka wrapper) can then be applied to similarily structured Web pages for extracting the appropriate patterns and translating them into XML. In the second part of the talk, some theoretical issues on Lixto will be presented. We will show that Lixto expresses monadic second order logic over trees (i.e., parsing trees of Web pages). Moreover, we will discuss some challenges for nonmonotonic reasoning related to information extraction.

More information on Lixto can be found at www.lixto.com or www.lixto.net.

Why Sam Doesn't Know Calculus

Vladimir Lifschitz
University of Texas at Austin

Abstract

Solutions to the frame problem available today can be applied to a larger problem that motivated research on nonmonotonic logic in the first place – to formalizing commonsense knowledge and reasoning. In this work, the need for defaults and nonmonotonic conclusions is a small but essential part of the issues involved.

Consider, for instance, the following problem from the Common Sense Problem Page (http://www-formal.Stanford.EDU/leora/cs/). Sam got straight C's in high school math and has not thought for a moment about math in the 20 years since. Infer that Sam is not the person to ask about a calculus problem. (Contributed by Ernie Davis.) Once we inferred that Sam didn't know calculus when he graduated from high school, how can we move on to the conclusion that Sam doesn't know calculus today? The argument is that Sam has not performed any actions that would cause him to learn calculus. This argument is nonmonotonic – it is based on the commonsense law of inertia.

We will discuss a representation of this example in a nonmonotonic causal logic and the use of an implementation of that logic, called the Causal Calculator, to justify the required conclusion.

This is joint work with the Austin Chapter of Texas Action Group (http://www.cs.utexas.edu/users/tag/).

Compilation of Propositional Weighted Bases

Adnan Darwiche
Computer Science Department
University of California, Los Angeles
CA 90095, USA
e-mail: darwiche@cs.ucla.edu

Pierre Marquis
CRIL-CNRS
Université d'Artois
F-62307, Lens Cedex, France
e-mail: marquis@cril.univ-artois.fr

Abstract

In this paper, we investigate the extent to which knowledge compilation can be used to improve inference from propositional weighted bases. We present a general notion of compilation of a weighted base that is parametrized by any equivalence–preserving compilation function. Both negative and positive results are presented. On the one hand, complexity results are identified, showing that the inference problem from a compiled weighted base is as difficult as in the general case, when the prime implicates, Horn cover or renamable Horn cover classes are targeted. On the other hand, we show that the inference problem becomes tractable whenever $DNNF$-compilations are used and clausal queries are considered. Moreover, we show that the set of all preferred models of a $DNNF$-compilation of a weighted base can be computed in time polynomial in the output size. Finally, we sketch how our results can be used in model-based diagnosis in order to compute the most probable diagnoses of a system.

1 INTRODUCTION

Penalty logic is a logical framework developed by Pinkas [19; 20] and by Dupin de St Cyr, Lang and Schiex [13], which enables the representation of propositional weighted bases. A *weighted base* is a finite set

$$W = \{\langle \phi_1, k_1 \rangle, \ldots, \langle \phi_n, k_n \rangle\}.$$

Each ϕ_i is a propositional formula, and k_i is its corresponding weight, i.e., the price to be paid if the formula is violated. In penalty logic, weights are positive integers[1] or $+\infty$ and they are *additively* aggregated.

A weighted base can be considered as a compact, implicit encoding of a total pre-ordering over a set Ω of propositional worlds. Indeed, given a weighted base W, the weight of each world ω can be defined as follows:

$$K_W(\omega) \stackrel{def}{=} \sum_{\langle \phi_i, k_i \rangle \in W,\, \omega \models \neg\phi_i} k_i.$$

That is, the weight of a world is the sum of all weights associated with sentences violated by the world. One can extend the function K_W to arbitrary sentences α:

$$K_W(\alpha) \stackrel{def}{=} \min_{\omega \models \alpha} K_W(\omega).$$

Finally, $min_W(\Omega)$ denotes the most preferred worlds in Ω, those having minimal weight:

$$min_W(\Omega) \stackrel{def}{=} \{\omega \,|\, \omega \in \Omega, \forall \omega' \in \Omega\, K_W(\omega) \leq K_W(\omega')\}.$$

The weight of base W, denoted $K(W)$, is the weight of some world in $min_W(\Omega)$. Obviously enough, we have $K(W) = K_W(true)$, and $\omega \in min_W(\Omega)$ if and only if $K_W(\omega) = K(W)$.

Example 1.1 *Let $W = \{\langle a \wedge b, 2\rangle, \langle \neg b, 1\rangle\}$ be a weighted base. Let us consider the following four worlds over the variables appearing in W, $Var(W)$:*

- $\omega_1 = (a, b)$
- $\omega_2 = (a, \neg b)$
- $\omega_3 = (\neg a, b)$
- $\omega_4 = (\neg a, \neg b)$

We then have $K_W(\omega_1) = 1$, $K_W(\omega_2) = 2$, $K_W(\omega_3) = 3$, and $K_W(\omega_4) = 2$. Accordingly, we have $K(W) = 1$ and $min_W(\Omega) = \{\omega_1\}$.

[1] Floating numbers can also be used; what is important is the fact that sum is a total function over the set of (totally ordered) numbers under consideration, and that it can be computed in polynomial time.

All formulas ϕ_i associated with finite weights in a weighted base are called *soft constraints*, while those associated with the weight $+\infty$ are called *hard constraints*.

Penalty logic has some valuable connections with possibilistic logic, as well as with Dempster–Shafer theory (see [13] for details). It is also closely connected to the optimization problem WEIGHTED-MAX-SAT considered in operations research. Accordingly, several proposals for the use of weighted bases can be found in the AI literature.

One of them concerns the compact representation of preferences in a decision making setting. Indeed, in some decision making problems, models (and formulas) can be used to encode decisions. Accordingly, the weight of a model represents the disutility of a decision, and a weighted base can be viewed as an implicit representation of the set of all decisions of an agent, totally ordered w.r.t. their (dis)utility. Lafage and Lang [16] take advantage of such an encoding for group decision making. A key issue here from a computational point of view is the problem consisting in computing (one or all) element(s) from $min_W(\Omega)$.

Another suggested use of penalty logic concerns inference from inconsistent belief bases. Based on the preference information given by K_W, several inference relations from a weighted base W can be defined. Among them is skeptical inference given by $\alpha \mid\!\sim_W \beta$ if and only if every world ω that is of minimal weight among the models of α is a model of β. In this framework, propositional formulas represent pieces of (explicit) belief. The inference relation $\mid\!\sim_W$ is interesting for at least two reasons. On the one hand, it is a *comparative* inference relation, i.e., a rational inference relation satisfying supraclassicality [13]. On the other hand, weighted bases can be used to encode some well-known forms of inference from stratified belief bases $B = (B_1, \ldots, B_k)$ [21; 1; 2]. Especially, the so-called skeptical *lexicographic inference* $B \mid\!\sim_{lex}$ can be recovered as a specific case of $true \mid\!\sim_{W_B}$ for some weighted base W_B.

Example 1.2 *Let $B = (B_1, B_2)$ be a belief base interpreted under lexicographic inference, where $B_1 = \{a \vee b \vee c\}$ (the most reliable stratum) and $B_2 = \{\neg a \wedge c, \neg b \wedge c, \neg c\}$. W can associate with B the weighted base*

$$W_B = \{\langle a \vee b \vee c, 4\rangle, \langle \neg a \wedge c, 1\rangle, \langle \neg b \wedge c, 1\rangle, \langle \neg c, 1\rangle\}.$$

The unique most preferred world for W_B is $(\neg a, \neg b, c)$ that is also the only lexicographically-preferred model of B.

Weighted bases enable more flexibility than stratified belief bases (e.g., violating two formulas of weight 5 is worse than violating a single formula of weight 9, but this cannot be achieved through a simple stratification)[2].

[2]Since lexicographic inference also includes inference from consistent sub–bases that are maximal w.r.t. cardinality as a subcase (to achieve it, just put every formula of the belief base into a single stratum), the latter can also be recovered as a specific case of inference from a weighted base.

The *inference* problem from a weighted base W consists in determining whether $true \mid\!\sim_W \beta$ holds given W and β. Up to now, weighted bases have been investigated from a theoretical point of view, only. Despite their potentialities, we are not aware of any industrial application of weighted bases. There is a simple (but partial) explanation of this fact: inference (and preferred model enumeration) from weighted bases are intractable. Actually, the inference problem is known as Δ_2^p-complete [12] (even in the restricted case where queries are literals). Furthermore, it is not hard to show that computing a preferred world from $min_W(\Omega)$ is $\mathsf{F}\Delta_2^p$-complete. This implies that any of the two problems is very likely to require an unbounded polynomial number of calls to an NP oracle to be solved in polynomial time on a deterministic Turing machine.

In this paper, we investigate the extent to which *knowledge compilation* [4] can be used to improve inference from weighted bases. The key idea of compilation is pre-processing of the fixed part of the inference problem. Several knowledge compilation functions dedicated to the clausal entailment problem have been pointed out so far (e.g., [23; 15; 14; 17; 6; 24; 25; 3; 8]). The input formula is turned into a compiled one during an off-line compilation phase and the compiled form is used to answer the queries on-line. Assuming that the formula does not often change and that answering queries from the compiled form is computationally easier than answering them from the input formula, the compilation time can be balanced over a sufficient number of queries. Thus, when queries are CNF formulas, the complexity of classical inference falls from coNP-complete to P. While none of the techniques listed above can ensure the objective of enhancing inference to be reached in the worst case (because the size of the compiled form can be exponentially larger than the size of the original knowledge base – see [25; 4]), experiments have shown such approaches valuable in many practical situations [24; 3; 10].

In the following, we show how compilation functions for clausal entailment from classical formulas can be extended to clausal inference from weighted bases. Any equivalence–preserving knowledge compilation function can be considered in our framework. Unfortunately, for many target classes for such functions, including the prime implicates, Horn cover and renamable Horn cover classes, we show that the inference problem from a compiled base remains Δ_2^p-complete, even for very simple queries (literals). Accordingly, in this situation, there is no guarantee that compiling a weighted base using any of the corresponding compilation functions may help. Then

we focus on $DNNF$-compilations as introduced in [8; 9]. This case is much more favourable since the clausal inference problem becomes tractable. We also show that the preferred models of a $DNNF$-compilation of a weighted base can be enumerated in output polynomial time. Finally, we sketch how our results can be used in the model–based diagnosis framework in order to compute the most probable diagnoses of a system.

2 FORMAL PRELIMINARIES

In the following, we consider a propositional language $PROP_{PS}$ defined inductively from a finite set PS of propositional symbols, the boolean constants *true* and *false* and the connectives $\neg$, $\wedge$, $\vee$ in the usual way. L_{PS} is the set of literals built up from PS. For every formula ϕ from $PROP_{PS}$, $Var(\phi)$ denotes the symbols of PS occurring in ϕ. As mentioned before, if $W = \{\langle\phi_1, k_1\rangle, \ldots, \langle\phi_n, k_n\rangle\}$ is a weighted base, then $Var(W) = \bigcup_{i=1}^{n} Var(\phi_i)$.

Formulas are interpreted in a classical way. As evoked before, Ω $(= 2^{PS})$ denotes the set of all interpretations built up from PS. Every interpretation (world) $\omega \in \Omega$ is represented as a tuple of literals. $Mod(\phi)$ is the set of all models of ϕ.

As usual, every finite set of formulas is considered as the conjunctive formula whose conjuncts are the elements of the set. A CNF formula is a (finite) conjunction of clauses, where a clause is a (finite) disjunction of literals. A formula ϕ is Blake if and only if it is a CNF formula where each prime implicate[3] of ϕ appears as a conjunct (one representative per equivalence class). A formula is Horn CNF if and only if it is a CNF formula s.t. every clause in it contains at most one positive literal. A formula ϕ is renamable Horn CNF if and only if $\sigma(\phi)$ is a Horn CNF formula, where σ is a substitution from L_{PS} to L_{PS} s.t. $\sigma(l) = l$ for every literal l of L_{PS} except those of a set L, and for every literal l of L, $\sigma(l) = \neg l$ and $\sigma(\neg l) = l$.

We assume the reader familiar with the complexity classes P, NP, coNP and Δ_2^p of the polynomial hierarchy. $\mathsf{F}\Delta_2^p$ denotes the class of function problems associated to Δ_2^p (see [18] for details).

3 COMPILING WEIGHTED BASES

In this section, we first show how knowledge compilation techniques for improving clausal entailment can be used in order to compile weighted bases. Then, we present some complexity results showing that compiling a weighted base is not always a good idea, since the complexity of inference from a compiled base does not necessarily decrease. We specifically focus on prime implicates [23] and Horn covers and renamable Horn covers compilations [3].

[3] A prime implicate of a formula ϕ is a logically strongest clause entailed by ϕ.

3.1 A FRAMEWORK FOR WEIGHTED BASES COMPILATION

Let $W = \{\langle\phi_1, k_1\rangle, \ldots, \langle\phi_n, k_n\rangle\}$ be a weighted base. In the case where $\bigwedge_{i=1}^{n} \phi_i$ is consistent, then $K(W) = 0$ and $min_W(\Omega)$ is the set of all models of $\bigwedge_{i=1}^{n} \phi_i$. Accordingly, in this situation, inference $\mid\sim_W$ is classical entailment, so it is possible to directly use any knowledge compilation function and compiling W comes down to compile $\bigwedge_{i=1}^{n} \phi_i$. However, this situation is very specific and out of the ordinary when weighted bases are considered (otherwise, weights would be useless). A difficulty is that, in the situation where $\bigwedge_{i=1}^{n} \phi_i$ is inconsistent, we cannot compile directly this formula using any equivalence–preserving knowledge compilation function (otherwise, trivialization would not be avoided). Indeed, in this situation, $\mid\sim_W$ is not classical entailment any longer, so a more sophisticated approach is needed.

In order to compile weighted bases, it is helpful to consider weighted bases in normal form:

Definition 3.1 (Weighted bases in normal form)
A belief base $W = \{\langle\phi_1, k_1\rangle, \ldots, \langle\phi_n, k_n\rangle\}$ is in normal form *if and only if for every $i \in 1 \ldots n$, either $k_i = +\infty$ or ϕ_i is a propositional symbol.*

Every weighted base can be turned into a query–equivalent base in normal form.

Definition 3.2 (V-equivalence of weighted bases)
Let W_1 and W_2 be two weighted bases and let $V \subseteq PS$. W_1 and W_2 are V-equivalent *if and only if for every pair of sentences α and β in $PROP_V$, we have $\alpha \mid\sim_{W_1} \beta$ precisely when $\alpha \mid\sim_{W_2} \beta$.*

Accordingly, two V–equivalent weighted bases must agree on queries built up from the symbols in V. Note that a stronger notion of equivalence can be defined by requiring that both bases induce the same weight function, i.e., $K_{W_1} = K_{W_2}$ [13]. Finally, note that if K_{W_1} and K_{W_2} agree on the sentences in $PROP_V$, then W_1 and W_2 must be V–equivalent.

Proposition 3.1
Let $W = \{\langle\phi_1, k_1\rangle, \ldots, \langle\phi_n, k_n\rangle\}$ be a weighted base. If

$$H \stackrel{def}{=} \{\langle\phi_i, +\infty\rangle \mid \langle\phi_i, +\infty\rangle \in W\},$$

$$S \stackrel{def}{=} \{\langle holds_i \Rightarrow \phi_i, +\infty\rangle, \langle holds_i, k_i\rangle \mid \langle\phi_i, k_i\rangle \in W \text{ and } k_i \neq +\infty\},$$

where $\{holds_1, \ldots, holds_n\} \subseteq PS \setminus Var(W)$*, then the weighted base* $W\downarrow \stackrel{def}{=} H \cup S$ *is in normal form. Moreover,* K_W *and* $K_{W\downarrow}$ *agree on all weights of sentences in* $PROP_{Var(W)}$ *and, hence,* $W\downarrow$ *is* $Var(W)$*–equivalent to* W*.*

We will call $W\downarrow$ the normal form of W. Intuitively, the variable $holds_i$ is guaranteed to be false in any world that violates the sentence ϕ_i and, hence, that world is guaranteed to incur the penalty k_i.

Example 1.1 (Continued) The weighted base

$$W\downarrow = \{\langle holds_1 \Rightarrow (a \wedge b), +\infty\rangle,$$
$$\langle holds_2 \Rightarrow \neg b, +\infty\rangle, \langle holds_1, 2\rangle, \langle holds_2, 1\rangle\}$$

is a normal form of W as given by Proposition 3.1. The normalized weighted base $W\downarrow$ induces the following weight function:

World	$K_{W\downarrow}$
$a, b, holds_1, holds_2$	$+\infty$
$a, b, holds_1, \neg holds_2$	1
$a, b, \neg holds_1, holds_2$	$+\infty$
$a, b, \neg holds_1, \neg holds_2$	3
$a, \neg b, holds_1, holds_2$	$+\infty$
$a, \neg b, holds_1, \neg holds_2$	$+\infty$
$a, \neg b, \neg holds_1, holds_2$	2
$a, \neg b, \neg holds_1, \neg holds_2$	3
$\neg a, b, holds_1, holds_2$	$+\infty$
$\neg a, b, holds_1, \neg holds_2$	$+\infty$
$\neg a, b, \neg holds_1, holds_2$	$+\infty$
$\neg a, b, \neg holds_1, \neg holds_2$	3
$\neg a, \neg b, holds_1, holds_2$	$+\infty$
$\neg a, \neg b, holds_1, \neg holds_2$	$+\infty$
$\neg a, \neg b, \neg holds_1, holds_2$	2
$\neg a, \neg b, \neg holds_1, \neg holds_2$	3

We have $K(W\downarrow) = 1$ and

$$min_{W\downarrow}(\Omega) = \{(a, b, holds_1, \neg holds_2)\}.$$

Moreover, K_W and $K_{W\downarrow}$ agree on all sentences constructed from the variables in $\{a, b\}$.

We now discuss the compilation of a weighted base in normal form. The basic idea is to combine all hard constraints in the base into a single constraint, which preserves the weight function induced by the base. We then compile that single hard constraint using an equivalence–preserving compilation function $COMP$, that is, a function which maps each sentence α into its compiled form $COMP(\alpha)$. From here on, we will use $\widehat{W}$ to denote the conjunction of all sentences in the weighted base W that have $+\infty$ weights:

$$\widehat{W} \stackrel{def}{=} \bigwedge_{\langle \phi_i, +\infty\rangle \in W} \phi_i.$$

Definition 3.3 (Compilation of a weighted base) *Let* $W = \{\langle \phi_1, k_1\rangle, \ldots, \langle \phi_n, k_n\rangle\}$ *be a weighted base. Let* $COMP$ *be any equivalence–preserving knowledge compilation function. The* $COMP$*-compilation of* W *is the weighted base*

$$W\downarrow_{COMP} \stackrel{def}{=} \{\langle COMP(\widehat{W\downarrow}), +\infty\rangle\} \cup$$
$$\{\langle holds_i, k_i\rangle \mid \langle holds_i, k_i\rangle \in W\downarrow \text{ and } k_i \neq +\infty\}.$$

That is, to compile a weighted base W, we perform three steps. First, we compute a normal form $W\downarrow$ according to Proposition 3.1, which is guaranteed to be V-equivalent to W, where V are the variables in W. Next, we combine all of the hard constraints of $W\downarrow$ into a single hard constraint $\widehat{W\downarrow}$. Finally, we compile $\widehat{W\downarrow}$ using the function $COMP$.

Example 1.1 (Continued) We have

$$\widehat{W\downarrow} = (\neg holds_1 \vee (a \wedge b)) \wedge (\neg holds_2 \vee \neg b).$$

Accordingly, the Blake-compilation of W is

$$\{(\neg holds_1 \vee a) \wedge (\neg holds_1 \vee b) \wedge (\neg holds_2 \vee \neg b)$$
$$\wedge (\neg holds_1 \vee \neg holds_2), +\infty\rangle, \langle holds_1, 2\rangle, \langle holds_2, 1\rangle\}.$$

Given Proposition 3.1, and since $COMP$ is equivalence–preserving, we have:

Corollary 3.1 *Let* $W = \{\langle \phi_1, k_1\rangle, \ldots, \langle \phi_n, k_n\rangle\}$ *be a weighted base. Let* $COMP$ *be any equivalence–preserving knowledge compilation function.* $K_{W\downarrow_{COMP}}$ *and* K_W *agree on the sentences in* $PROP_{Var(W)}$*. Moreover,* W *and* $W\downarrow_{COMP}$ *are* $Var(W)$*-equivalent.*

3.2 SOME COMPLEXITY RESULTS

In the following, the next tractable classes of formulas, that are target classes for some existing equivalence–preserving compilation functions $COMP$, are considered:

- The *Blake* class is the set of formulas given in prime implicates normal form,
- The *Horn cover* class is the set of disjunctions of Horn CNF formulas,
- The *renamable Horn cover* class (r. Horn cover for short) is the set of the disjunctions of renamable Horn CNF formulas.

The Blake class is the target class of the compilation function $COMP_{Blake}$ described in [23]. The Horn cover class and the renamable Horn cover class are target classes for the tractable covers compilation functions given in

[3]. We shall note respectively $COMP_{Horn\ cover}$ and $COMP_{r.\ Horn\ cover}$ the corresponding compilation functions.

Accordingly, a Blake (resp. Horn cover, r. Horn cover) compiled weighted base W is defined as a weighted base in normal form whose unique hard constraint belongs to the Blake (resp. Horn cover, r. Horn cover) class.

In the next section, we will also focus on the *DNNF* class. We consider it separately because — unlike the other classes — it will lead to render tractable clausal inference from compilations.

Of course, all these compilation functions $COMP$ are subject to the limitation evoked above: in the worst case, the size of the compiled form $COMP(\Sigma)$ is exponential in the size of Σ. Nevertheless, there is some empirical evidence that some of these approaches can prove computationally valuable for many instances of the clausal entailment problem (see e.g., the experimental results given in [24; 3; 10]).

As evoked previously, knowledge compilation can prove helpful only if inference from the compiled form is computationally easier than direct inference. Accordingly, it is important to identify the complexity of inference from a compiled weighted base if we want to draw some conclusions about the usefulness of knowledge compilation in this context. Formally, we are going to consider the following decision problems:

Definition 3.4 (FORMULA $\vdash_W$)
FORMULA $\vdash_W$ *is the following decision problem:*

- **Input:** *A weighted base W and a formula β from $PROP_{PS}$.*
- **Query:** *Does $true \vdash_W \beta$ hold?*

CLAUSE $\vdash_W$ (resp. LITERAL $\vdash_W$) is the restriction of FORMULA $\vdash_W$ to the case where β is required to be a CNF formula (resp. a term).

When no restriction is put on W, FORMULA $\vdash_W$ is known as Δ_2^p-complete [12], even in the restricted LITERAL $\vdash_W$ case. Now, what if W is a *compiled* weighted base? We have identified the following results:

Proposition 3.2 (Inference from compiled weighted bases) *The complexity of* CLAUSE $\vdash_W$ *and of its restrictions to literal inference when W is a Blake (resp. Horn cover, r. Horn cover) compiled weighted base is reported in Table 1.*

Hardness results can be easily derived from results given in [5] due to the fact that (skeptical) lexicographic inference

$COMP$	CLAUSE / LITERAL $\vdash_W$
Blake	Δ_2^p-complete
Horn cover	Δ_2^p-complete
r Horn cover	Δ_2^p-complete

Table 1: Complexity of clausal inference from compiled weighted bases.

$\vdash_{lex}$ from a stratified belief base can be easily encoded as inference from a weighted base. Indeed, if m is the maximum number of formulas belonging to any stratum B_i of $B = (B_1, \ldots, B_k)$, then let $W_B = \{\langle \phi, (m+1)^{k-i} \rangle \mid \phi \in B_i\}$ (see Example 1.2 for an illustration). It is not hard to prove that $B \vdash_{lex} \beta$ if and only if $true \vdash_{W_B} \beta$.

The complexity results reported in Table 1 do not give good news: there is no guarantee that compiling a belief base using the *Blake* (or the *Horn cover* or the *r. Horn cover*) compilation function leads to improve inference since its complexity from the corresponding compiled bases is just as hard as the complexity of $\vdash_W$ in the general case.

Fortunately, it is not the case that such negative results hold for any compilation function. As we will see in the next section, $DNNF$-compilations of weighted bases exhibit a much better behaviour.

4 COMPILING WEIGHTED BASES USING DNNF

In this section, we focus on $DNNF$-compilations of weighted bases. After a brief recall of what $DNNF$-compilation is, we show that $DNNF$-compilations support two important computational tasks in polynomial time, especially preferred model enumeration and clausal inference.

4.1 A GLIMPSE AT THE $DNNF$ LANGUAGE

The $DNNF$ language is the set of sentences, defined as follows [8]:

Definition 4.1 (DNNF) *Let PS be a finite set of propositional variables. A sentence in $DNNF$ is a rooted, directed acyclic graph (DAG) where each leaf node is labeled with true, false, x or $\neg x$, $x \in PS$; each internal node is labeled with $\wedge$ or $\vee$ and can have arbitrarily many children. Moreover, the* decomposability *property is satisfied: for each conjunction C in the sentence, the conjuncts of C do not share variables.*

Figure 1 depicts a $DNNF$ of the hard constraint $\widehat{W\downarrow}$, where W is the weighted base given in Example 1.1. Note here that $W\downarrow$ is the normal form constructed from W ac-

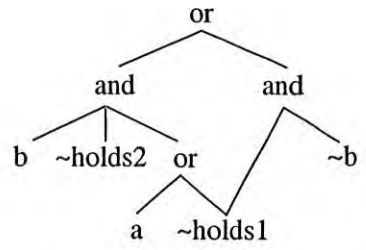


Figure 1: A sentence in $DNNF$.

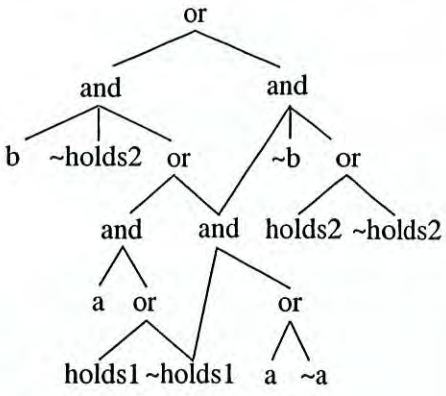


Figure 2: A sentence in smooth $DNNF$.

cording to Proposition 3.1, and $\widehat{W\downarrow}$ is the conjunction of all hard constraints in $W\downarrow$.

An interesting subset of $DNNF$ is the set of *smooth* $DNNF$ sentences [9]:

Definition 4.2 (Smooth DNNF) *A $DNNF$ sentence satisfies the* smoothness *property if and only if for each disjunction C in the sentence, each disjunct of C mentions the same variables.*

Interestingly, every $DNNF$ sentence can be turned into an equivalent, smooth one in polynomial time [9].

For instance, Figure 2 depicts a smooth $DNNF$ which is equivalent to the $DNNF$ in Figure 1. Note that for readibility reasons some leaf nodes are duplicated in the figure.

Among the various tasks that can be achieved in a tractable way from a $DNNF$ sentence are *conditioning*, *clausal entailment*, *forgetting* and *model enumeration* (given that the $DNNF$ is smooth) [9; 11].

4.2 TRACTABLE QUERIES

Given a weighted base W, and given a $DNNF$–compilation of W, we now show how the compilation can be used to represent the preferred models of W as a $DNNF$ in polynomial time.

Definition 4.3 (Minimization of a weighted base) *A* minimization *of a weighted base W is a propositional formula Δ where the models of Δ are* $\min_W(\Omega)$.

Note that this notion generalizes the notion of minimization of a propositional formula ϕ reported in [8], for which the preferred models are those containing a maximal number of variables assigned to true. Such a minimization can be easily achieved in a weighted base setting by considering the base $\{\langle\phi, +\infty\rangle\} \cup \bigcup_{x \in Var(\phi)} \{\langle x, 1\rangle\}$.

Definition 4.4 (Minimization of $DNNF$-compilation) *Let W be a $DNNF$-compilation of a weighted base. Let $\langle\alpha, +\infty\rangle$ be the single hard constraint in W, where α is a $DNNF$ sentence. Suppose that α is also smooth.*

- *We define $k(\alpha)$ inductively as follows:*
  - $k(true) \stackrel{def}{=} 0$ *and* $k(false) \stackrel{def}{=} +\infty$.
  - *If α is a literal, then:*
    - *If $\alpha = \neg holds_i$, then $k(\alpha) \stackrel{def}{=} k_i$, where $\langle holds_i, k_i\rangle \in W$.*
    - *Otherwise, $k(\alpha) \stackrel{def}{=} 0$.*
  - $k(\alpha = \bigvee_i \alpha_i) \stackrel{def}{=} min_i k(\alpha_i)$.
  - $k(\alpha = \bigwedge_i \alpha_i) \stackrel{def}{=} \sum_i k(\alpha_i)$.
- *We define $min(\alpha)$ inductively as follows:*
  - *If α is a literal or a boolean constant, then $min(\alpha) \stackrel{def}{=} \alpha$.*
  - $min(\alpha = \bigvee_i \alpha_i) \stackrel{def}{=} \bigvee_{k(\alpha_i)=k(\alpha)} min(\alpha_i)$.
  - $min(\alpha = \bigwedge_i \alpha_i) \stackrel{def}{=} \bigwedge_i min(\alpha_i)$.

We have the following result:

Proposition 4.1 *Let W be a $DNNF$-compilation of a weighted base. Let $\langle\alpha, +\infty\rangle$ be the single hard constraint in W, where α is a smooth $DNNF$ sentence. Then $min(\alpha)$ is a smooth $DNNF$ and is a minimization of W.*

Figure 3 depicts the weight $k(\alpha)$ of every subformula α of the smooth $DNNF$ sentence given in Figure 2. Figure 4 (left) depicts the minimization of the $DNNF$ in Figure 3. Figure 4 (right) depicts a simplification of this minimized $DNNF$ which has a single model.

Since $min(\alpha)$ can be computed in time polynomial in the size of $DNNF$ α, and since clausal entailment can be done in time linear in the size of α [8], we have:

Corollary 4.1 *The clausal inference problem* CLAUSE $\vdash_W$ *from $DNNF$-compilations of weighted bases is in* P.

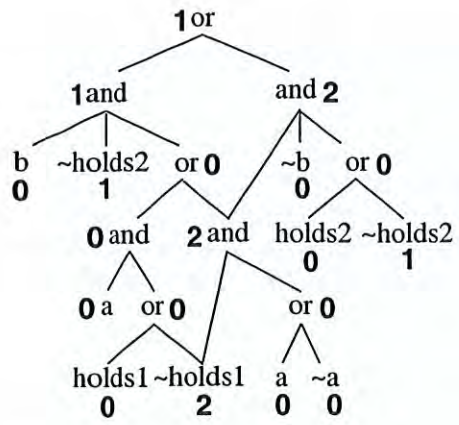


Figure 3: Weights on a smooth $DNNF$ sentence.

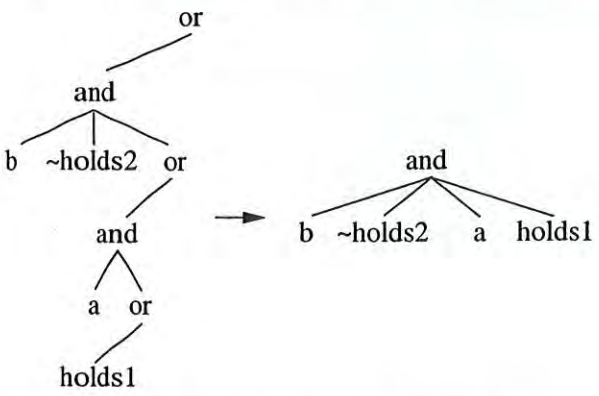


Figure 4: Minimization of a smooth $DNNF$ sentence.

Since model enumeration can be done in output polynomial time from a smooth $DNNF$, we also have:

Corollary 4.2 *The preferred model enumeration problem from $DNNF$-compilations of weighted bases can be solved in output polynomial time.*

5 APPLICATION TO MODEL-BASED DIAGNOSIS

We now briefly sketch how the previous results can be used to compute the set of most probable diagnoses of a system in time polynomial in the size of system description and the output size. The following results generalize those given in [7; 9] to the case where the probability of failure of components is available.

We first need to briefly recall what a consistency-based diagnosis of a system is [22]:

Definition 5.1 (Consistency-based diagnosis)

- *A* diagnostic problem $\mathcal{P} = \langle SD, OK, OBS \rangle$ *is a triple consisting of:*
  - *a formula SD from $PROP_{PS}$, the system description.*
  - *a finite set $OK = \{ok_1, \ldots, ok_n\}$ of propositional symbols. "ok_i is true" means that the component i of the system to be diagnosed is not faulty.*
  - *OBS is a term, typically gathering the inputs and the outputs of the system.*
- *A* consistency-based *diagnosis Δ for $\mathcal{P}$ is a complete OK-term (i.e., a conjunction of literals built up from OK in which every ok_i occurs either positively or negatively) s.t. $\Delta \wedge SD \wedge OBS$ is consistent.*

Because a system can have a number of diagnoses that is exponential in the number of its components, preference criteria are usually used to limit the number of candidates. The most current ones consist in keeping the diagnoses containing as few negative OK-literals as possible (w.r.t. set inclusion or cardinality).

When the *a priori* probability of failure of components is available (and such probabilities are considered independent), the most probable diagnoses for $\mathcal{P}$ can also be preferred. Such a notion of preferred diagnosis generalizes the one based on minimality w.r.t. cardinality (the latter corresponds to the case where the probability of failure of components is uniform and $< \frac{1}{2}$).

Interestingly, the most probable diagnoses for $\mathcal{P}$ can be enumerated in output polynomial time as soon as a smooth $DNNF$-compilation $\mathcal{P}_{DNNF}$ corresponding to $\mathcal{P}$ has been derived first.

Definition 5.2 (Compilation of a diagnostic problem) *Let $\mathcal{P}$ be a diagnostic problem for which the a priori probability of failure p_i of any component i is available.*

$$\mathcal{P}_{DNNF} \stackrel{def}{=} \{\langle DNNF(SD \mid OBS), +\infty \rangle\} \cup \{\langle ok_i, log\ p_i \rangle \mid ok_i \in OK\}$$

is the smooth $DNNF$-compilation associated with $\mathcal{P}$.

In this definition, $SD \mid OBS$ denotes the conditioning of SD by the term OBS, i.e., the formula obtained by replacing in SD every variable x by $true$ (resp. $false$) if x (resp. $\neg x$) is a positive (resp. negative) literal of OBS.

The log transformation performed here enables to compute the log of the probability of a diagnosis Δ as $\sum_{\neg ok_i \in \Delta} log\ p_i$. Because log is strictly nondecreasing,

the induced preference ordering between diagnoses is preserved.

Proposition 5.1

- *$K(\mathcal{P}_{DNNF})$ is the log of the probability of any most probable diagnosis for $\mathcal{P}$.*
- *The most probable diagnoses for $\mathcal{P}$ are the the models of $Forget(min(DNNF(SD \mid OBS)), PS \setminus OK)$[4].*

An important point is that $\mathcal{P}_{DNNF}$ does not have to be re-compiled each time the observations change; indeed, a $DNNF$ sentence $DNNF(SD \mid OBS)$ equivalent to the conditioning of SD by the observations OBS can be computed as $DNNF(SD) \mid OBS$, the conditioning of a $DNNF$ sentence equivalent to SD by OBS. Since conditioning can be achieved in linear time from a $DNNF$ formula, it is sufficient to compile only the system description SD (that is the fixed part of the diagnostic problem) so to compute $DNNF(SD)$ instead of $DNNF(SD \mid OBS)$.

Because (1) forgetting variables in a $DNNF$ formula can be done in polynomial time [8] and (2) the models of a smooth $DNNF$ formula can be generated in time polynomial in the output size [9], we obtain that:

Corollary 5.1 *The most probable diagnoses for a diagnostic problem $\mathcal{P}$ can be enumerated in time polynomial in the size of $\mathcal{P}_{DNNF}$.*

6 CONCLUSION

In this paper, we have studied how existing knowledge compilation functions can be used to improve inference from propositional weighted bases. Both negative and positive results have been put forward. On the one hand, we have shown that the inference problem from a compiled weighted base is as difficult as in the general case, when prime implicates, Horn cover or renamable Horn cover target classes are considered. On the other hand, we have shown that this problem becomes tractable whenever $DNNF$-compilations are used. Finally, we have sketched how our results can be used in model-based diagnosis in order to compute the most probable diagnoses of a system.

Acknowledgements

The first author has been partially supported by NSF grant IIS-9988543 and MURI grant N00014-00-1-0617. The second author has been partly supported by the IUT de Lens, the Université d'Artois, the Région Nord/Pas-de-Calais under the TACT-TIC project, and by the European Community FEDER Program.

[4]For every formula ϕ and every set of variables X, $Forget(\phi, X)$ denotes the logically strongest consequence of ϕ that is independent from X, i.e., that can be turned into an equivalent formula in which no variable from X occurs.

References

[1] S. Benferhat, C. Cayrol, D. Dubois, J. Lang, and H. Prade. Inconsistency management and prioritized syntax-based entailment. In *Proc. of the 13^{th} International Joint Conference on Artificial Intelligence (IJCAI'93)*, pages 640–645, Chambéry, 1993.

[2] S. Benferhat, D. Dubois, and H. Prade. How to infer from inconsistent beliefs without revising. In *Proc. of the 14^{th} International Joint Conference on Artificial Intelligence (IJCAI'95)*, pages 1449–1455, Montreal, 1995.

[3] Y. Boufkhad, E. Grégoire, P. Marquis, B. Mazure, and L. Saïs. Tractable cover compilations. In *Proc. of the 15^{th} International Joint Conference on Artificial Intelligence (IJCAI'97)*, pages 122–127, Nagoya, 1997.

[4] M. Cadoli and F.M. Donini. A survey on knowledge compilation. *AI Communications*, 10:137–150, 1997. (printed in 1998).

[5] S. Coste-Marquis and P. Marquis. Compiling stratified belief bases. In *Proc. of the 14^{th} European Conference on Artificial Intelligence (ECAI'00)*, pages 23–27, Berlin, 2000.

[6] M. Dalal. Semantics of an anytime family of reasoners. In *Proc. of the 12^{th} European Conference on Artificial Intelligence (ECAI'96)*, pages 360–364, Budapest, 1996.

[7] A. Darwiche. Compiling devices: a structure-based approach. In *Proc. of the 6^{th} International Conference on Knowledge Representation and Reasoning (KR'98)*, pages 156–166, Trento, 1998.

[8] A. Darwiche. Compiling knowledge into decomposable negation normal form. In *Proc. of the 16^{th} International Joint Conference on Artificial Intelligence (IJCAI'99)*, pages 284–289, Stockholm, 1999.

[9] A. Darwiche. Decomposable negation normal form. *Journal of the Association for Computing Machinery*, 48(4):1–42, 2001.

[10] A. Darwiche. A Compiler for Deterministic Decomposable Negation Normal Form. Technical Report D-125, Computer Science Department, UCLA, Los Angeles, Ca 90095, 2002. To appear in AAAI'02.

[11] A. Darwiche and P. Marquis. A perspective on knowledge compilation. In *Proc. of the* 17^{th} *International Joint Conference on Artificial Intelligence (IJCAI'99)*, pages 175–182, Seattle (WA), 2001.

[12] F. Dupin de St Cyr. *Gestion de l'évolutif et de l'incertain en logiques pondérées*. PhD thesis, Université Paul Sabatier, 1996.

[13] F. Dupin de St Cyr, J. Lang, and Th. Schiex. Penalty logic ans its link with Dempster-Shafer theory. In *Proc. of the* 10^{th} *Conference on Uncertainty in Artificial Intelligence (UAI'94)*, pages 204–211, 1994.

[14] R. Dechter and I. Rish. Directional resolution: the Davis-Putnam procedure, revisited. In *Proc. of the* 4^{th} *International Conference on Knowledge Representation and Reasoning (KR'94)*, pages 134–145, Bonn, 1994.

[15] A. del Val. Tractable databases: how to make propositional unit resolution complete through compilation. In *Proc. of the* 4^{th} *International Conference on Knowledge Representation and Reasoning (KR'94)*, pages 551–561, Bonn, 1994.

[16] C. Lafage and J. Lang. Logical representation of preferences for group decision making. In *Proc. of the* 7^{th} *International Conference on Knowledge Representation and Reasoning (KR'00)*, pages 457–468, Breckenridge (CO), 2000.

[17] P. Marquis. Knowledge compilation using theory prime implicates. In *Proc. of the* 14^{th} *International Joint Conference on Artificial Intelligence (IJCAI'95)*, pages 837–843, Montreal, 1995.

[18] Ch. Papadimitriou. *Computational complexity*. Addison–Wesley, 1994.

[19] G. Pinkas. Propositional nonmonotonic reasoning and inconsistency in symmetric neural networks. In *Proc. of the* 12^{th} *International Joint Conference on Artificial Intelligence (IJCAI'91)*, pages 525–530, 1991.

[20] G. Pinkas. Reasoning, nonmonotonicity and learning in connectionnist networks that capture propositional knowledge. *Artificial Intelligence*, 77:203–247, 1995.

[21] G. Pinkas and R.P. Loui. Reasoning from inconsistency: a taxonomy of principles for resolving conflict. In *Proc. of the* 3^{rd} *International Conference on Knowledge Representation and Reasoning (KR'92)*, pages 709–719, Cambridge (MA), 1992.

[22] R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32:57–95, 1987.

[23] R. Reiter and J. de Kleer. Foundations of assumption-based truth maintenance systems: preliminary report. In *Proc. of the* 6^{th} *National Conference on Artificial Intelligence (AAAI'87)*, pages 183–188, Seattle (WA), 1987.

[24] R. Schrag. Compilation for critically constrained knowledge bases. In *Proc. of the* 13^{th} *National Conference on Artificial Intelligence (AAAI'96)*, pages 510–515, Portland (OR), 1996.

[25] B. Selman and H.A. Kautz. Knowledge compilation and theory approximation. *Journal of the Association for Computing Machinery*, 43:193–224, 1996.

Linking Makinson and Kraus-Lehmann-Magidor preferential entailments

Yves Moinard
IRISA, Campus de Beaulieu,
35042 RENNES-Cedex FRANCE,
tel.: (33) 2 99 84 73 13,
e-mail: moinard@irisa.fr

Abstract

About ten years ago, various notions of preferential entailment have been introduced. The main reference is a paper by Kraus, Lehmann and Magidor (KLM), one of the main competitor being a more general version defined by Makinson (MAK). These two versions have already been compared, but it is time to revisit these comparisons. Here are our three main results: (1) These two notions are equivalent, provided that we restrict our attention, as done in KLM, to the cases where the entailment respects logical equivalence (on the left and on the right). (2) A serious simplification of the description of the fundamental cases in which MAK is equivalent to KLM, including a natural passage in both ways. (3) The two previous results are given for preferential entailments more general than considered in some of the original texts, but they apply also to the original definitions and, for this particular case also, the models can be simplified.

1 INTRODUCTION

Here is one possible presentation of preferential entailments: We are given some knowledge, represented as a set of logical formulas. This set can be associated with various kinds of objects, providing its "semantics": it can be associated with its set of models, or equivalently in the propositional case, with the set of the complete theories which entail the formulas. Or it can be associated with the set of theories (complete or not) which entail the formulas. Then we are given a binary relation among these objects, and we keep only the objects which are "preferred" (meaning minimal) for this relation. We get a stronger set of formulas, deduced "by default": we get also all the formulas associated with this reduced set of objects. This allows to reason in a non monotonic way, since augmenting the knowledge may invalidate previous conclusions. Indeed, some objects may become minimal in the smaller set associated with the new knowledge. We can allow more flexibility by considering copies of models, or copies of theories, defining the relation among these sets of copies. We get then four kinds of preferential entailments, called KLM below, which have been introduced by Kraus, Lehmann and Magidor (1990) in [7] ([7] requires some conditions on the relation, but adding these conditions is straightforward in our results).

Makinson (1994, first version in 1989) has defined a more general version [8], called MAK here. An unstructured "semantics" is defined simply by a satisfaction relation from some set of objects to the set of formulas, without any condition. It is then useless to consider sets of these objects instead of singletons: its suffices to define directly as our starting set, the set of sets that we would want to consider. Also, since

nothing prevents two different objects from being associated to the same set of formulas, it is useless to consider copies of objects. A drawback of this simple definition is that the notion of classical deduction is lost. The "entailment" can be highly non standard, departing from classical logic and natural ways of reasoning. We can e.g. deduce $A \wedge B$ without deducing A. As a bonus, we can consider as our starting logic basically any non classical logic.

If we want to compare the two notions, we must determine which non standard behavior we admit in the "semantics" defining the preferential entailment. If we want to extend KLM entailment in order to deal with any situation accessible to MAK entailment, all we have to do is to admit any unconstrained relation $\models$ between the set of states and the set of formulas. We can even restrict our attention to the simplest of the four cases, defining the relation directly on the set of the objects describing the semantics. Thus, the interesting point is the other direction. We show that, providing classical equivalence is respected, MAK entailment is equivalent to KLM entailment. We describe a simple subclass of MAK entailment, which includes all the cases where MAK entailment respects classical equivalence, and for which it is easy to describe a passage from MAK formulation to KLM formulation and back. We improve previous results obtained by Dix and Makinson (1992) in [4] and by Voorbraak (1993) in [13]. With respect to [4], the description of the subclass of MAK is much simpler. Thanks to recent results on preferential entailments, we establish equivalence between the two formalisms in all the cases where it is possible, namely when MAK respects classical equivalence.

In Section 2 we introduce the notations and the logical pre-requisite necessary for this text. In Section 3 we remind the definitions and main properties of the kind of preferential entailments considered here, giving our results in Section 4.

2 FRAMEWORK

• $\mathbf{L}$, φ, $\mathcal{T}$: We work in a propositional language $\mathbf{L}$, and we use the same denotation $\mathbf{L}$ for its set of formulas. Letters φ, ψ denote *formulas* (identified with their equivalence class). Letters $\mathcal{T}$ or $\mathcal{C}$ denote sets of formulas.

• V, $\mathbf{M}$, μ, $\mathcal{P}(E)$, $\mu \models \cdots$: $V(\mathbf{L})$ (vocabulary), denotes a set of propositional symbols and μ denotes an *interpretation for* $\mathbf{L}$, identified with the subset of $V(\mathbf{L})$ that it satisfies. The *satisfaction relation* is denoted by $\models$, $\mu \models \varphi$ and $\mu \models \mathcal{T}$ being defined classically. For any set E, $\mathcal{P}(E)$ denotes the set of its subsets. The set $\mathcal{P}(V(\mathbf{L}))$ of the interpretations for $\mathbf{L}$ is denoted by $\mathbf{M}$. A *model* of $\mathcal{T}$ is an interpretation μ such that $\mu \models \mathcal{T}$. The sets of the models of $\mathcal{T}$ and φ are denoted respectively by $\mathbf{M}(\mathcal{T})$ and $\mathbf{M}(\varphi)$.

• $\mathcal{T} \models \cdots, Th(\mathcal{T}), \mathbf{T}$: $\mathcal{T} \models \varphi$ and $\mathcal{T} \models \mathcal{T}_1$ are defined classically. A *theory* is a subset of $\mathbf{L}$ closed for deduction, and $\mathbf{T}$ denotes the set $\{\mathcal{T} \subseteq \mathbf{L} \ / \ \mathcal{T} = Th(\mathcal{T})\}$ of the theories of $\mathbf{L}$.

• $\mathbf{M}_1 \models \cdots, Th(\mu), Th(\mathbf{M}_1)$: A theory $\mathcal{C} \in \mathbf{T}$ is *complete* if $\forall \varphi \in \mathbf{L}$, $\varphi \in \mathcal{C}$ iff $\neg\varphi \notin \mathcal{C}$. The set $Th(\mu) = \{\varphi \in \mathbf{L} \ / \ \mu \models \varphi\}$ of the formulas satisfied by μ is the *theory of* μ. For any subset $\mathbf{M}_1$ of $\mathbf{M}$, $\mathbf{M}_1 \models \mathcal{T}$ means $\mu \models \mathcal{T}$ for any $\mu \in \mathbf{M}_1$ and the *theory of* $\mathbf{M}_1$ is the set $Th(\mathbf{M}_1) = \{\varphi \in \mathbf{L} \ / \ \mathbf{M}_1 \models \varphi\}$ [thus $Th(\mathbf{M}_1) = \bigcap_{\mu \in \mathbf{M}_1} Th(\mu)$]. This ambiguous use of Th and of $\models$ (applied to sets of formulas or interpretations) is usual. For any $\mathcal{T} \in \mathbf{T}$, we get $\mathcal{T} = \bigcap_{\mathcal{T}_i \in \mathbf{T},\ \mathcal{T}_i \models \mathcal{T}} \mathcal{T}_i = \bigcap_{\mathcal{C}_i \in \mathbf{T},\ \mathcal{C}_i \text{ complete},\ \mathcal{C}_i \models \mathcal{T}} \mathcal{C}_i$. The set of all the complete theories is in a natural one-to-one mapping with $\mathbf{M}$: For any $\mu \in \mathbf{M}$, $Th(\mu)$ is complete, and for any complete $\mathcal{C} \in \mathbf{T}$, $\mathbf{M}(\mathcal{C})$ is a singleton $\{\mu\} \subseteq \mathbf{M}$. For any $\mathcal{T} \subseteq \mathbf{L}$, we get $\mu \models \mathcal{T}$ iff $Th(\mu) \models \mathcal{T}$ and for any $\mathbf{M}_1 \subseteq \mathbf{M}$, $\mathbf{M}_1 \models \mathcal{T}$ iff $Th(\mathbf{M}_1) \models \mathcal{T}$.

3 PREFERENTIAL ENTAILMENTS

3.1 KLM ENTAILMENT

Since their introduction [7], these kinds of preferential entailment have been extensively studied. As [4] remarks, "the use of the term *preferential* is [..] rather anarchic [...]". The situation has not really improved since these "early years", however, it is clear that now the word

is not restricted to the "cumulative cases" as done in [7]. The expression "preferential entailment" was first introduced by Shoham (1988) in [12], and then regularly generalized and/or modified. The basic idea however is still the same: we consider a set of objects describing the semantics, and a binary relation $\prec$ on this set of objects. We get a "preferential semantics" in which only the objects, associated to a set of formulas, which are minimal for $\prec$, are considered. The definitions we give can be found in e.g. [7, Definitions 3.10, 3.13] ("single formula version") and [6, Definitions 4.26–29] ("theory version", only version considered here), with some modifications which have already been considered in e.g. [4, 2, 5]. These modifications are either cosmetic, or consist in dropping some special condition imposed in the original text to the relation $\prec$, since (1) we do not need these restrictions, and (2) our study can accommodate in a straightforward way these restrictions.

Definition 3.1 *A* KLM model *is a triple* $\mathbf{S} = (S, l, \prec)$, *where* S *is a set, the elements of which are called* states, l *is a function* $S \to \mathcal{P}(\mathbf{M})$ *that labels every state with a set of interpretations and* $\prec$ *is a binary relation on* S, *called a* preference relation.

We define a satisfaction relation $\models$: *for any* $s \in S$, $s \models \varphi$ *whenever* $l(s) \models \varphi$ *and, for any* $\mathcal{T} \subseteq \mathbf{L}$, $s \models \mathcal{T}$ *whenever* $l(s) \models \mathcal{T}$. *For any set of formulas* $\mathcal{T} \subseteq \mathbf{L}$, *we define* $S(\mathcal{T}) = \{s \in \mathbf{S} \ / \ s \models \mathcal{T}\}$ *and the set* $S_\prec(\mathcal{T})$ *of the states in* $S(\mathcal{T})$ *which are* minimal for $\prec$ *by* $S_\prec(\mathcal{T}) = \{s \in S(\mathcal{T}) \ / \ s' \prec s \text{ for no } s' \in S(\mathcal{T})\}$.

$S(\mathcal{T})$ is $\widehat{\mathcal{T}}$ in the original texts. Notice that, as noted by Bochman (1999) in [1], we can replace the set $l(s)$ of interpretations by a theory, precisely the theory $Th(l(s))$, and we will in fact generally prefer this formulation, where l is a mapping $S \to \mathbf{T}$ instead of $S \to \mathcal{P}(\mathbf{M})$. Also, we drop here the *consistency of states condition* $l(s) \neq \emptyset$ (or alternatively $l(s) \neq \mathbf{L}$ if we consider labelling with theories) which appears in the original definitions. As explained below, this condition is unnecessary.

The role of l is to allow *"copies of" sets of interpretations* (or alternatively *"copies" of theories*), since various states can be mapped by l to the same object.

Definition 3.2 *Let us call an* entailment relation, *any relation* $\mid\!\sim$ *in* $\mathcal{P}(\mathbf{L}) \times \mathbf{L}$. *Any entailment relation can be extended into a relation in* $\mathcal{P}(\mathbf{L}) \times \mathcal{P}(\mathbf{L})$ *by defining* $\mathcal{T} \mid\!\sim \mathcal{T}'$ *as* $\mathcal{T} \mid\!\sim \varphi$ *for any* $\varphi \in \mathcal{T}'$.

From any entailment relation, we can define a mapping C *from* $\mathcal{P}(\mathbf{L})$ *to itself, called an* entailment, *as follows:* $C(\mathcal{T}) = \{\varphi \in \mathbf{L} \ / \ \mathcal{T} \mid\!\sim \varphi\}$.

Definition 3.3 *[7] A* KLM entailment relation $\mid\!\sim_\prec^{KLM}$ *is defined as follows from a KLM model* $\mathbf{S}$: *for any* $\mathcal{T} \cup \{\varphi\} \subseteq \mathbf{L}$, $\mathcal{T} \mid\!\sim_\prec^{KLM} \varphi$ *whenever* $s \models \varphi$ *for any* $s \in S_\prec(\mathcal{T})$. *We write also* $\varphi \in C_{KLM}(\mathcal{T})$ *instead of* $\mathcal{T} \mid\!\sim_\prec^{KLM} \varphi$ *and call the entailment* C_{KLM} *a* KLM preferential entailment, *or a* KLM entailment *for short.*

Definition 3.4 *A* pre-circumscription f *(in* $\mathbf{L}$*) is an extensive (i.e.,* $f(\mathcal{T}) \supseteq \mathcal{T}$ *for any* $\mathcal{T}$*) mapping from* $\mathbf{T}$ *to* $\mathbf{T}$. *For any subset* $\mathcal{T}$ *of* $\mathbf{L}$, *we use the abbreviation* $f(\mathcal{T}) = f(Th(\mathcal{T}))$, *assimilating a pre-circumscription to a particular extensive entailment. We write* $f(\varphi)$ *for* $f(\{\varphi\}) = f(Th(\varphi))$.

Thus, we call here pre-circumscription any entailment which respects full logical equivalence and which is extensive. By "respects full logical equivalence", we mean that, if $\mathcal{T}_1$ and $\mathcal{T}_2$ are two logically equivalent sets of formulas [i.e. $Th(\mathcal{T}_1) = Th(\mathcal{T}_2)$], then (1) ("left side") $f(\mathcal{T}_1) = f(\mathcal{T}_2)$, and (2) ("right side") $\mathcal{T}_1 \subseteq f(\mathcal{T})$ iff $\mathcal{T}_2 \subseteq f(\mathcal{T})$. The "right side" is equivalent to "right weakening": if $\mathcal{T}_1 \models \varphi$ and $\mathcal{T}_1 \subseteq f(\mathcal{T})$, then $\varphi \in f(\mathcal{T})$.

Definition 3.5 *An entailment* C *satisfies (CT),* cumulative transitivity, *also known as "cut", if*

for any $\mathcal{T}' \subseteq C(\mathcal{T})$, *we get* $C(\mathcal{T} \cup \mathcal{T}') \subseteq C(\mathcal{T})$.

Here are the two main (and characterizing) properties of KLM entailments:

Property 3.6 *[7, 8] Any KLM entailment* C_{KLM} *is a pre-circumscription satisfying* (CT).

Property 3.7 *[13, 9] Any pre-circumscription satisfying* (CT) *is a KLM entailment.*

Particular KLM models can be considered. The three kinds described now originate also from [7], where no special names are given. Let $\mathbf{S} = (S, l, \prec)$ be a KLM model.

Definition 3.8 *1. If $S = \mathcal{P}(\mathbf{M})$ (or equivalently, under the alternative formulation in terms of theories, $S = \mathbf{T}$) and $l =$ identity, then* **S** *is a* simplified *(or* unlabelled*) KLM model.*

2. If each $l(s)$ is a singleton in $\mathcal{P}(\mathbf{M})$ (or equivalently a complete theory), then **S** *is a* singular *KLM model.*

3. If **S** *is simplified and singular, then* **S** *is a* strongly singular *KLM model.*

With the unrestricted case, we get then four kinds of models, which could give rise to four kinds of KLM entailments. It happens ([9], this result could also be extracted from an independent work by Voorbraak [13]) that we can without lack of generality restrict our attention to simplified KLM models: in the proof of Property 3.7, we can easily get a simplified KLM model. Thus, we do not really need to use "states" in KLM models. Notice that in the particular case of a singular KLM model, we generally cannot suppress the states if we want to keep only singletons in the image $l(S)$ of l (as shown in a very simple finite example in [7, p.193], which applies here): we cannot suppress the states without leaving this attractive particular case[1]. This means that if we start from a preference relation defined in a set of copies of interpretations, we cannot always get an equivalent relation defined directly on the set **M** of the interpretations. This feature is a good motivation for using states, but only in the case of singular models. Also we cannot express any KLM entailment thanks to a singular KLM model (a small finite counter-example in [7, proof of Lemma 4.5] applies here)[2]. This means that if we start from a preference relation defined on a set of copies of sets of interpretations (or equivalently of copies of theories), then we can find an equivalent relation defined directly on the set $\mathcal{P}(\mathbf{M})$ (or $\mathbf{T}$), but we generally cannot define the relation on the set **M** of interpretations or even on a set of copies of interpretations. Thus, we get exactly three kinds of KLM preferential entailments (see their syntactical characterizations in [11]), instead of four. A consequence of this reduction to simplified models is that any singular model is equivalent to a simplified model, which is not absolutely obvious from the definitions ("equivalent", meaning here giving rise to the same preferential entailment).

[1] Even in this case, the states can be suppressed, providing we enlarge the vocabulary of the initial language **L** in such a way that each different state gives rise to a different interpretation in the new language: this method is introduced by Costello (1998) in [3] for the cumulative and finite case

[2] Again, this limitation can be overcome [10], at least in the finite case: at the price of a severe modification of the vocabulary of the initial language **L**, any KLM preferential entailment can be expressed in terms of a strictly singular KLM model.

3.2 MAK ENTAILMENT

Makinson considers an entailment more general than its KLM counterpart, with a simpler definition. The price is that this notion leaves classical consequence altogether, getting a highly non standard preferential entailment in which we can conclude $A \wedge B$ without concluding B, and in which what we conclude from $A \wedge B$ is not related to what we conclude from $\{A, B\}$. This can be useful if we want to extend the notion of preferential entailment to non classical logics. However, if we want to stay in our good old classical way of reasoning, this is rather confusing. In any case, a fair comparison with KLM definitions needs to equate the ways we want to reason at first. This is what we will do in Section 4, after giving the definitions now.

Definition 3.9 *[8] A* MAK model *is a triple $\mathbf{S} = (S, \models, \prec)$ where S is a set, the elements of which are called* states, *$\prec$ is a binary relation on S, called a* preference relation *(till now, this is as in KLM models Definition 3.1) and where $\models$ is any* satisfaction relation *on S. We write $s \models_{\prec} \varphi$ [respectively $s \models_{\prec} \mathcal{T}$] whenever $s \models \varphi$ [respectively $s \models \mathcal{T}$, i.e. $s \models \varphi$ for any $\varphi \in \mathcal{T}$]*

and for no $s' \in S$ such that $s' \prec s$ we have $s' \models \varphi$ [respectively $s' \models \mathcal{T}$].

A MAK entailment relation $\vdash_{\prec}^{MAK}$ *is defined as follows: For any set of formulas $\mathcal{T} \cup \{\varphi\}$, $\mathcal{T} \vdash_{\prec}^{MAK} \varphi$ whenever $s \models \varphi$ for all $s \in S$ satisfying $s \models_{\prec} \mathcal{T}$.*

A MAK preferential entailment, *or* MAK entailment *for short, is an entailment C_{MAK} defined by a MAK entailment relation: $C_{MAK}(\mathcal{T}) = \{\varphi \in \mathbf{L} / \mathcal{T} \vdash_{\prec}^{MAK} \varphi\}$.*

The names KLM model and MAK model are from [4]. What makes this short definition so powerful is that no condition is required for $\models$. This makes the preferential entailment very different from what we expect for an "entailment". C_{MAK} is far from being a pre-circumscription: if $\mathcal{T}_1$ and $\mathcal{T}_2$ are classically equivalent, we do not know anything about $C_{MAK}(\mathcal{T}_2)$ when we know $C_{MAK}(\mathcal{T}_1)$. Moreover, we almost need an extensive description of all the sets $C_{MAK}(\mathcal{T})$, since they are not classical theories. If we drop the identification between a formula and its equivalence class, we can even consider logics where $C_{MAK}(A \wedge B)$ is different from $C_{MAK}(B \wedge A)$. It may even seem strange that such a formalism has interesting properties, however we get cumulative transitivity:

Property 3.10 *[8] Any MAK entailment is extensive and satisfies (CT).*

This implies also *idempotence* (Makinson, [8]): $C_{MAK}(C_{MAK}(\mathcal{T})) = C_{MAK}(\mathcal{T})$. Nevertheless, the significance of (CT) for such a non standard "entailment" is far from being as great as when we deal only with pre-circumscriptions.

Before making a comparison between MAK and KLM, let us give a few natural definitions, which extend to $(S, \models)$ what is usually done with classical interpretations $(\mathbf{M}, \models)$.

Definitions 3.11 *Let $(S, \models, \prec)$ be a MAK model. As in Definition 3.1, for $s \in S$, $\mathcal{T} \subseteq \mathbf{L}$, and $\varphi \in \mathbf{L}$, $s \models \mathcal{T}$ means $s \models \varphi$ for any $\varphi \in \mathcal{T}$, $S(\mathcal{T}) = \{s \in S / s \models \mathcal{T}\}$* [and $S(\varphi) = S(\{\varphi\})$]. *We define the entailment $Cn_{\models}$ as follows: $Cn_{\models}(\mathcal{T}) = \{\varphi \in \mathbf{L} /$ for any $s \in S(\mathcal{T}), s \models \varphi\}$.*

We define also, for each $s \in S$: $Cn_{\models}(s) = \{\varphi \in \mathbf{L} / s \models \varphi\}$.

We get (straightforward [4]): $Cn_{\models}$ is a *Tarski entailment*, i.e. it is an extensive entailment satisfying idempotence (point 1) and *monotony* (point 2): for any sets $\mathcal{T}, \mathcal{T}'$ of formulas,

1. $\mathcal{T} \subseteq Cn_{\models}(\mathcal{T}) = Cn_{\models}(Cn_{\models}(\mathcal{T}))$.
2. If $\mathcal{T} \subseteq \mathcal{T}'$ then $Cn_{\models}(\mathcal{T}) \subseteq Cn_{\models}(\mathcal{T}')$.

Notice that we get (immediate from the definitions): $Cn_{\models}(\mathcal{T}) = \bigcap_{s \in S(\mathcal{T})} Cn_{\models}(s)$.
As a particular case, we get $Cn_{\models}(\mathcal{T}) = \mathbf{L}$ if $S(\mathcal{T}) = \emptyset$.

As in classical logic, we use the same notation $Cn_{\models}$ for two different, but closely related, notions (cf Th in Section 2): an entailment, defined by $Cn_{\models}(\mathcal{T})$, and the notion of "theory of a state", defined by $Cn_{\models}(s)$. A justification for using the same writing $Cn_{\models}$ is that $Cn_{\models}(s)$ is indeed a *theory in the meaning of* $Cn_{\models}$: for any $s \in S$, $Cn_{\models}(Cn_{\models}(s)) = Cn_{\models}(s)$.

Proof: $Cn_{\models}(s) \subseteq Cn_{\models}(Cn_{\models}(s))$ since $Cn_{\models}$ is a Tarski entailment.
$Cn_{\models}(Cn_{\models}(s)) = \bigcap_{s' \in S(Cn_{\models}(s))} Cn_{\models}(s')$ by definition of $Cn_{\models}(\mathcal{T})$, and $s \in S(Cn_{\models}(s))$ [i.e. $s \models Cn_{\models}(s)$], thus $Cn_{\models}(Cn_{\models}(s)) \subseteq Cn_{\models}(s)$.

Here is a last result of this kind [4]: For any $\mathcal{T} \subseteq \mathbf{L}$, we get $Cn_{\models}(\mathcal{T}) \subseteq C_{MAK}(\mathcal{T})$.

Notice that, contrarily to $Cn_{\models}$, C_{MAK} is not a Tarski entailment: it falsifies monotony.

4 RELATING MAK AND KLM ENTAILMENTS

4.1 THE MAK ENTAILMENTS WHICH ARE KLM ENTAILMENTS

MAK notion encompasses KLM notion, as noticed in [8, 4]:

Property 4.1 *For each KLM model $\mathbf{S} = (S, l, \prec)$ (defining a KLM entailment C_{KLM}), there exists a MAK model $(S, \models, \prec)$ such that its associated MAK entailment C_{MAK} is equal to C_{KLM}.*

Notice that we even do not need to make l to vanish in the KLM model (even if we know that we could do so without loss of generality). The set of states and the preference relation are unmodified. It suffices to define $\models\!\!\equiv$, as in [4], by: $s \models\!\!\equiv \varphi$ whenever $l(s) \models \varphi$. It is clear from the definitions that we get indeed $\mid\!\sim_{\prec}^{MAK} = \mid\!\sim_{\prec}^{KLM}$.

What is interesting then is to describe precisely the subclass of MAK models which can be translated into KLM models. Here is a characterization of this subclass:

Theorem 4.2 *A MAK model* $\mathbf{S} = (S, \models\!\!\equiv, \prec)$ *gives rise to a MAK entailment* C_{MAK} *which is equal to some KLM entailment* C_{KLM} *iff the MAK entailment* C_{MAK} *is a pre-circumscription.*

Proof: The condition is necessary from Property 3.6. MAK entailments satisfy (CT) from Property 3.10, thus Property 3.7 gives the result.

This shows that what lacks to MAK entailment in order to be a KLM entailment is exactly the full preservation of logical equivalence.

Even if this result is satisfactory from a formal perspective, it is more a characterization of the subclass of the MAK entailments which can be turned into KLM entailments than a characterization of the subclass of MAK models which can be turned into KLM models. In some way, this makes no difference since the preferential entailments are fully defined from the models, but we could expect an easier and more direct property, which can be checked directly on the MAK model, without needing to compute its related MAK entailment.

4.2 A SUBCLASS OF MAK MODELS WHICH ARE KLM MODELS

The fact that, for any KLM entailment, we have $Th(\mathcal{T}) \subseteq C_{KLM}(\mathcal{T})$ for any $\mathcal{T} \subseteq \mathbf{L}$ has already been taken into account in [4] for describing a subclass of the MAK models which (1) can be turned into KLM models and (2) is powerful enough to give rise to all the MAK entailments which are also KLM entailments. However, the description of the subclass given in [4] is needlessly complex. We describe here a simpler and more general subclass, which in our opinion describes all the interesting and non pathological MAK models which satisfy conditions (1) and (2) above.

Definitions 4.3 *1. A MAK model* $\mathbf{S} = (S, \models\!\!\equiv, \prec)$ *is* supra classical *whenever we get* $Th(Cn_{\models\!\!\equiv}(s)) = Cn_{\models\!\!\equiv}(s)$ *for any state* s *in* S. *This means that the "world" associated to each state is classically deductively closed (i.e. is a classical theory).*

2. *An entailment* C *is* supra classical *if it satisfies* $Th(\mathcal{T}) \subseteq C(\mathcal{T})$ *for any* $\mathcal{T} \subseteq \mathbf{L}$.

The first definition concerns only the couple $(S, \models\!\!\equiv)$ describing the semantics to which $\prec$ will be applied, but is independent of the preference relation $\prec$. Voorbraak [13] calls "**L**-faithful" what we call supra classical models.

For the second definition, notice that any pre-circumscription is a supra classical entailment.

Lemma 4.4 *If a MAK model* $\mathbf{S} = (S, \models\!\!\equiv, \prec)$ *is such that the entailment* $Cn_{\models\!\!\equiv}$ *is supra classical, then the MAK model* **S** *is supra classical.*

Proof: (1) $Cn_{\models\!\!\equiv}(s) \subseteq Th(Cn_{\models\!\!\equiv}(s))$ since $\mathcal{T} \subseteq Th(\mathcal{T})$.

(2) $Th(Cn_{\models\!\!\equiv}(s)) \subseteq Cn_{\models\!\!\equiv}(Cn_{\models\!\!\equiv}(s))$ by hypothesis, and we already know that we have in any case $Cn_{\models\!\!\equiv}(Cn_{\models\!\!\equiv}(s)) = Cn_{\models\!\!\equiv}(s)$, which establishes $Th(Cn_{\models\!\!\equiv}(s)) = Cn_{\models\!\!\equiv}(s)$.

Lemma 4.5 *If a MAK model* $\mathbf{S} = (S, \models\!\!\equiv, \prec)$ *is supra classical, then the (Tarski) entailment* $Cn_{\models\!\!\equiv}$ *and the (preferential) MAK entailment* C_{MAK} *defined by* **S** *are pre-circumscriptions.*

Proof: (1) Let $\varphi \in Th(\mathcal{T})$ and $s \in S(\mathcal{T})$, i.e. $\mathcal{T} \subseteq Cn_{\models\!\!\equiv}(s)$, then $\varphi \in Cn_{\models\!\!\equiv}(s)$ by supra classicality of **S**. Thus, $\varphi \in \bigcap_{s \in S(\mathcal{T})} Cn_{\models\!\!\equiv}(s)$, i.e. $\varphi \in Cn_{\models\!\!\equiv}(\mathcal{T})$: $Cn_{\models\!\!\equiv}$ is supra classical. From $Cn_{\models\!\!\equiv}(\mathcal{T}) = \bigcap_{s \in S(\mathcal{T})} Cn_{\models\!\!\equiv}(s)$ we get also $Cn_{\models\!\!\equiv}(\mathcal{T}) \in \mathbf{T}$ since each $Cn_{\models\!\!\equiv}(s)$ is in **T**. Since by Definition 3.9 we get $C_{MAK}(\mathcal{T}) = \bigcap_{s \in S_{\prec}(\mathcal{T})} Cn_{\models\!\!\equiv}(s)$ (where the subset $S_{\prec}(\mathcal{T})$ of $S(\mathcal{T})$ is defined exactly as in Definition 3.1 for KLM models), we get a fortiori $\varphi \in C_{MAK}(\mathcal{T}) \in \mathbf{T}$.

(2) If $\mathcal{T}_1$ and $\mathcal{T}_2$ are two equivalent sets, then, by supra classicality of **S**, for each $s \in S$, we have $s \models \mathcal{T}_1$ iff $s \models \mathcal{T}_2$, i.e. we have $S(\mathcal{T}_1) = S(\mathcal{T}_2)$. A fortiori we get then $S_\prec(\mathcal{T}_1) = S_\prec(\mathcal{T}_2)$. Thus we get $Cn_\models(\mathcal{T}_1) = Cn_\models(\mathcal{T}_2)$ and $C_{MAK}(\mathcal{T}_1) = C_{MAK}(\mathcal{T}_2)$ by the definitions of $Cn_\models(\mathcal{T})$ and $C_{MAK}(\mathcal{T})$ respectively.

We have established:

Property 4.6 *1. A MAK model* $\mathbf{S}=(S, \models, \prec)$ *is supra classical iff the (Tarski) entailment* $Cn_\models$ *that it defines is supra classical, iff the entailment* $Cn_\models$ *is a pre-circumscription.*

2. If a MAK model is supra classical, then the (preferential) MAK entailment C_{MAK} *that it determines is a pre-circumscription.*

We are now in position to establish our second main result:

Theorem 4.7 *1. If a MAK model* **S** *is supra classical, then the MAK entailment* C_{MAK} *that it determines is equal to some KLM entailment* C_{KLM}.

Precisely, if $\mathbf{S} = (S, \models, \prec)$ *is supra classical, then there exists a KLM model* $\mathbf{S}' = (S, l, \prec)$ *with* S *and* $\prec$ *unmodified, such that the MAK entailment defined by* **S** *is the KLM entailment defined by* $\mathbf{S}'$.

2. Any KLM preferential entailment C_{KLM} *is equal to a MAK entailment defined by a supra classical MAK model.*

Precisely, if $\mathbf{S} = (S, l, \prec)$, *then there exists a supra classical MAK model* $\mathbf{S}' = (S, \models, \prec)$ *with* S *and* $\prec$ *unmodified, such that the KLM entailment defined by* **S** *is the MAK entailment defined by* $\mathbf{S}'$.

Proof: (1) Theorem 4.2 and Property 4.6-2 give the first sentence.

Let $\mathbf{S} = (S, \models, \prec)$ be a supra classical MAK model. We get a KLM model as follows: we keep the set S and the relation $\prec$ unmodified. We define l as the mapping $S \to \mathbf{T}$ by taking $l(s) = Cn_\models(s)$. It is immediate to see that the KLM entailment C_{KLM} is equal to C_{MAK}.

(2) We have a constructive proof already: It suffices to see the construction given in Property 4.1: it is clear from the definitions that the MAK model obtained there is supra classical since we have already noticed that each $l(s)$ in Definition 3.1 can be equated to a classical theory. Notice that this theorem could also have been obtained as a consequence of some results in an earlier independent work by Voorbraak [13]. Rather strangely, Voorbraak does not enounce this result in all generality, referring to [4] for further results on the subject.

Thus, we get characterization results and constructive passages simpler and easier than those given in [4]. However, our results are slightly more general (see why in note 4): the subclass of the MAK models considered here is slightly greater than the subclass considered by Dix and Makinson since they consider a strict subclass of the MAK models which can be "amplified" (in their terms). It is easy to see, from [4] together with our results, that the class of the MAK models which can be "amplified" coincide with the class of the supra classical MAK models. Moreover, our comparison does not need to consider a third intermediate (between $Cn_\models$ and C_{MAK}) non classical entailment[3], which plays an important role in the results of [4], but which complicates the direct comparison between KLM and MAK preferential entailments. This simplification comes mainly from our results about supra classical MAK models. And the condition that each $Cn_\models(s)$ must be a theory is easily checked, without the need to compute the associated MAK entailment or to introduce a third non classical entailment.

One consequence of our results about MAK entailments is that, if we are concerned only by those MAK entailments which respect full logical equivalence, then we can restrict our attention to a yet narrower class of MAK models. Indeed, we have seen just above why in

[3] For readers familiar with [4], let us notice that an immediate consequence of our results is that, even in the exact framework and formulation considered in [4, main theorem], the condition (3a) given there ($s \models \varphi$ and $s \models \varphi'$ implies $s \models \varphi \wedge \varphi'$) is redundant.

this case we can restrict our attention to the easily described class of supra classical MAK models. Now, since we know that, for KLM entailments, we can consider only the simplified version of KLM models, our passages between MAK models and KLM models show that we can also require a *unicity of states* condition for MAK models. By "unicity of states", we mean that, for any different $s, s' \in S$, the "worlds" $Cn_{\models}(s)$ and $Cn_{\models}(s')$ corresponding to these two states are different. The class of the supra classical MAK models satisfying unicity of states is powerful enough to generate all the MAK entailments which are pre-circumscriptions. Let us describe briefly now the analogous of the singular and the strictly singular KLM models in terms of MAK models.

Definition 4.8 *A MAK model* $\mathbf{S} = (S, \models, \prec)$ *is* classical *if the "worlds"* $Cn_{\models}(s)$ *are (classical) complete theories, for any state* s *in* S.

Remark 4.9 • *A MAK model is supra classical iff the satisfaction relation* $\models$ *respects the binary connector* $\wedge$*: for each* $s \in S$*, we have*

$$s \models \varphi_1 \wedge \varphi_2 \quad \textit{iff} \quad s \models \varphi_1 \textit{ and } s \models \varphi_2. \quad (R_\wedge)$$

•• *A MAK model is classical iff it is supra classical and* $\models$ *respects the negation* $\neg$*:*

$$s \not\models \varphi \quad \textit{iff} \quad s \models \neg\varphi. \quad (R_\neg)$$

Proof: • If each $Cn_{\models}(s)$ is in $\mathbf{T}$, then, since $\{\varphi_1, \varphi_2\} \equiv \{\varphi_1 \wedge \varphi_2\}$, we get $\{\varphi_1, \varphi_2\} \subseteq Cn_{\models}(s)$ iff $\{\varphi_1 \wedge \varphi_2\} \subseteq Cn_{\models}(s)$. Conversely, let us suppose $(R_\wedge)$. Then, if $\{\varphi_i\}_{i\in I} \subseteq Cn_{\models}(s)$ and $\{\varphi_i\}_{i\in I} \models \varphi$, by compactness of $\models$ there exists a finite $J \subseteq I$ such that $\{\varphi_i\}_{i\in J} \models \varphi$, i.e. $\bigwedge_{i\in J} \varphi_i \models \varphi$, i.e. $\bigwedge_{i\in J} \varphi_i \equiv (\bigwedge_{i\in J} \varphi_i) \wedge \varphi$, thus, by $(R_\wedge)$, $\varphi \in Cn_{\models}(s)$. Remind that we identify a formula with its equivalence class. Makinson does not always make this assumption in [8], thus, his original formalism is slightly more general than the version given in the present text. However, since with KLM entailments a formula can always be replaced by an equivalent formula, we have to make this assumption (or any equivalent one) when we want to compare the two formalisms. This means that if this assumption is not made till the beginning (as in this text), then it must be added, e.g. by requiring in Definition 4.3-1 that $\models$ is standard. Notice that Definition 4.3-1 as it stands implies that two formulas equivalent (for $\models$) are always in the same sets $Cn_{\models}(s)$, thus are "equivalent for $\models$".

•• A theory $Cn_{\models}(s)$ is complete iff $[\varphi \in Cn_{\models}(s)$ iff $\neg\varphi \notin Cn_{\models}(s)]$, i.e. iff $\models$ satisfies $(R_\neg)$.

A MAK model is classical iff $\models$ respects all the logical connectors. For instance, it is immediate to see that $(R_\wedge)$ and $(R_\neg)$ imply $(R_\vee)$:

$$s \models \varphi_1 \vee \varphi_2 \quad \text{iff} \quad s \models \varphi_1 \text{ or } s \models \varphi_2. \quad (R_\vee)$$

Classical MAK models correspond to singular KLM models while the classical MAK models which respect unicity of states correspond to the strictly singular KLM models.

4.3 COMING BACK TO THE ORIGINAL KLM ENTAILMENTS

This work applies also to cases where special conditions are required for the models. We think that the simplicity and the naturalness of our translation is a first serious indication for this. Let us consider the original definitions.

Definitions 4.10 *1. A* consistent KLM model, *is such that each state is* consistent, *meaning that* $l(s)$ *is consistent.*

2. *A KLM model* $\mathbf{S} = (S, l, \prec)$ *is* smooth (stoppered *in [8]) if, for each* $\mathcal{T} \subseteq \mathbf{L}$ *and* $s \in S(\mathcal{T}) - S_\prec(\mathcal{T})$*, there exists* $s' \in S_\prec(\mathcal{T})$ *such that* $s' \prec s$ *("minoration by a minimal state").*

[7] considers only the KLM models which are consistent and smooth. The authors consider that the "converse of (CT)" [if $\mathcal{T}' \subseteq C(\mathcal{T})$, then $C(\mathcal{T}) \subseteq C(\mathcal{T} \cup \mathcal{T}')$], called *cumulative monotony* (CM), is as important as (CT), and they only care of *cumulative entailments*, which satisfy (CT) and (CM). They give the following characterization:

Original KLM characterization [7]: A pre-circumscription C is cumulative iff it is a KLM

entailment defined by a smooth and consistent KLM model.

It happens that this characterization result also holds without the consistency condition, which confirms our opinion made after Definition 3.1 that, for KLM entailments, the requirement that $l(s)$ must be consistent is needless[4].

We get, with the KLM models as defined here, a first modification of KLM characterization:

A "KLM Characterization" allowing inconsistent states: A pre-circumscription C is cumulative iff it is a KLM entailment defined by a smooth KLM model.

The proof is an easy modification of the proof of the original characterization [7], moreover this result has already appeared as [8, Observation 3.4.5] and [13, Proposition 5.4].

We can go even further, by requiring that the KLM model is a *simplified KLM model*, meaning an "unlabelled model", or a "model without states": $S = \mathbf{T}$ and $l = identity$.

A "KLM Characterization" with simplified models: A pre circumscription C is cumulative iff it is a KLM entailment defined by a smooth simplified KLM model.

Indeed, the "if" side comes from the "if" side of the previous characterization, allowing inconsistent states. For the "only if" side, it suffices to define the simplified KLM model associated to C as follows:

(1) $S = \mathbf{T}, l = identity$ (the model is simplified, no need for states).

(2) For $\mathcal{T}_1, \mathcal{T}_2$ in $\mathbf{T}$, $\mathcal{T}_1 \prec_C \mathcal{T}_2$ iff

(2a) $\mathcal{T}_1 = \mathbf{L}$ and $\mathcal{T}_2 \neq f(\mathcal{T})$ for any $\mathcal{T} \in \mathbf{T}$, or

(2b) $\mathcal{T}_2 \neq \mathbf{L}$, $\mathcal{T}_2 \neq \mathcal{T}_1$ and there exists $\mathcal{T}_3, \mathcal{T}_4$ in $\mathbf{T}$ such that $f(\mathcal{T}_3) = \mathcal{T}_1$, $f(\mathcal{T}_4) = \mathcal{T}_2$, and $\mathcal{T}_3 \subseteq \mathcal{T}_2$.

Then, if C is a cumulative pre-circumscription, an easy translation of the proof of the characterization result from [7] (where l is injective, as also taken into account in a "suppression of states" result given in [1]) shows that we get, in a way very similar to the original proof of [7]:

(1) C is equal to the KLM entailment defined by this simplified KLM model, and

(2) this [simplified] KLM model is smooth (and irreflexive).

Notice that this result has also appeared as [13, Proposition 5.5], with an apparently different proof.

We can then get immediately the corresponding characterization results in terms of MAK entailment, by using Theorems 4.2 and 4.7.

5 CONCLUSION AND PERSPECTIVES

We have shown that the notions of preferential entailment as defined by Kraus, Lehmann and Magidor and as defined by Makinson are much closely related than was supposed before. Indeed, these two notions coincide exactly in all the cases where they can coincide, that is when the underlying logic respects classical equivalence. Moreover, we have shown that a similar result holds also for the respective models defining the two notions. It was already known that any KLM model could easily be turned into a MAK model. We have exhibited a natural subclass of the MAK models which can, exactly as easily, be turned into a KLM model. The subclass obtained here is slightly greater, and is much easier to describe, than what was previously known. And this subclass of models is "complete": it generates all the KLM preferential entailments. This subclass is the class of the MAK models for which all the states have a "classical" behavior: the set of formulas they

[4] For readers familiar with [4], let us remind that Dix and Makinson add to (3a) (see note 3) a "consistency of states" condition (3b), for the MAK models, in their main theorem. This condition is necessary in their text only because they disallow inconsistent states in KLM models, following [7]. Since inconsistent states are not a real problem, this condition, which restricts slightly the class of the MAK models concerned, can be suppressed without modifying the results about the preferential entailments.

satisfy is closed for classical deduction. This subclass is the most natural class to consider. Indeed, this is the class such that, for any preference relation $\prec$, we are certain from the beginning that the MAK preferential entailment generated has a classical behavior with respect to logical equivalence. There exist some MAK models outside this class which give rise to a KLM entailment, but these models are rather special, since it turns out that their preference relation, in some way, eliminates all the states in the model with an unclassical behavior. We have also shown that our results apply to important particular subclasses of KLM models and MAK models, namely those which are simplified in that either the labelling mapping l is needless [KLM side], or some "unicity of state" condition is required [MAK side]. And we have shown that, even for the cumulative entailments considered in the original texts, these simplified models suffice, and that the passages between KLM models and MAK models work in this case also. As (non trivial) future work, let us remark that these results should help further study on the subject, since they show that this kind of preferential entailment is not as "cumbersome" as it is qualified even in the founding paper [7]. Even automatic computation could take advantage from these results, since the models considered here have nice properties, which, hopefully, could help designing new kinds of "preferential entailments demonstrators".

Acknowledgements

The author wants to thank the referees for their very useful comments.

References

[1] Alexander Bochman. An Epistemic Representation of Nonmonotonic Inference. Unpublished manuscript, 1999.

[2] Alexander Bochman. Credulous Nonmonotonic Inference. In Th. Dean, ed., *IJCAI-99*, pp. 30–35, 1999.

[3] Tom Costello. The expressive power of circumscription. *Artificial Intelligence*, 104(1–2):313–329, September 1998.

[4] Jürgen Dix and David Makinson. The Relation between KLM and MAK Models for Nonmonotonic Inference Operations. *Journal of Logic, Language, and Information*, 1:131–140, 1992.

[5] Joeri Engelfriet. Non-cumulative reasoning: rules and models. *Journal of Logic and Computation*, 10(5):705–719, 2000.

[6] Michael Freund and Daniel Lehmann. Nonmonotonic inference operations. *Bulletin of the IGPL* , 1(1):23–68, 1993.

[7] Sarit Kraus, Daniel Lehmann, and Menachem Magidor. Nonmonotonic Reasoning, Preferential Models and Cumulative Logics. *Artificial Intelligence*, 44(1–2):167–207, July 1990.

[8] David Makinson. General patterns in nonmonotonic reasoning. In Dov M. Gabbay, C.J. Hogger, and J.A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming, Volume 3: Non-Monotonic and Uncertainty Reasoning*, pages 35–110. OUP, 1994.

[9] Yves Moinard. Characterizing general preferential entailments. In Werner Horn, editor, *14th ECAI*, pages 474–478, Berlin, August 2000. IOS Press.

[10] Yves Moinard. General preferential entailments as circumscriptions. In S. Benferhat and Ph. Besnard, eds., *ECSQARU, LNAI 2143*, pp. 532–543, 2001. Springer-Verlag.

[11] Yves Moinard and Raymond Rolland. Characterization of preferential entailments. *Bulletin of the IGPL,* to appear.

[12] Yoav Shoham. *Reasoning about change*. MIT Press, Cambridge, 1988.

[13] Frans Voorbraak. Preference-based Semantics for Nonmonotonic Logics. In Ruzena Bajcsy, editor, *IJCAI-93*, pages 584–589, 1993. Morgan Kaufmann.

[14] Frans Voorbraak. Preference-based Semantics for Nonmonotonic Logics. Technical Report 96, Department of Philosophy, Utrecht University, 3548 CS Utrecht, The Nederlands, August 1993.

Some logics of belief and disbelief

Samir Chopra
schopra@cse.unsw.edu.au
Knowledge Systems Group
School of Computer Science and Engineering
University of New South Wales, Australia

Johannes Heidema
heidej@unisa.ac.za
Department of Mathematics
University of South Africa
Unisa 0003, South Africa

Thomas Meyer
tmeyer@cs.up.ac.za
Department of Computer Science
University of Pretoria
Pretoria 0002, South Africa

Abstract

The introduction of explicit notions of rejection, or disbelief, into logics for knowledge representation can be justified in a number of ways. Motivations range from the need for versions of negation weaker than classical negation, to the explicit recording of classic belief contraction operations in the area of belief change, and the additional levels of expressivity obtained from an extended version of belief change which includes disbelief contraction. In this paper we present four logics of disbelief which address some or all of these intuitions. Soundness and completeness results are supplied and the logics are compared with respect to applicability and utility.

1 Introduction

The introduction of explicit notions of *belief rejection* or *disbelief* into logics for knowledge representation can be justified in a number of ways. One such justification is found in the research area of *belief change* [1, 3]. Classic belief change recognises two basic operations on the beliefs of an agent: *revision*, where the aim is to incorporate a new belief into an agent's belief state while still maintaining consistency, and *contraction*, where an agent needs to discard one of its beliefs while, at the same time, trying to retain as many of its remaining beliefs as possible. The argument for introducing explicit disbeliefs into the object language of choice is based on the conviction that belief contraction should be seen as a belief change operation on par with revision, and not just as an intermediate step for performing revision, as it is sometimes presented. The following example, based on an example in [9], but originally found in [4], illustrates the point.[1]

Example 1 *Consider the situation where a totally ignorant agent revises its current (empty) belief base with $p \rightarrow q$, then contracts with q, and then revises with p. In a framework which caters for the explicit representation of beliefs only, the principle of informational economy[2] dictates that there is only one feasible result for this sequence of base change operations: the set $\{p \rightarrow q, p\}$. The argument against the claim that this is the only acceptable outcome is that, because the representational framework does not allow for the explicit recording of the contraction of q, the choice of permissible outcomes is skewed in favour of revision operations. Contrast this with a situation in which disbelieving a sentence ϕ can be represented in the object language as a sentence of the form $\overline{\phi}$. In this case the sets $\{p \rightarrow q, \overline{q}\}$ and $\{\overline{q}, p\}$ will be acceptable outcomes as well.*

In the example above, classical belief contraction is equated with disbelief *revision*. But the extended version of belief change in which it is also possible to perform contraction with *disbeliefs* has no equivalent in classical belief change. The contraction of disbeliefs provides for a particularly interesting class of operations if disbeliefs are thought of as *guards* against the introduction of certain beliefs, in the spirit of default logic [10]. Disbelief *contraction* corresponds to the removal of such a guard, which might then trigger the addition of previously suppressed beliefs into the current belief set of an agent. An adequate modelling of this type of extended (dis)belief change requires some level of interaction between beliefs and disbeliefs, something which has not been dealt with adequately in the literature. It also presupposes a consequence relation which handles disbeliefs appropriately, since it allows for contraction with disbeliefs which are *consequences* of previously asserted disbeliefs.

Another motivation for the introduction of disbeliefs is that it allows us to reason about rejected beliefs and their consequences [12, 5, 6]. In this scenario the argument is that

[1] Observe that this is an example of *base change* [2, 7, 11].

[2] The methodological injunction that the revision process preserve as many of the old beliefs as possible.

in many practical situations it makes more sense to adopt a "negative stance": to reason about how to reject new beliefs on the basis of rejecting old ones. Under these circumstances the focus is on the provision of an appropriate consequence relation with regard to disbeliefs.

Yet another motivation for an explicit representation of disbeliefs is that it can be viewed as a weaker version of classical negation. The value of such an additional version of negation is that it enables us to resolve well-known paradoxes of belief such as the lottery paradox.

Example 2 *Given a finite number of lottery tickets issued in a particular week, I express doubt about the possibility of any individual ticket winning the lottery, although I do believe that exactly one of these tickets will win the lottery.*

The lottery paradox would be resolved by replacing classical negation (I believe the claim that ticket number 113 will not win the lottery) with the much weaker notion of disbelief (I disbelieve the claim that ticket number 113 will win the lottery). This should not be seen as an *ad hoc* response to a particular paradox of belief. To the contrary, such a weaker version of negation is prevalent in many practical situations. Here is another example.

Example 3 *I have two very good friends, Agnetha and Björn, who are the only suspects in the murder of my friends Annifrid and Benny. I refuse to believe that Agnetha was able to commit the murder (represented as the disbelief $\overline{a}$), and similarly for Björn (represented as the disbelief $\overline{b}$). Yet, the evidence compels me to believe that one of them committed the crime (represented as $a \vee b$).*

The challenge is to provide a formal definition of disbelief in such a way that sentences of the form $\overline{\phi}$ can be regarded as a legitimate way of negating ϕ, while at the same time ensuring the consistency of sets like $\{\overline{\phi}, \overline{\psi}, \phi \vee \psi\}$.

In this paper we present four different logics of belief and disbelief. Two of these logics, **WBD** and **GBD** have been used implicitly in work on disbelief, but not in the form of a *logic* of belief and disbelief. The first one, **WBD**, has a very weak notion of consequence with regard to disbeliefs. It has been used for the explicit recording of belief contraction operations in [9] and [4]. This logic provides an extremely weak link between beliefs and disbeliefs. The relation between beliefs and disbeliefs is made only via a notion of consistency, and this is done for the sole purpose of providing an accurate model of classical belief contraction in terms of disbeliefs.

The second logic, **GBD**, was presented in [5, 6], albeit in a different form, and is a formalisation of the idea that it is useful to reason about rejected beliefs and their consequences. The notion of consequence, with regard to disbeliefs, associated with **GBD** is stronger than that of **WBD**. We argue in section 4 that it is *too* strong. On the other hand, the connection between beliefs and disbeliefs in **GBD** is just as weak as in **WBD**. We argue that it needs to be strengthened.

The third logic, **BD**, is designed with the express intention of obtaining a weaker version of classical negation. Its version of consequence with respect to disbeliefs seems to be pitched at just the right level. It is weaker than the version provided by **GBD**, but stronger than that of **WBD**. And unlike the cases for **WBD** and **GBD** there is a strong and intuitively plausible connection between beliefs and disbeliefs. Indeed, one of the consequences of the belief in a sentence $\neg\phi$ is that ϕ will be disbelieved (that is, $\overline{\phi}$ will hold), as befits a view of disbelief as a weaker version of classical negation. It is our contention that **BD** is the most useful of the logics introduced in this paper.

With the introduction of **BN**, the fourth and final logic to be discussed, the aim is to take matters a step further by obtaining new beliefs from existing disbeliefs. In the process of doing so, however, disbelief collapses into classical negation. That is, disbelieving $\overline{\phi}$ becomes equivalent to believing $\neg\phi$, and **BN** becomes equivalent to classical logic in its level of expressivity.

In section 2 we present the language on which all of our logics are based. In each of sections 3 to 6 we present and discuss a logic of belief and disbelief. Each logic is equipped with a proof-theoretic and a semantic version of consequence. It is shown that these logics are all sound and complete. In section 7 we draw conclusions and point to future research.

1.1 Formal preliminaries

We assume a classical propositional logic **PL** with a language $L_{\mathbf{PL}}$ generated from a (possibly countably infinite) set of propositional atoms, together with the the usual propositional connectives, and with $\bot$ and $\top$ as canonical representatives of the set of contradictions and tautologies, respectively. V is the set of classical valuations of **PL**. For $\phi \in L_{\mathbf{PL}}$, $M(\phi)$ is the set of models of ϕ. Classical deduction for **PL** is denoted by $\vdash_{\mathbf{PL}}$. For any set X we denote the powerset of X (the set of all subsets of X) by $\mathcal{P}X$. Given a language L and a consequence relation $\vdash_X$ from $\mathcal{P}L$ to L we let $C_X(\Gamma) = \{\alpha \mid \Gamma \vdash_X \alpha\}$ be the consequence operation associated with $\vdash_X$. For example, $C_{\mathbf{PL}}$ denotes classical consequence for propositional logic. It is well-known that $C_{\mathbf{PL}}$ satisfies the properties of Inclusion: $X \subseteq C(X)$, Idempotency: $C(X) = C(C(X))$, Monotonicity: $X \subseteq Y$ implies $C(X) \subseteq C(Y)$, and Compactness: if $\phi \in C(X)$ then $\phi \in C(Y)$ for some finite subset Y of X.[3] An opera-

[3] Instead of taking **PL** as our "basic logic" we can work with any logic whose associated consequence relation satisfies these

tion C from $\mathcal{P}L$ to $\mathcal{P}L$ is a *Tarskian consequence operation* iff it satisfies Inclusion, Idempotency and Monotonicity.

2 A language for beliefs and disbeliefs

The language L on which we base all of the logics to be presented is a simple extension of $L_{\mathbf{PL}}$. For each potential belief ϕ, expressed as a sentence of $L_{\mathbf{PL}}$, there is a corresponding potential disbelief $\overline{\phi}$.

Definition 1 *$L = L_B \cup L_D$, where $L_B = L_{\mathbf{PL}}$ is the set of potential beliefs and $L_D = \{\overline{\phi} \mid \phi \in L_P\}$, the set of potential disbeliefs. An* information set *Γ is any subset of L. The* beliefs *in Γ are defined as $\Gamma_B = L_B \cap \Gamma$. The* disbeliefs *in Γ are defined as $\Gamma_D = L_D \cap \Gamma$.*

We use ϕ and ψ to denote potential beliefs and α and β to denote arbitrary sentences of L (potential beliefs and disbeliefs). The language L is fairly restrictive in the sense that $-$ is not viewed as a propositional connective. Thus, we cannot construct sentences of the form $p \vee \overline{q}$, for example. The reason for this is twofold. Firstly, in many (but not all) motivations for the introduction of disbeliefs, such a level of expressivity is simply not necessary. And secondly, this paper should be seen as a first step towards a description of logics of belief and disbelief. Our intention is to treat $-$ as a full-blown "propositional" connective in future research.

3 The logic WBD

The first logic we consider is relatively weak in terms of the consequences of explicit disbeliefs. The motivation for the introduction of **WBD** is primarily for the explicit expression of classical belief contraction. It was used as such in [4] and [9], albeit implicitly. A proof theory for **WBD** is obtained from the following three inference rules.

(B) If $\Gamma_B \vdash_{\mathbf{PL}} \phi$ then $\Gamma \vdash \phi$

(D$\perp$) If $\phi \vdash_{\mathbf{PL}} \perp$ then $\Gamma \vdash \overline{\phi}$

(WD) If $\overline{\psi} \in \Gamma_D$ and $\phi \vdash_{\mathbf{PL}} \psi$ then $\Gamma \vdash \overline{\phi}$

(B) is a supraclassicality requirement. (D$\perp$) requires that contradictions always be disbelieved, analogous to the case of tautologies always being believed. (WD) requires that disbelieving a sentence $\overline{\psi}$ leads to a disbelief in all those sentences classically stronger than ψ.

Definition 2 *In* **WBD**, *α can be deduced from an information set Γ, written as $\Gamma \vdash_{\mathbf{WBD}} \alpha$, iff (Γ, α) is in the smallest binary relation from $\mathcal{P}L$ to L closed under (B), (D$\perp$), and (WD).*

four properties.

The logic **WBD** has a well-behaved Tarskian consequence operation.

Proposition 1 *$C_{\mathbf{WBD}}$ satisfies Inclusion, Idempotency, Monotonicity and Compactness.*

Observe that beliefs and disbeliefs are completely decoupled in **WBD**. In particular, **WBD** satisfies the following two properties which show that disbeliefs and beliefs are only derivable from the set of disbeliefs and beliefs respectively:

(B$\not\rightarrow$D) $\forall \Omega \subseteq L_B$ and $\phi \in L_B$, $\Gamma \vdash \overline{\phi}$ iff $\Gamma_D \cup \Omega \vdash \overline{\phi}$

(D$\not\rightarrow$B) $\forall \Delta \subseteq L_D$ and $\phi \in L_B$, $\Gamma \vdash \phi$ iff $\Gamma_B \cup \Delta \vdash \phi$

Proposition 2 *$\vdash_{\mathbf{WBD}}$ satisfies (B$\not\rightarrow$D) and (D$\not\rightarrow$B).*

In fact, the only way in which beliefs and disbeliefs are related in **WBD** is by way of the notion of **WBD**-*inconsistency*.

Definition 3 *Γ is* WB-inconsistent *iff $\Gamma_B \vdash_{\mathbf{WBD}} \perp$. Γ is* WD-inconsistent *iff $\Gamma_D \vdash_{\mathbf{WBD}} \overline{\top}$. Γ is* **WBD**-inconsistent *iff $\Gamma \vdash_{\mathbf{WBD}} \phi$ and $\Gamma \vdash_{\mathbf{WBD}} \overline{\phi}$ for some $\phi \in L_B$.*

We have similar versions of inconsistency for the other logics too.

We now turn to the semantics for **WBD**. A useful intuition is to think of the beliefs in Γ as the information an agent has acquired on its own, and each disbelief $\overline{\phi}$ as information it has obtained from a different source, informing it that ϕ does *not* hold. The agent has more faith in its own capabilities than in those of its sources, and information obtained from its sources is therefore seen as less reliable. The information obtained from a specific source is independent of the beliefs of the agent and the information obtained from other sources. A model for **WBD** consists of a set of valuations, corresponding to the worlds that the agent regards as possible, together with a set of sets of valuations, with each element of this set corresponding to the worlds that a particular source of the agent regards as *possible*. A potential belief is satisfied in a model if it is true in all the worlds that the agent regards as possible, and a potential disbelief $\overline{\phi}$ is satisfied in a model if at least one of the sources regards ϕ as impossible. Satisfaction is denoted by $\Vdash$.

Definition 4 *A* **WBD**-model *$\mathcal{M}$ is an ordered pair $(M, \mathcal{N})$ where $M \subseteq V$ and $\emptyset \subset \mathcal{N} \subseteq \mathcal{P}V$. For $\phi \in L_B$, $\mathcal{M} \Vdash \phi$ iff $M \subseteq M(\phi)$. For $\overline{\phi} \in L_D$, $\mathcal{M} \Vdash \overline{\phi}$ iff $\exists N \in \mathcal{N}$ such that $N \subseteq M(\neg\phi)$.*

In a **WBD**-model $\mathcal{M} = (M, \mathcal{N})$, M represents the models of the beliefs of the agent, and each $N \in \mathcal{N}$ represents the models of a sentence that the source associated with N

holds to be possible. We require an agent to have at least one source of information. That is, we require that $\mathcal{N} \neq \emptyset$. Entailment for **WBD** (denoted by $\models_{\mathbf{WBD}}$) is then defined in the normal model-theoretic fashion.

Definition 5 *$\mathcal{M} \Vdash \Gamma$ iff $\mathcal{M} \Vdash \alpha \; \forall \alpha \in \Gamma$. $\Gamma \models_{\mathbf{WBD}} \alpha$ iff $\mathcal{M} \Vdash \alpha$ for every* **WBD***-model $\mathcal{M}$ s.t. $\mathcal{M} \Vdash \Gamma$.*

It turns out that the logic **WBD** is sound and complete.

Theorem 1 *$\Gamma \vdash_{\mathbf{WBD}} \alpha$ iff $\Gamma \models_{\mathbf{WBD}} \alpha$, for all $\alpha \in L$.*

Based on the semantics of **WBD** we can also define appropriate notions of (un)satisfiability which, via theorem 1, can be shown to coincide with inconsistency. We postpone presentation of these details to the full version of this paper.

The logic **WBD** is, essentially, the logic on which the work in [4] and [9] is based. Both these papers argue for an explicit expression of disbeliefs in order to maintain a record of belief contraction. Although **WBD** seems adequate for this purpose, at least in simple settings, two types of criticisms can be levelled at it: the extreme weakness of the notion of consequence associated with disbeliefs, and the complete decoupling of beliefs and disbeliefs. These weaknesses become apparent in scenarios where it is appropriate to perform various belief change operations on beliefs as well as disbeliefs. So, while contracting with a belief might correspond to revising with a disbelief, contracting with a *disbelief* has no equivalent in classical belief change. And such an operation is useful if disbeliefs are thought of as *guards* against the introduction of certain information, in the spirit of default logic [10]. The removal of such a guard might then trigger the addition of beliefs into an agent's current belief set. An adequate modelling of this type of extended (dis)belief change requires some level of interaction between beliefs and disbeliefs. It also presupposes a consequence relation which handles disbeliefs appropriately, since it allows for contraction with disbeliefs which are *consequences* of previously asserted disbeliefs. In section 4 we consider an attempt to rectify the weakness of consequences derived from disbeliefs. In section 5 we address this issue as well, and we also consider the the provision of a link between beliefs and disbeliefs.

4 The logic GBD

One way in which to address the weakness of the consequences to be derived from disbeliefs is to regard consequence, with respect to disbeliefs, as the exact dual of consequence with respect to beliefs. This is the approach followed in [6] where, interestingly enough, the reason for defining such a logic is to define disbelief change.

Let $\overline{\Gamma}_D = \{\neg\phi \mid \overline{\phi} \in \Gamma_D\}$ and consider the following rule:

(GD) If $\overline{\Gamma}_D \vdash_{\mathbf{PL}} \neg\phi$ then $\Gamma \vdash \overline{\phi}$

(GD) requires that the consequences of disbeliefs be the exact dual of classical consequence. The proof theory for the logic **GBD** is then obtained from **WBD** by replacing (WD) with (GD).

Definition 6 *In* **GBD** *α can be deduced from an information set Γ, written as $\Gamma \vdash_{\mathbf{GBD}} \alpha$, iff (Γ, α) is in the smallest binary relation from $\mathcal{P}L$ to L closed under (B), (D$\perp$), and (GD).*

GBD also has a well-behaved Tarskian consequence operation.

Proposition 3 *$C_{\mathbf{GBD}}$ satisfies Inclusion, Idempotency, Monotonicity and Compactness.*

The logic **GBD** satisfies the following property, which was proposed in [5].

(Rej) If $\Gamma \vdash \overline{\beta}$ and $\Gamma \vdash \overline{\neg(\alpha \rightarrow \beta)}$ then $\Gamma \vdash \overline{\alpha}$

(Rej) can be thought of as a version of the inference rule Modus Tollens. In fact, if disbelief is replaced with classical negation, (Rej) coincides exactly with Modus Tollens.

GBD also has a complete decoupling of beliefs and disbeliefs, as the following proposition shows.

Proposition 4 *$\vdash_{\mathbf{GBD}}$ satisfies $(B \not\rightarrow D)$ and $(D \not\rightarrow B)$.*

The three versions of inconsistency for **GBD** are defined as for **WBD**.

Definition 7 *Γ is* GB-inconsistent *iff $\Gamma_B \vdash_{\mathbf{GBD}} \perp$. Γ is* GD-inconsistent *iff $\Gamma_D \vdash_{\mathbf{GBD}} \overline{\top}$. Γ is* **GBD**-inconsistent *iff $\Gamma \vdash_{\mathbf{GBD}} \phi$ and $\Gamma \vdash_{\mathbf{GBD}} \overline{\phi}$ for some $\phi \in L_B$.*

Observe that beliefs and disbeliefs are related only by **GBD**-inconsistency.

A semantics for **GBD** is obtained by considering only those **WBD**-models in which $\mathcal{N}$ has a single element. Intuitively, all disbeliefs are obtained from a single source. This ensures that disbeliefs can be combined to obtain new disbeliefs, which allows for a stronger notion of consequence regarding disbeliefs. **GBD**-entailment are defined as for **WBD**.

Definition 8 *A* **GBD**-model *is an ordered pair $\mathcal{M} = (M, N)$ where $M, N \subseteq V$. For $\phi \in L_B$, $\mathcal{M} \Vdash \phi$ iff $M \subseteq M(\phi)$. For $\overline{\phi} \in L_D$, $\mathcal{M} \Vdash \overline{\phi}$ iff $N \subseteq M(\neg\phi)$. $\mathcal{M} \Vdash \Gamma$ iff $\mathcal{M} \Vdash \alpha \; \forall \alpha \in \Gamma$. $\Gamma \models_{\mathbf{GBD}} \alpha$ iff $\mathcal{M} \Vdash \alpha$ for every* **GBD***-model $\mathcal{M}$ s.t. $\mathcal{M} \Vdash \Gamma$.*

The logic **GBD** is sound and complete.

Theorem 2 $\Gamma \vdash_{\mathbf{GBD}} \alpha$ *iff* $\Gamma \models_{\mathbf{GBD}} \alpha$, *for all* $\alpha \in L$.

We postpone a full description of the appropriate versions of unsatisfiability for **GBD**, which coincide with the three versions of inconsistency for **GBD**, to the full version of this paper.

The crucial difference between **WBD** and **GBD** is that **GBD** combines disbeliefs to obtain new disbeliefs, which allows for a much stronger notion of consequence with respect to disbeliefs. In particular, **GBD** satisfies the following property.

(D∨) If $\Gamma \vdash \overline{\phi}$ and $\Gamma \vdash \overline{\psi}$ then $\Gamma \vdash \overline{\phi \vee \psi}$

It is our contention that such a property is undesirable for a notion of disbelief. Since disbelief is *not* intended to be equivalent to classical negation, it is reasonable to require that it be possible to express notions not expressible in classical logic. One of these is *agnosticism*, in which an agent refuses to commit to a potential belief or its negation. Formally, this amounts to both $\overline{\phi}$ and $\overline{\neg\phi}$ being consequences of a consistent information set Γ. But if the consequence relation satisfies (D∨), it means that the agent is forced to accept $\overline{\phi \vee \neg\phi}$, and therefore $\overline{\top}$ as well. Disbelieving a tautology amounts to GD-inconsistency. It is the analogue of classical inconsistency (believing the negation of the tautology), but with classical negation replaced by disbelief. Now, if agnosticism in the sense described above leads to inconsistency with respect to disbeliefs, it is an indication that the consequence relation for **GBD** is too strong to account for a proper treatment of disbeliefs. In section 5 we consider a logic which seems to be pitched at the right level in this regard.

5 The logic BD

In this section we introduce a logic in which there is interaction between beliefs and disbeliefs. Disbelief is seen as a weaker notion of classical negation. As a result, believing the negation of a sentence also results in disbelieving that sentence, although the converse relationship does not hold. Interestingly enough, this coupling of beliefs and disbeliefs comes about as a result of the introduction of the following inference rule, which looks like a simple strengthening of the consequence relation with respect to disbeliefs.

(D) If $\overline{\psi} \in \Gamma_D$ and $\Gamma_B \cup \{\phi\} \vdash_{\mathbf{PL}} \psi$ then $\Gamma \vdash \overline{\phi}$

(D) requires that disbelieving a sentence ψ also leads to a disbelief in those sentences classically stronger than ψ, *but with respect to* Γ_B. Observe that (WD) is the special case of (D) where $\Gamma_B = \emptyset$. The difference between the rules (GD), (WD) and (D) is perhaps best brought out through an example:

Example 4 *Consider our earlier example of the murder mystery. Then (WD) commits me to the following: If I refuse to believe that Agnetha killed Annifrid and Benny, then I also refuse to believe that Agnetha and Björn killed them (a similar commitment is enforced by (D) and (GD)). However, in addition, (D) commits me to: If I believe that Agnetha is a Swede, I believe that if you killed Annifrid and Benny then you are a murderer, and I refuse to believe that a Swede can be a murderer, then I also refuse to believe that Agnetha could have killed Annifrid and Benny. This argument cannot be made with either (WD) or (GD). To further illustrate the difference, consider what (GD) requires us to infer in the lottery example. Consider for the time being, a version with just two tickets. I believe that exactly one of* t_1 *or* t_2 *will win the lottery; I refuse to believe that* t_1 *wins the lottery and similarly for* t_2. *Then (GD) compels me to refuse to believe that exactly one of the two will win. Therefore I end with a contradiction. In contrast, (D) and (WD) do not require such a commitment.*

From the example above it should be clear that both (D) and (GD) are stronger than (WD). Furthermore, as the second part of the example makes clear, since (D) is stronger than (WD), some conclusions obtained from the former will not be obtainable from the latter, but others will. The example provided is one of those obtainable from (D) but not (WD). Since (D) and (GD) are incomparable in strength, they will have some conclusions that coincide (as above) but there will also be conclusions obtained from (D) which are not obtainable from (GD) and vice-versa. The example provided for (D) is an instance of a conclusion obtainable from (D) but not from (GD).

Definition 9 *In* **BD**, α *can be deduced from an information set* Γ, *written as* $\Gamma \vdash_{\mathbf{BD}} \alpha$, *iff* (Γ, α) *is in the smallest binary relation from* $\mathcal{P}L$ *to* L *closed under (B), (D⊥), and (D).*

BD has a well-behaved Tarskian consequence operation:

Proposition 5 $C_{\mathbf{BD}}$ *satisfies Inclusion, Idempotency, Monotonicity and Compactness.*

The manner in which disbeliefs are obtained from beliefs in **BD** can be expressed by the following property which links up a belief in $\neg\phi$ to a disbelief in ϕ. Observe that neither **WBD** nor **GBD** satisfies $(B \rightarrow D)$.

(B→D) If $\Gamma \vdash \neg\alpha$ then $\Gamma \vdash \overline{\alpha}$

Proposition 6 $\vdash_{\mathbf{BD}}$ *satisfies (B→D) and does not satisfy (B↛D).*

The logic **BD** does not support any connection between beliefs and disbeliefs in the converse direction, though, as the following result shows.

Proposition 7 $\vdash_{\mathbf{BD}}$ *satisfies (D↛B).*

The different notions of inconsistency for **BD** are defined in the same way as for **WBD** and **GBD**.

Definition 10 *An information set* Γ *is* B-inconsistent *iff* $\Gamma_B \vdash_{\mathbf{BD}} \bot$, D-inconsistent *iff* $\Gamma_D \vdash_{\mathbf{BD}} \overline{\top}$, *and* **BD**-inconsistent *iff* $\Gamma \vdash_{\mathbf{BD}} \phi$ *and* $\Gamma \vdash_{\mathbf{BD}} \overline{\phi}$ *for some* $\phi \in L_B$.

Because there is interaction between beliefs and disbeliefs in **BD**, there is also a connection between the different notions of inconsistency for **BD**. In particular, we have the following results.

Proposition 8 *If* Γ *is B-inconsistent then it is also* **BD***-inconsistent, but the converse does not hold.* Γ *is* **BD***-inconsistent iff it is D-inconsistent.*

In the logic **BD**, then, **BD**-inconsistency collapses into D-inconsistency. Given the intuition of disbelief as a weaker version of classical negation, this is a particularly desirable state of affairs. In classical logic, asserting both ϕ and $\neg\phi$ is tantamount to the assertion that $\neg\top$ is the case, while in **BD**, asserting both ϕ and $\overline{\phi}$ amounts to the assertion that $\overline{\top}$ is the case.

Observe that **BD**-inconsistency (or D-inconsistency) amounts to disbelieving the tautology, which leads to a disbelief in every sentence in L_B. So, while B-inconsistency, like classical consistency, leads to the acceptance of *every* sentence in the language, **BD**-inconsistency leads only to the acceptance of every *disbelief* in the language. **BD**-inconsistency can thus be seen as a weaker version of classical inconsistency. We regard this as a particularly attractive feature of the logic **BD**. The logics **WBD** and **GBD** also have similar features, but the connection between classical inconsistency and the weaker version of inconsistency, based on disbeliefs, is not as intuitively appealing.

The semantics for **BD** is obtained by considering only those **WBD**-models for which every $N \in \mathcal{N}$ is a subset of M. That is, the worlds that the sources of an agent may regard as possible have to be worlds that the agent itself regards as possible. **BD**-entailment are defined as for **WBD**-entailment and **GBD**-entailment.

Definition 11 *A* **BD**-model *is a tuple* $(M, \mathcal{N})$ *where* $M \subseteq V$ *and* $\emptyset \subset \mathcal{N} \subseteq \mathcal{P}M$. *For a* **BD***-model* $\mathcal{M} = (M, \mathcal{N})$ *and* $\phi \in L_B$, $\mathcal{M} \Vdash \phi$ *iff* $M \subseteq M(\phi)$. *For* $\overline{\phi} \in L_D$, $\mathcal{M} \Vdash \overline{\phi}$ *iff* $\exists N \in \mathcal{N}$ *s.t.* $N \subseteq M(\neg\phi)$. $\mathcal{M} \Vdash \Gamma$ *iff* $\mathcal{M} \Vdash \alpha\ \forall \alpha \in \Gamma$. $\Gamma \models_{\mathbf{BD}} \alpha$ *iff* $\mathcal{M} \Vdash \alpha$ *for every* **BD***-model* $\mathcal{M}$ *s.t.* $\mathcal{M} \Vdash \Gamma$.

The logic **BD** is sound and complete.

Theorem 3 $\Gamma \vdash_{\mathbf{BD}} \alpha$ *iff* $\Gamma \models_{\mathbf{BD}} \alpha$, *for all* $\alpha \in L$.

Based on the semantics of **BD** we can define appropriate notions of (un)satisfiability which can be shown to coincide with inconsistency. We postpone presentation of these details to the full version of this paper.

We conclude this section by pointing out that **BD** is able to handle examples such as the lottery paradox and its variants (cf. examples 2 and 3) in a manner that is intuitively satisfactory. It can be shown that any information set of the form $\Gamma = \{\overline{\phi_1}, \ldots, \overline{\phi_n}, \bigvee_{i=1}^{i \leq n} \phi_i\}$ is neither B-inconsistent, nor **BD**-inconsistent.[4] **BD** thus allows us, for example, to disbelieve the fact that any particular ticket will win the lottery, while still believing that exactly one of the tickets will win the lottery, without collapsing into *any* kind of inconsistency.

6 The logic BN

We have seen that **BD** allows for the generation of disbeliefs from beliefs. An interesting question is whether it makes sense to do the opposite; that is, to generate beliefs from disbeliefs. We consider two ways of doing so. For the first one, note that the generation of disbeliefs from beliefs in **BD** is achieved by the inference rule (D). Now, it is easily verified that, in the presence of (B) and (D⊥), (D) is equivalent to the following property:

(D′) If $\Gamma \vdash \overline{\psi}$ and $\Gamma \cup \{\phi\} \vdash \psi$ then $\Gamma \vdash \overline{\phi}$

On the basis of this equivalence we consider the following property:

(B′) If $\Gamma \vdash \psi$ and $\Gamma \cup \{\overline{\phi}\} \vdash \overline{\psi}$ then $\Gamma \vdash \phi$

(B′) asserts that if I currently believe ψ, and if the addition of ϕ as a disbelief leads me to disbelieve ψ, then ϕ should be one of my current beliefs. It is analogous to (D′), but with the roles of beliefs and disbeliefs reversed. It turns out, however, that (B′) is a derived rule of the logic **BD**.

Proposition 9 $\vdash_{\mathbf{BD}}$ *satisfies the property (B′).*

A more direct way to obtain new beliefs from current disbeliefs is to consider the converse of the property (B→D).

(D→B) If $\Gamma \vdash \overline{\phi}$ then $\Gamma \vdash \neg\phi$.

Observe that **WBD**, **GBD** and **BD** do not satisfy (D→B). A proof theory for the logic we call **BN** is then obtained by adding (D→B) to the inference rules of **BD**.

Definition 12 *For* $\alpha \in L$ *and* $\Gamma \subseteq L$, α *can be deduced from* Γ *in* **BN**, *written as* $\Gamma \vdash_{\mathbf{BN}} \alpha$, *iff* (Γ, α) *is in the smallest binary relation from* $\mathcal{P}L$ *to* L *closed under (B), (WD), (D⊥) and (D→B).*

[4] By proposition 8 this means that Γ is also not D-inconsistent.

The introduction of (D→B) does indeed give us a logic that is stronger than **BD**. It turns out, however, that disbelief now collapses into classical negation

Theorem 4 *For every* $\phi \in L_B$, $\Gamma \vdash_{\mathbf{BN}} \overline{\phi}$ *iff* $\Gamma \vdash_{\mathbf{BN}} \neg\phi$.

BN is therefore exactly as expressive as **PL**.

7 Conclusion and future research

Of the four logics of belief and disbelief presented in this paper, the logic **BD** is the most deserving of such a label. **BD** covers all the motivations we have mentioned for the explicit introduction of disbeliefs. Contraction with a belief ϕ can be represented explicitly in **BD** as revision with the disbelief $\overline{\phi}$, just as it can in **WBD** and **GBD**. This ability has a useful side-effect as well. In classical belief change, the result of the pathological case in which a belief set is contracted with a tautology, is taken to be the belief set itself, primarily because it is unclear what else the result could be. In fact, it is the only case in which a contraction does not succeed. But in **BD** (as in **WBD** and **GBD**), contraction by ϕ corresponds to a revision by $\overline{\phi}$. So contraction by a tautology is equivalent to revising with the sentence $\overline{\top}$, a disbelief in the tautology. Furthermore, it can be verified that a disbelief in the tautology results in *disbelieving* all (propositional) sentences, but has no effect on the current *beliefs*. The explicit introduction of disbeliefs thus enables us to devise a result which is much more in line with our intuitions.

Like **WBD** and **GBD**, **BD** also provides a well-behaved notion of consequence with respect to disbeliefs. In **BD**, consequence with respect to disbeliefs is stronger than in **WBD** but weaker than in **GBD**. In particular, (D) is satisfied in **BD** but not in **WBD**, and **BD** does not satisfy (D∨), a property satisfied by **GBD**. As a result of not satisfying (D∨), consequence for **BD** is weak enough to allow for the expression of *agnosticism* (disbelieving both ϕ and $\neg\phi$) without collapsing into inconsistency. In this respect it is superior to the logic **GBD**. At the same time, there is a link between beliefs and disbeliefs in **BD** which is lacking in both **WBD** and **GBD**, and which ensures that **BD** is a suitable base logic for an extended version of belief change in which it is possible to contract with disbeliefs as well. The link between beliefs and disbeliefs also ensures that disbelief, as defined in **BD**, is an appropriate weaker version of classical negation, as is apparent from the fact that **BD** satisfies (B→D) and (D↛B), as well as from the intuitively acceptable manner in which the lottery paradox and its variants are handled.

If disbelief is truly to be viewed as a weaker version of negation, it is necessary to conduct a proper investigation into the connection between disbelief, classical negation, and the other propositional connectives. In order to do so, it is necessary to treat $-$ as full-blown propositional connective, in which sentences such as $p \vee \overline{q}$, $\neg\overline{p}$, and $\overline{\overline{p}}$ have a well-defined meaning.

There is an obvious connection between beliefs and disbeliefs in **BD** and the modal logic operators of necessity and possibility, specifically in the epistemic logic **KD45** [8]. The statement $\Box\phi$ in **KD45** corresponds to the assertion that ϕ is believed, while $\Diamond\neg\phi$ corresponds to the assertion that $\neg\phi$ is possible, and hence that ϕ is disbelieved. Observe, though, that statements in **BD** are object-level assertions, or assertions from a first person perspective ("The sky is blue"), while the corresponding statements in epistemic logics are meta-level assertions, or assertions from a third person perspective ("The agent believes the sky is blue"). However, the differences run slightly deeper and are easily illustrated. To compare **BD** with **KD45** we need to restrict **KD45** to sentences of the form $\Box\phi$ and $\Diamond\phi$. Then **KD45** satisfies (B), (D⊥) and (D). So it is at least as strong as **BD**. It does not satisfy ($B \not\vdash D$), and satisfies ($D \not\vdash B$). It does not satisfy (Rej). It supports agnosticism (I disbelieve ϕ as well as $\neg\phi$) but does not satisfy (D∨). In these respects it has the same behaviour as **BD**. But it is stronger than **BD**. This can be seen by looking at what happens when the tautology is disbelieved. In **KD45** (indeed, in any normal modal logic i.e., one containing the axiom schema **K**), if $\Diamond(\neg\top)$ can be deduced from Γ, then so can $\Box\phi$ and $\Diamond\phi$ for every propositional sentence ϕ. That is, disbelieving $\top$ in **KD45** leads to a belief as well as a disbelief in every propositional sentence. In contrast disbelieving the tautology in **BD** leads to a disbelief in every propositional sentence, but our beliefs remain unaffected. This is a particularly desirable property—as pointed out in the discussion following Proposition 8.

In the full version of the paper we plan to develop a restricted version of **KD45** as defined above as a separate logic of belief and disbelief. Developing an axiomatization will be the primary task since the normal **KD45** characterization requires a more expressive language than we possess at the moment. Furthermore, we will provide a formal proof to the effect that the semantics of the logic **BD** is not adequately captured by a class of Kripke models. Applications of the logics that we have developed to belief revision is a non-trivial task that needs separate study.

8 Acknowledgements

The first author would like to thank the University of South Africa in Pretoria for the invitation and financial support to visit its Departments of Mathematics and Computer Science in November 2001, when this paper was conceived. The authors would also like to thank Aditya Ghose for some insightful comments and suggestions.

References

[1] Carlos E. Alchourrón, Peter Gärdenfors, and David Makinson, 'On the logic of theory change: Partial meet functions for contraction and revision', *Journal of Symbolic Logic*, **50**, 510–530, (1985).

[2] André Fuhrmann, 'Theory contraction through base contraction', *Journal of Philosophical Logic*, **20**, 175–203, (1991).

[3] Peter Gärdenfors, *Knowledge in Flux : Modeling the Dynamics of Epistemic States*, The MIT Press, Cambridge, Massachusetts, 1988.

[4] Aditya Ghose and Randy Goebel, 'Belief states as default theories: Studies in non-prioritised belief change', in *ECAI 98. 13th European Conference on Artificial Intelligence*, ed., Henri Prade, pp. 8–12, New York, (1998). John Wiley & Sons, Ltd.

[5] Anna Gomolińska, 'On the logic of acceptance and rejection', *Studia Logica*, **60**, 233–251, (1998).

[6] Anna Gomolińska and David Pearce, 'Disbelief Change', in *Electronic essays on the occasion of the fiftieth birthday of Peter Gärdenfors*, (2001).

[7] Sven-Ove Hansson, 'Changes of disjunctively closed bases', *Journal of Logic, Language and Information*, 2(4), 255–284, (1993).

[8] G.E. Hughes and M.J. Cresswell, *An introduction to Modal Logic*, Methuen, 1972.

[9] Thomas Meyer, Aditya Ghose, and Samir Chopra, 'Non-prioritised Ranked Belief Change', in *Theoretical Aspects of Rationality and Knowledge: Proceedings of the Eighth Conference (TARK 2001)*, ed., Johan van Benthem, pp. 151–162, San Francisco, California, (2001). Morgan Kaufmann.

[10] Raymond Reiter, 'A logic for default reasoning', *Artificial Intelligence*, **13**, 81–132, (1980).

[11] Hans Rott, *Change, Choice and Inference*, Oxford University Press, 1996.

[12] J. Slupecki, G. Bryll, and U Wybraniec-Skardowska, 'Theory of rejected propositions II', *Studia Logica*, **30**, 97–145, (1972).

What is a Joint Goal?
Games with Beliefs and Defeasible Desires

Mehdi Dastani
Institute of Information and Computing Sciences
Utrecht University
email: mehdi@cs.uu.nl

Leendert van der Torre
Department of Artificial Intelligence
Vrije Universiteit Amsterdam
email: torre@cs.vu.nl

Abstract

In this paper we introduce a qualitative decision and game theory based on belief (B) and desire (D) rules. We show that a group of agents acts as if it is maximizing achieved joint goals.

1 Introduction

One of the main problems in agent theory is the distinction between formal theories and tools developed for individual autonomous agents, and theories and tools developed for multi agent systems. In the social sciences, this distinction is called the micro-macro dichotomy. The prototypical example is the distinction between classical decision theory based on the expected utility paradigm (usually identified with the work of Neumann and Morgenstern [15] and Savage [11]) and classical game theory (such as the work of Nash and more recently the work of Axelrod). Whereas classical decision theory is a kind of optimization problem (maximizing the agent's expected utility), classical game theory is a kind of equilibria analysis.

There are several approaches in practical reasoning (within philosophy), cognitive science and artificial intelligence to bring the micro and macro description together. The basic idea is two-fold:

1. The decision making of individual autonomous agents is described in terms of other concepts than maximizing utility. For example, since the early 40s there is a distinction between classical decision theory and artificial intelligence based on utility aspiration levels and goal based planning (as pioneered by Simon [12]). Cognitive theories are typically based on vague concepts from folk psychology like beliefs, desires and intentions.

2. The decision making of a group or society of agents is described in terms of concepts generalized from those used for individual agents, such as joint goals, joint intentions, joint commitments, etc. Moreover, also new concepts are introduced at this social level, such as norms (a central concept in most social theories).

It is still an open problem how the micro-macro dichotomy of classical decision and game theory is related to the micro-macro dichotomy of these alternative theories. Has or can the micro and macro level be brought together by replacing classical theories by alternative theories? Before this question can be answered, the relation between the classical and alternative theories has to be clarified. Doyle and Thomason [6] argue that classical decision theory should be reunited with alternative decision theories in so-called qualitative decision theory (QDT), which studies qualitative versions of classical decision theory, hybrid combinations of quantitative and qualitative approaches to decision making, and decision making in the context of artificial intelligence applications such as planning, learning and collaboration. Qualitative decision theories have been developed based on beliefs (probabilities) and desires (utilities) using formal tools such as modal logic [2] and on utility functions and knowledge [7]. More recently these beliefs-desires models have been extended with intentions or BDI models [5, 10].

In this paper we introduce a rule based qualitative decision and game theory, based on belief (B) and desire (D) rules. We call an individual autonomous agent which minimizes its unreached desires a BD rational agent. We define goals as a set of formulas which can be derived by beliefs and desires in a certain way, such that BD rational agents act as if they maximize the set of achieved goals, and agents maximizing their sets of achieved goals are BD rational. Moreover, groups of

agents which end up in equilibria act as if they maximize joint goals.

Like classical decision theory but in contrast to several proposals in the BDI approach [5, 10], the theory does *not* incorporate decision processes, temporal reasoning, and scheduling.

1.1 QDT and NMR

Qualitative decision theory is related to non-monotonic logic ("qualitative") and to reasoning about uncertainty ("decision theory" formalizes decision making under uncertainty). However, they are based on different disciplines. According to a distinction made by Aristotle, non-monotonic reasoning and reasoning about uncertainty formalize theoretical (or conclusion oriented) reasoning, whereas qualitative decision theory formalizes practical (or action oriented) reasoning.

Thomason [13] observes that classical decision theory neglects the issue of 'correct inference', and that the absence of a logical theory of practical reasoning is largely due to the unavailability of appropriate inference mechanisms. To handle even the simplest cases of practical reasoning, it is essential to have a reasoning mechanism that allows for practical conclusions that are non-monotonic in the agent's beliefs.

Where classical decision theory is based on probabilities and utilities, qualitative decision theory is based on beliefs and desires. In a modal approach, where the beliefs and desires are represented by modalities B and D respectively, and an action operator by the modal operator Do, we may have for example that:

$$B(thirsty), D(drink) \vdash Do(go-to-pub)$$

$$B(thirsty), D(drink), B(pub{-}closed) \vdash Do(go{-}to{-}shop)$$

A drawback of such a modal logic approach is that on the one hand modal logic is notorious for its problems with formalizing conditionals or rules, and practical reasoning on the other hand is usually seen as a kind of rule based reasoning. In this paper we therefore do not use modal logic but we use a rule based formalism.

The layout of this paper is as follows. Section 2 develops a qualitative logic of decision. This logic tells us what the optimal decision is, but it does not tell us how to find this optimal decision. Section 3 considers the AI solution to this problem [12, 8]: break down the decision problem into goal generation and goal based decisions.

2 A qualitative decision and game theory

The qualitative decision and game theory introduced in this section is based on sets of belief and desire rules. We define an agent system specification, we show how we can derive a game specification from it, and we give some familiar notions from game theory such as Pareto efficient decisions (choosing an optimal decision) and Nash equilibria. First we consider the logic of rules we adopt.

2.1 Logic of rules

The starting point of any theory of decision is a distinction between choices made by the decision maker and choices imposed on it by its environment. For example, a software upgrade agent (decision maker) may have the choice to upgrade a computer system at a particular time of the day. The software company (environment) may in turn allow/disallow such a upgrade at a particular time. Let $S = \{\alpha_1, \ldots, \alpha_n\}$ be the society or set of agents, then we therefore assume n disjoint sets of propositional atoms: $A = A_1 \cup \ldots \cup A_n = \{a, b, c, \ldots\}$ (agents' decision variables [7] or controllable propositions [2]) and $W = \{p, q, r, \ldots\}$ (the world parameters or uncontrollable propositions). In the sequel we consider each decision maker as entities consisting of defeasible rules. Such a decision maker generates its decisions by applying subsets of defeasible rules to its input. This results in the so-called conditional mental attitudes [4]. Before we proceed some notations will be introduced.

- L_{A_i}, L_W and L_{AW} for the propositional languages built up from these atoms in the usual way, and variables x, y, ... to stand for any sentences of these languages.
- Cn_{A_i}, Cn_W and Cn_{AW} for the consequence sets, and $\models_{A_i}$, $\models_W$ and $\models_{AW}$ for satisfiability, in any of these propositional logics.
- $x \Rightarrow y$ for an ordered pair of propositional sentences called a rule.
- $E_R(T)$ for the R extension of T, as defined in Definition 1 below.

In our framework the generation of decisions are formalized based on the notion of extension. In particular, the decision of an agent, which is specified by a set of defeasible rules R and has the input T, is the extension calculated based on R and T. This is formalized in the following definition.

Definition 1 (Extension) *Let $R \subseteq L_{AW} \times L_{AW}$ be a set of rules and $T \subseteq L_{AW}$ be a set of sentences. The consequents of the T-applicable rules are:*

$$R(T) = \{y \mid x \Rightarrow y \in R, x \in T\}$$

and the R extension of T is the set of the consequents of the iteratively T-applicable rules:

$$E_R(T) = \cap_{T \subseteq X, R(Cn_{AW}(X)) \subseteq X} X$$

We give some properties of the R extension of T in Definition 1. First note that $E_R(T)$ is *not* closed under logical consequence. The following proposition shows that $E_R(T)$ is the smallest superset of T closed under the rules R interpreted as inference rules.

Proposition 1 *Let*

- $E^0_R(T) = T$
- $E^i_R(T) = E^{i-1}_R(T) \cup R(Cn_{AW}(E^{i-1}_R(T)))$ *for* $i > 0$

We have $E_R(T) = \cup_0^\infty E^i_R(T)$.

Usually, an decision making agent is required to preserve its decisions under the growth of inputs. The following proposition shows that $E_R(T)$ is monotonic.

Proposition 2 *We have* $R(T) \subseteq R(T \cup T')$ *and* $E_R(T) \subseteq E_R(T \cup T')$.

Monotonicity is illustrated by the following example.

Example 1 *Let* $R = \{\top \Rightarrow p, a \Rightarrow \neg p\}$ *and* $T = \{a\}$, *where* $\top$ *stands for any tautology like* $p \vee \neg p$. *We have* $E_R(\emptyset) = \{p\}$ *and* $E_R(T) = \{a, p, \neg p\}$, *i.e. the R extension of T is inconsistent.*

Of course, allowing inconsistent decisions may not be intuitive. We are here concerned about possible decisions rather than reasonable or feasible decisions. Later we will define reasonable or feasible decisions by excluding inconsistent decisions.

2.2 Agent system specification

An agent system specification given in Definition 2 contains a set of agents and for each agent a description of its decision problem. The agent's decision problem is defined in terms of its beliefs and desires, which are considered as defeasible belief and desire rules, a priority ordering on the desire rules, as well as a set of facts and an initial decision (or prior intentions). We assume that agents are autonomous, in the sense that there are no priorities between desires of distinct agents.

Definition 2 (Agent system specification) *An agent system specification is a tuple* $AS = \langle S, F, B, D, \geq, \delta^0 \rangle$ *that contains a set of agents S, and for each agent i a finite set of facts* $F_i \subseteq L_W$ $(F = \bigcup_{i=1}^n F_i)$, *a finite set of belief rules* $B_i \subseteq L_{AW} \times L_W$ $(B = \bigcup_{i=1}^n B_i)$, *a finite set of desire rules* $D_i \subseteq L_{AW} \times L_{AW}$ $(D = \bigcup_{i=1}^n D_i)$, *a relation* $\geq_i \subseteq D_i \times D_i$ $(\geq = \bigcup_{i=1}^n \geq_i)$ *which is a total ordering (i.e. reflexive, transitive, and antisymmetric and for any two elements d_1 and d_2 in D_i, either $d_1 \geq_i d_2$ or $d_2 \geq_i d_1$), and a finite initial decision* $\delta^0_i \subseteq L_A$ $(\delta^0 = \bigcup_{i=1}^n \delta^0_i)$. *For an agent $i \in S$ we write $x \Rightarrow_i y$ for one of its rules.*

A belief rule 'the agent α_i believes y in context x' is an ordered pair $x \Rightarrow_i y$ with $x \in L_{AW}$ and $y \in L_W$, and a desire rule 'the agent desires y in context x' is an ordered pair $x \Rightarrow_i y$ with $x \in L_{AW}$ and $y \in L_{AW}$. It implies that the agent's beliefs are about the world $(x \Rightarrow_i p)$, and not about the agent's decisions. These beliefs can be about the effects of decisions made by the agent $(a \Rightarrow_i p)$ as well as beliefs about the effects of parameters set by the world $(p \Rightarrow_i q)$. Moreover, the agent's desires can be about the world $(x \Rightarrow_i p$, desire-to-be), but also about the agent's decisions $(x \Rightarrow_i a$, desire-to-do). These desires can be triggered by parameters set by the world $(p \Rightarrow_i y)$ as well as by decisions made by the agent $(a \Rightarrow_i y)$. Modelling mental attitudes such as beliefs and desires in terms of defeasible rules results in what might be called conditional mental attitudes [4].

2.3 Agent Decisions

The belief rules are used to determine the expected consequences of a decision, where a decision δ is any subset of L_A that contains the initial decision δ^0. The set of expected consequences of this decision δ is the belief extension of $F \cup \delta$. Moreover, we consider a feasible decision as a decision that does not imply a contradiction.

Definition 3 (Decisions) *Let* $AS = \langle \{\alpha_1, \ldots, \alpha_n\}, F, B, D, \geq, \delta^0 \rangle$ *be an agent system specification. An AS decision profile δ for agents α_1, ..., α_n is $\delta = \langle \delta_1, \ldots \delta_n \rangle$ where δ_i is a decision of agent α_i such that*

$$\delta^0_i \subseteq \delta_i \subseteq L_{A_i} \quad \text{for } i = 1 \ldots n$$

A feasible decision for agent i is δ_i such that

$$E_{B_i}(F_i \cup \delta_i) \text{ is consistent}$$

A feasible decision profile is a decision profile such that

$$E_B(F \cup \delta) \text{ is consistent}$$

where we write $E_B(F \cup \delta)$ *for* $\bigcup_{i=1}^{n} E_{B_i}(F_i \cup \delta_i)$.

The following example illustrates the decisions of a single agent.

Example 2 *Let* $A_1 = \{a, b, c, d, e\}$, $W = \{p, q\}$ *and* $AS = \langle \{\alpha_1\}, F, B, D, \geq, \delta^0 \rangle$ *with* $F_1 = \{\neg p\}$, $B_1 = \{c \Rightarrow q, d \Rightarrow q, e \Rightarrow \neg q\}$, $D_1 = \{\top \Rightarrow a, \top \Rightarrow b, b \Rightarrow p, \top \Rightarrow q, d \Rightarrow q\}$, $\geq_1 = \{b \Rightarrow p > \top \Rightarrow b\}$, *and* $\delta_1^0 = \{a\}$. *The initial decision* δ_1^0 *reflects that the agent has already decided in an earlier stage to reach the desire* $\top \Rightarrow a$. *Note that the consequents of all* B_1 *rules are sentences of* L_W, *whereas the antecedents of the* B_1 *rules as well as the antecedents and consequents of the* D_1 *rules are sentences of* L_{AW}. *We have due to the definition of* $E_R(S)$:
$E_B(F \cup \{a\}) = \{\neg p, a\}$
$E_B(F \cup \{a, b\}) = \{\neg p, a, b\}$
$E_B(F \cup \{a, c\}) = \{\neg p, a, c, q\}$
$E_B(F \cup \{a, d\}) = \{\neg p, a, d, q\}$
$E_B(F \cup \{a, e\}) = \{\neg p, a, e, \neg q\}$
...
$E_B(F \cup \{a, d, e\}) = \{\neg p, a, d, e, q, \neg q\}$
...
Therefore $\{a, d, e\}$ *is not a feasible AS decision profile, because its belief extension is inconsistent. Continued in Example 4.*

The following example illustrates that the set of feasible decisions of an agent may depend on the decisions of other agents.

Example 3 *Let* $A_1 = \{a\}$, $A_2 = \{b\}$, $W = \{p\}$ *and* $AS = \langle \{\alpha_1, \alpha_2\}, F, B, D, \geq, \delta^0 \rangle$ *with* $F_1 = F_2 = \emptyset$, $B_1 = \{a \Rightarrow p\}$, $B_2 = \{b \Rightarrow \neg p\}$, $D_1 = \{\top \Rightarrow p\}$, $D_2 = \{\top \Rightarrow \neg p\}$, $\geq$ *is the identity relation, and* $\delta_1^0 = \delta_2^0 = \emptyset$. *We have that* $\langle \emptyset, \{b\} \rangle$ *is a feasible decision profile, but* $\langle \{a\}, \{b\} \rangle$ *is not. If* $\delta_1 = \emptyset$, *then agent* α_2 *can decide* $\delta_2 = \{b\}$. *However, if* $\delta_1 = \{a\}$, *then agent* α_2 *cannot decide so, i.e.* $\delta_2 \neq \{b\}$.

2.4 Agent preferences

In this section we introduce a way to compare decisions. We compare decisions by comparing sets of desire rules that are not reached by the decisions. Since only ordering on individual desire rules, and not ordering on sets of desire rules, are given, we first lift the ordering on individual desire rules to an ordering on sets of desire rules.

Definition 4 *Let* $AS = \langle S, F, B, D, \geq, \delta^0 \rangle$ *be an agent system specification,* D_i', D_i'' *two subsets of* D_i, *and* $D_i \setminus D_i'$ *the set of* D_i *elements which are not* D_i' *elements. We have* $D_i' \succeq D_i''$ *if* $\forall d'' \in D_i'' \setminus D_i' \exists d' \in D_i' \setminus D_i''$ *such that* $d' > d''$. *We write* $D_i' \succ D_i''$ *if* $D_i' \succeq D_i''$ *and* $D_i'' \not\succeq D_i'$, *and we write* $D_i' \simeq D_i''$ *if* $D_i' \succeq D_i''$ *and* $D_i'' \succeq D_i'$.

The following propositions show that the priority relation $\succeq$ is reflexive, anti-symmetric, and transitive.

Proposition 3 *Let* $D_1 \not\subseteq D_3$ *and* $D_3 \not\subseteq D_1$. *For finite sets, the relation* $\succeq$ *is reflexive (* $\forall D\ D \succeq D$*), anti-symmetric* $\forall D_1, D_2\ (D_1 \succeq D_2 \wedge D_2 \succeq D_1) \rightarrow D_1 = D_2$, *and transitive, i.e.* $D_1 \succeq D_2$ *and* $D_2 \succeq D_3$ *implies* $D_1 \succeq D_3$.

The desire rules are used to compare the decisions. The comparison is based on the set of unreached desires and not on the set of violated or reached desires. A desire $x \Rightarrow y$ is unreached by a decision if the expected consequences of this decision imply x but not y. The desire rule is violated or reached if these consequences imply respectively $x \wedge \neg y$ or $x \wedge y$, respectively.

Definition 5 (Comparing decisions) *Let* $AS = \langle S, F, B, D, \geq, \delta^0 \rangle$ *be an agent system specification and* δ *be a AS decision. The unreached desires of decision* δ *for agent* α_i *are:*

$$U_i(\delta) = \{x \Rightarrow y \in D_i \mid E_B(F \cup \delta) \models x \text{ and } E_B(F \cup \delta) \not\models y\}$$

Decision δ *is at least as good as decision* δ' *for agent* α_i, *written as* $\delta \geq_i^U \delta'$, *iff*

$$U_i(\delta') \succeq U_i(\delta)$$

Decision δ *dominates decision* δ' *for agent* α_i, *written as* $\delta >_i^U \delta'$, *iff*

$$\delta \geq_i^U \delta' \text{ and } \delta' \not\geq_i^U \delta$$

The following continuation of Example 2 illustrates the comparison of decisions.

Example 4 (Continued) *We have:*
$U(\{a\}) = \{\top \Rightarrow b, \top \Rightarrow q\}$,
$U(\{a, b\}) = \{b \Rightarrow p, \top \Rightarrow q\}$,
$U(\{a, c\}) = \{\top \Rightarrow q\}$,
$U(\{a, d\}) = \{\top \Rightarrow b, d \Rightarrow q\}$,
$U(\{a, e\}) = \{\top \Rightarrow b, \top \Rightarrow q\}$,
$U(\{a, b, c\}) = \{b \Rightarrow p\}$.
...
We thus have for example that the decision $\{a, c\}$ *dominates the initial decision* $\{a\}$, *i.e.* $\{a, c\} >^U \{a\}$. *There are two decisions for which their set of unreached contains only one desire. Due to the priority relation, we have that* $\{a, c\} >^U \{a, b, c\}$.

2.5 Agent games

In this subsection, we consider agents interactions based on agent system specifications, their corresponding agent decisions, and the ordering on the decisions as explained in previous subsections. Game theory is the usual tool to model the interaction between self-interested agents. Agents select optimal decisions under the assumption that other agents do likewise. This makes the definition of an optimal decision circular, and game theory therefore restricts its attention to equilibria. For example, a decision is a Nash equilibrium if no agent can reach a better (local) decision by changing its own decision. The most used concepts from game theory are Pareto efficient decisions, dominant decisions and Nash decisions. We first repeat some standard notations from game theory [1, 9].

As mentioned, we use δ_i to denote a decision of agent α_i and $\delta = \langle \delta_1, \ldots, \delta_n \rangle$ to denote a decision profile containing one decision for each agent. δ_{-i} is the decision profile of all agents except the decision of agent α_i. (δ_{-i}, δ_i') denotes a decision profile which is the same as δ except that the decision of agent i from δ is replaced with the decision of agent i from δ'. $\delta_i' >_i^U \delta_i$ denotes that decision δ_i' is better than δ_i according to his preferences $>_i^U$ and $\delta_i' \geq_i^U \delta_i$ if better or equal. Δ is the set of all decision profiles for agents $\alpha_1, \ldots, \alpha_n$, $\Delta_f \subseteq \Delta$ is the set of feasible decision profiles, and Δ^i is the set of possible decisions for agent α_i.

Definition 6 (Game specification) *Let $AS = \langle S = \{\alpha_1, \ldots, \alpha_n\}, F, B, D, \geq, \delta^0 \rangle$ be specification of agent system in S, A_i be the set of AS feasible decisions of agent α_i according to Definition 3, $\Delta_f = A_1 \times \ldots \times A_n$, and $\geq_i^U$ be the AS preference relation of agent α_i defined on its feasible decisions according to definition 5. Then, the game specification of AS is the tuple $\langle S, \Delta_f, (\geq_i^U) \rangle$.*

We now consider different types of decision profiles which are similar to types of strategy profiles from game theory.

Definition 7 *A PS decision profile $\delta = \langle \delta_1, \ldots, \delta_n \rangle \in \Delta_f$ is:*

Pareto decision *if there is no $\delta' = \langle \delta_1', \ldots, \delta_n' \rangle \in \Delta_f$ for which $\delta_i' >_i^U \delta_i$ for all agents α_i.*

strongly Pareto decision *if there is no $\delta' = \langle \delta_1', \ldots, \delta_n' \rangle \in \Delta_f$ for which $\delta_i' \geq_i^U \delta_i$ for all agents α_i and $\delta_j' >^U \delta_j$ for some agents α_j.*

dominant decision *if for all $\delta' \in \Delta_f$ and for every agent i it holds: $(\delta'_{-i}, \delta_i) \geq_i^U (\delta'_{-i}, \delta_i')$ i.e. a decision is dominant if it yields a better payoff than any other decisions regardless of what the other agents decide.*

Nash decision *if for all agents i it holds: $(\delta_{-i}, \delta_i) \geq_i^U (\delta_{-i}, \delta_i')$ for all $\delta_i' \in \Delta_f^i$*

It is a well known fact that Pareto decisions exist (for finite games), whereas dominant decisions do not have to exist. The latter is illustrated by the following example.

Example 5 *Let α_1 and α_2 be two agents, $F_1 = F_2 = \emptyset$, and initial decisions $\delta_1^0 = \delta_2^0 = \emptyset$. They have the following beliefs en desires:*
$B_{\alpha_1} = \{a \Rightarrow p, \neg a \Rightarrow \neg p\}$
$D_{\alpha_1} = \{\top \Rightarrow p, \top \Rightarrow q\}$
$\geq_{\alpha_1} = \top \Rightarrow p > \top \Rightarrow q > \top \Rightarrow \neg q > \top \Rightarrow \neg p$
$B_{\alpha_2} = \{b \Rightarrow q, \neg b \Rightarrow \neg q\}$
$D_{\alpha_2} = \{\top \Rightarrow \neg p, \top \Rightarrow \neg q\}$
$\geq_{\alpha_2} = \top \Rightarrow \neg p > \top \Rightarrow \neg q > \top \Rightarrow q > \top \Rightarrow p$

Let Δ_f be feasible decision profiles, E_B be the outcomes of the decisions, and $U(\delta_i)$ be the set of unreached desires for agent α_i.

Δ	E_B	U_{δ_1}	U_{δ_2}
$\langle a, b \rangle$	$\{p, q\}$	$\emptyset$	$\{\top \Rightarrow \neg p, \top \Rightarrow \neg q\}$
$\langle a, \neg b \rangle$	$\{p, -q\}$	$\{\top \Rightarrow q\}$	$\{\top \Rightarrow \neg p\}$
$\langle \neg a, b \rangle$	$\{\neg p, q\}$	$\{\top \Rightarrow p\}$	$\{\top \Rightarrow \neg q\}$
$\langle \neg a, \neg b \rangle$	$\{\neg p, \neg q\}$	$\{\top \Rightarrow p, \top \Rightarrow q\}$	$\emptyset$

According to definition 5, for A_1 :
$U(\langle a, b \rangle) > U(\langle a, \neg b \rangle) > U(\langle \neg a, b \rangle) > U(\langle \neg a, \neg b \rangle)$
and for A_2 :
$U(\langle \neg a, \neg b \rangle) > U(\langle \neg a, b \rangle) > U(\langle a, \neg b \rangle) > U(\langle a, b \rangle)$.
None of these decision profiles are dominant decisions, i.e. the agents specifications has no dominant solution with respect to their unreached desires.

The following example illustrates a typical cooperation game.

Example 6 $B_1 = \{a \Rightarrow p\ ,\ b \Rightarrow \neg p \wedge q\}$, $D_1 = \{\top \Rightarrow p \wedge q\}$ $B_2 = \{c \Rightarrow q\ ,\ d \Rightarrow p \wedge \neg q\}$ $D_2 = \{\top \Rightarrow p \wedge q\}$. *The agents have a common goal $p \wedge q$, which they can only reach by cooperation.*

The following example illustrates a qualitative version of the notorious prisoner's dilemma, where the selfish behavior of individual autonomous agents leads to global bad decisions.

Example 7 *Let $A_1 = \{a\}$ (α_1 cooperates), $A_2 = \{b\}$ (α_2 cooperates), and AS be an agent system specifica-*

tion with $D_1 = \{\top \Rightarrow \neg a \wedge b, \top \Rightarrow b, \top \Rightarrow \neg(a \wedge \neg b)\}$, $D_2 = \{\top \Rightarrow a \wedge \neg b, \top \Rightarrow a, \top \Rightarrow \neg(\neg a \wedge b)\}$. *The only Nash decision is* $\{\neg a, \neg b\}$, *whereas both agents would prefer* $\{a, b\}$.

Starting from an agent system specification, we can derive the game specification and in this game specification we can use standard techniques to for example find the Pareto decisions. However, the problem with this approach is that the translation from an agent system specification to a game specification is computationally expensive. For example, a compact agent representation with only a few belief and desire rules may lead to a huge set of decisions if the number of decision variables is high.

The main challenge of qualitative game theory is therefore whether we can bypass the translation to game specification, and define properties directly on the agent system specification. For example, are there particular properties of agent system specification for which we can prove that there always exists a dominant decision for its corresponding derived game specification? A simple example is an agent system specification in which each agent has the same belief and desire rules.

In this paper we do not further pursue these issues, but we turn to our focus of interest: joint goals.

3 Joint goals

In this section we ask ourselves the question whether and how we can interpret a decision profile or equilibrium as goal-based or goal-oriented behavior. We first define decision rules and sets of decision profiles closed under indistinguishable decision profiles.

3.1 Decision rule

A decision rule maps a agent system specification to a set of possible decision profiles.

Definition 8 *A decision rule is a function from agent system specifications to sets of feasible decision profiles.*

Decision theory prescribes a decision maker to select the optimal or best decision, which can be defined as a decision that is not dominated. Is there an analogous prescription for societies of agents? A set of *cooperating* agents has to select an optimal or Pareto decision. We call such cooperating agents a BD rational society of cooperating agents.

Definition 9 *A BD rational society of cooperating agents is a set of agents, defined by an agent system specification AS, that selects a Pareto AS decision.*

In this paper we consider decision rules based on unreached desires. We therefore assume that a decision rule cannot distinguish between decision profiles δ_1 and δ_2 such that $\delta_1 \sim^U \delta_2$. We say that two decision profiles δ and δ' are indistinguishable if $U(\delta) = U(\delta')$, and we call a set of decisions U-closed if the set is closed under indistinguishable decision profiles.

Definition 10 *Let* $AS = \langle S, F, B, D, \geq, \delta^0 \rangle$ *be an agent system specification and* Δ *a set of AS decision profiles.* Δ *is U-closed if* $\delta \in \Delta$ *implies* $\delta' \in \Delta$ *for all AS decision profiles* δ' *such that* $U(\delta) = U(\delta')$.

An example of a decision rule is the function that maps agent system specifications to Pareto decision profiles (a BD rational decision rule). Another example is a function that maps agent system specifications to Nash equilibria if they exist, otherwise to Pareto decisions.

3.2 Goals

In this section we show that every society of agents can be understood as planning for joint goals, whether the decision is reached by cooperation or is given by a (e.g. Nash) equilibrium. We define not only goals which must be reached, called positive goals, but we add negative goals. Negative goals are defined in the following definition as states the agent has to avoid. They function as constraints on the search process of goal-based decisions.

Definition 11 (Goal-based decision) *Let* $AS = \langle S, F, B, D, \geq, \delta^0 \rangle$ *be an agent system specification, and the so-called positive joint goal set* G^+ *and the negative joint goal set* G^- *be subsets of* L_{AW}. *A decision* δ *is a* $\langle G^+, G^- \rangle$ *decision if* $E_B(F \cup \delta) \models_{AW} G^+$ *and for each* $g \in G^-$ *we have* $E_B(F \cup \delta) \not\models_{AW} g$.

Joint goals are defined with respect to a set of decision profiles. The definition of Δ joint goal set encodes a decision profile together with its indistinguishable decision profiles as a positive and negative goal set.

Definition 12 (Δ goal set) *Let* $AS = \langle S, F, B, D, \geq, \delta^0 \rangle$ *be an agent system specification and* Δ *an U-closed set of feasible decisions. The two sets of formulas* $\langle G^+, G^- \rangle \subseteq (L_{AW}, L_{AW})$ *is a* Δ *joint goal set of AS if there is an AS decision* $\delta \in \Delta$ *such that*

$$G^+ = \{y \mid x \Rightarrow y \in D, E_B(F \cup \delta) \models_{AW} x \wedge y\}$$

$$G^- = \{x \mid x \Rightarrow y \in D, E_B(F \cup d) \not\models_{AW} x\}$$

$\langle G^+, G^- \rangle \subseteq (L_{AW}, L_{AW})$ is a feasible joint goal set of AS if there is an U-closed set of feasible decisions Δ such that $\langle G^+, G^- \rangle \subseteq (L_{AW}, L_{AW})$ is a Δ joint goal set of AS. $\langle G^+, G^- \rangle \subseteq (L_{AW}, L_{AW})$ is a joint goal set of AS if there is a set $D' \subseteq D$ such that

$$G^+ = \{y \mid x \Rightarrow y \in D'\}$$

$$G^- = \{x \mid x \Rightarrow y \in D'\}$$

The first part of the representation theorem follows directly from the definitions.

Proposition 4 *Let $AS = \langle S, F, B, D, \geq, \delta^0 \rangle$ be an agent system specification and Δ be an U-closed set of feasible decision profiles. For a decision profile $\delta \in \Delta$ of AS there is a Δ joint goal set $\langle G^+, G^- \rangle$ of AS such that δ is a $\langle G^+, G^- \rangle$ decision.*

Proof. Follows directly from the definitions.

Proposition 5 *Let $AS = \langle S, F, B, D, \geq, \delta^0 \rangle$ be an agent system specification and Δ an U-closed set of feasible decision profiles. For a Δ joint goal set $\langle G^+, G^- \rangle$ of AS, a $\langle G^+, G^- \rangle$ decision is a Δ decision.*

Proof. Follows from U-closed property.

The representation theorem is a combination of Proposition 4 and 5.

Theorem 1 *Let $AS = \langle S, F, B, D, \geq, \delta^0 \rangle$ be an agent system specification and Δ an U-closed set of feasible decision profiles. A decision profile δ is in Δ if and only if there is a Δ goal set $\langle G^+, G^- \rangle$ of AS such that δ is a $\langle G^+, G^- \rangle$ decision profile.*

The second theorem follows from the first one.

Theorem 2 *Let $AS = \langle S, F, B, D, \geq, \delta^0 \rangle$ be an agent system specification. A decision profile δ is a feasible AS decision profile if and only if there is a feasible goal set $\langle G^+, G^- \rangle$ of AS such that δ is a $\langle G^+, G^- \rangle$ decision profile.*

Consider a society of agent that tries to determine its Pareto decision profiles. The game specifications suggest the following algorithm:

```
calculate all decision profiles
for all decision profiles,
  calculate consequences
order all decision profiles
```

The goal-based representation suggest an alternative approach:

```
calculate all joint goals
filter feasible joint goals
for each feasible joint goal set
  find goal-based decision profiles
order these decision profiles
```

In other words, the goal-based representation suggests to calculate the joint goals first. However, the problem to calculate these joint goals is still computationally hard. There are two ways to proceed:

- Define heuristics for the optimization problem;
- Find a fragment of the logic, such that the optimization becomes easier.

For example, consider the following procedure to find (positive) goals:

$$G \subseteq E_{B \cup D}(F \cup \delta^0)$$

This procedure is not complete, because it does not take effects of actions into account. Thus, in the general case it can be used as a heuristic. Moreover, it is complete for the fragment in which the belief rules do not contain effects of actions, i.e. $B \subseteq S \times L_W \times L_W$.[1]

4 Concluding remarks

In this paper we have defined a qualitative decision and game theory in the spirit of classical decision and game theory. The theory illustrates the micro-macro dichotomy by distinguishing the optimization problem from game theoretic equilibria. We also showed that any group decision, whether based on optimization or on an equilibrium, can be represented by positive and negative goals.

We think that the method of this paper is more interesting than its formal results. The decision and game theory are based on several ad hoc choices which need further investigation. For example, the desire rules are defeasible but the belief rules are not (the obvious extension leads to wishful thinking problems as studied in [14, 3]). However, the results suggest that any group decision can be understood as reaching for goals. We hope that further investigations along this line brings the theories and tools used for individual agents and multi agent systems closer together.

[1] We cannot add beliefs on decision variables, as will be clear in the next section. Suppose $A = \{a\}$ and $B = \{\top \rightarrow a\}$. Clearly all decisions should be considered. However, all decision profiles given in Definition 3 imply a and thus all decisions not implying a would be excluded.

References

[1] K. Binmore. *Fun and Games : A Text on Game Theory.* 1992.

[2] C. Boutilier. Toward a logic for qualitative decision theory. In *Proceedings of the KR'94*, pages 75–86, 1994.

[3] J. Broersen, M. Dastani, , and L. van der Torre. Resolving conflicts between beliefs, obligations, intentions and desires. In *Symbolic and Quantitative Approaches to Reasoning and Uncertainty. Proceedings of ECSQARU'01*, LNCS 2143, pages 568–579. Springer, 2001.

[4] J. Broersen, M. Dastani, Z. Huang, J. Hulstijn, and L. van der Torre. The BOID architecture: Conflicts between beliefs, obligations, intentions and desires. In *Proceedings of he Fifth International Conference on Autonomous Agents (AA2001)*, pages 9–16. ACM Press, 2001.

[5] P.R. Cohen and H.J. Levesque. Intention is choice with commitment. *Artificial Intelligence*, 42:213–261, 1990.

[6] J. Doyle and R. Thomason. Background to qualitative decision theory. *AI magazine*, 20(2), summer 1999.

[7] J. Lang. Conditional desires and utilities - an alternative approach to qualitative decision theory. In *In Proceedings of the European Conference on Artificial Intelligence (ECAI'96)*, pages 318–322, 1996.

[8] A. Newell. The knowledge level. *Artificial Intelligence*, 1982.

[9] Martin J. Osborne and Ariel Rubenstein. *A Course in Game Theory.* The MIT Press, Cambridge, Massachusetts, 1994.

[10] A. Rao and M. Georgeff. Modeling rational agents within a BDI architecture. In *Proceedings of the KR91*, 1991.

[11] L. Savage. *The foundations of statistics.* 1954.

[12] H. A. Simon. *The Sciences of the Artificial.* MIT Press, Cambridge, MA, second edition, 1981.

[13] R. Thomason. Practical reason. Presentation at PRR'99 workshop at IJCAI'99.

[14] R. Thomason. Desires and defaults: A framework for planning with inferred goals. In *Proceedings KR 2000*, pages 702–713, 2000.

[15] J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior.* Princeton University Press, Princeton, NJ, 1 edition, 1944.

Interpolation Theorems for Nonmonotonic Reasoning Systems

Eyal Amir
Computer Science Division
University of California at Berkeley
Berkeley, CA 94720-1776, USA
eyal@cs.berkeley.edu

Abstract

Craig's interpolation theorem [Craig, 1957] is an important theorem known for propositional logic and first-order logic. It says that if a logical formula β logically follows from a formula α, then there is a formula γ, including only symbols that appear in both α, β, such that β logically follows from γ and γ logically follows from α. Such theorems are important and useful for understanding those logics in which they hold as well as for speeding up reasoning with theories in those logics. In this paper we present interpolation theorems in this spirit for three nonmonotonic systems: circumscription, default logic and logic programs with the stable models semantics (a.k.a. answer set semantics). These results give us better understanding of those logics, especially in contrast to their nonmonotonic characteristics. They suggest that some *monotonicity* principle holds despite the failure of classic monotonicity for these logics. Also, they sometimes allow us to use methods for the decomposition of reasoning for these systems, possibly increasing their applicability and tractability. Finally, they allow us to build structured representations that use those logics.

1 Introduction

Craig's interpolation theorem [Craig, 1957] is an important theorem known for propositional logic and first-order logic (FOL). It says that if α, β are two logical formulae and $\alpha \vdash \beta$, then there is a formula $\gamma \in \mathcal{L}(\alpha) \cap \mathcal{L}(\beta)$ such that $\alpha \vdash \gamma$ and $\gamma \vdash \beta$ ("$\vdash$" is the classical logical deduction relation; $\mathcal{L}(\alpha)$ is the language of α (the set of formulae built with the nonlogical symbols of α, $L(\alpha)$)). Such interpolation theorems allow us to break inference into pieces associated with sublanguages of the language of that theory [McIlraith and Amir, 2001], for those formal systems in which they hold. In AI, these properties have been used to speed up inference for constraint satisfaction systems (CSPs), propositional logic and FOL (e.g., [Dechter and Pearl, 1988; Darwiche, 1998; McIlraith and Amir, 2001; Dechter and Rish, 1994; Darwiche, 1997; Amir and McIlraith, 2000; Dechter, 1999] and to build structured representations [Darwiche, 1998; Amir, 2000; Darwiche, 1997]

In this paper we present interpolation theorems for three nonmonotonic systems: *circumscription* [McCarthy, 1980], *default logic* [Reiter, 1980] and *logic programs* with the Answer Set semantics [Gelfond and Lifschitz, 1991; Gelfond and Lifschitz, 1988]. In the nonmonotonic setup there are several interpolation theorems for each system, with different conditions for applicability and different form of interpolation. This stands in contrast to classical logic, where Craig's interpolation theorem always holds. Our theorems allow us to use methods for the decomposition of reasoning (a-la [Amir and McIlraith, 2000; McIlraith and Amir, 2001]) under some circumstances for these systems, possibly increasing their applicability and tractability for structured theories. We list the main theorems that we show in this paper below, omitting some of their conditions for simplicity.

For circumscription we show that, under some conditions, $Circ[\alpha; P; Q] \models \beta$ iff there is some set of formulae $\gamma \subseteq \mathcal{L}(\alpha) \cap \mathcal{L}(\beta)$ such that $\alpha \models \gamma$ and $Circ[\gamma; P; Q] \models \beta$. For example, to answer $Circ[BlockW; block; L(BlockW)] \models on(A, B)$, we can compute this formula $\gamma \in \mathcal{L}(\{block, on, A, B\})$ from $BlockW$ *without applying circumscription*, and then solve $Circ[\gamma; block; L(BlockW)] \models on(A, B)$ (where γ may be significantly smaller than $BlockW$).

For default logic, letting $\alpha \mid\!\sim_D \beta$ mean that every extension of $\langle \alpha, D \rangle$ entails β (*cautious entailment*), we show that, under some conditions, if $\alpha \mid\!\sim_D \beta$, then there is a

formula $\gamma \in \mathcal{L}(\alpha \cup D) \cap \mathcal{L}(\beta)$ such that $\alpha \mid\!\sim_D \gamma$ and $\gamma \mid\!\sim_D \beta$. For logic programs we show that if P_1, P_2 are two logic programs and $\varphi \in \mathcal{L}(P_2)$ such that $P_1 \cup P_2 \mid\!\sim^b \varphi$, then there is $\gamma \in \mathcal{L}(P_1) \cap \mathcal{L}(P_2)$ such that $P_1 \mid\!\sim^b \gamma$ and $P_2 \cup \gamma \mid\!\sim^b \varphi$ (here $\mid\!\sim^b$ is the *brave* entailment for logic programs).

This paper focuses on the form of the interpolation theorems that hold for those nonmonotonic logics. We do not address the possible application of these results to the problem of automated reasoning with those logics. Nonetheless, we mention that direct application of those results is possible along the lines already explored for propositional logic and FOL in [Amir and McIlraith, 2000; McIlraith and Amir, 2001].

No interpolation theorems were shown for nonmonotonic reasoning systems before this paper. Nonetheless, some of our theorems for default logic and logic programs are close to the *splitting theorems* of [Lifschitz and Turner, 1994; Turner, 1996], which have already been used to decompose reasoning for those logics. The main difference between our theorems and those splitting theorems is that the latter change some of the defaults/rules involved to provide the corresponding entailment. Also, they do not talk about an interpolant γ, but rather discuss combining extensions.

Since its debut, the nonmonotonic reasoning line of work has expanded and several textbooks now exist that give a fair view of nonmonotonic reasoning and its uses (e.g., [Gabbay *et al.*, 1993]). The reader is referred to those books for background and further details.

2 Logical Preliminaries

In this paper, we use the notion of *logical theory* for every set of axioms in FOL or propositional logic, regardless of whether the set of axioms is deductively closed or not. We use $L(\mathcal{A})$ to denote the signature of $\mathcal{A}$, i.e., the set of nonlogical symbols. $\mathcal{L}(\mathcal{A})$ denotes the language of $\mathcal{A}$, i.e., the set of formulae built with $L(\mathcal{A})$. $Cn(\mathcal{A})$ is the set of logical consequences of $\mathcal{A}$ (i.e., those formulae that are valid consequences of $\mathcal{A}$ in FOL). For a first-order structure, M, in L, we write $U(M)$ for the universe of elements of M. For every symbol, s, in L, we write s^M for the interpretation of s in M.

Finally, we note Craig's Interpolation Theorem.

Theorem 2.1 ([Craig, 1957]) *Let α, β be sentences such that $\alpha \vdash \beta$. Then there is a formula γ involving only nonlogical symbols common to both α and β, such that $\alpha \vdash \gamma$ and $\gamma \vdash \beta$.*

3 Circumscription

3.1 McCarthy's Circumscription: Overview

McCarthy's circumscription [McCarthy, 1980; McCarthy, 1986] is a nonmonotonic reasoning system in which inference from a set of axioms, A, is performed by minimizing the extent of some predicate symbols $\vec{P}$, while allowing some other nonlogical symbols, $\vec{Z}$ to vary.

Formally, McCarthy's circumscription formula

$$Circ[A(P,Z);P;Z] = A(P,Z) \wedge \forall p,z\,(A(p,z) \Rightarrow \neg(p < P)) \quad (1)$$

says that in the theory A, with parameter relations and function vectors (sequence of symbols) P, Z, P is a minimal element such that $A(P,Z)$ is still consistent, when we are allowed to vary Z in order to allow P to become smaller.

Take for example the following simple theory:

$$T = block(B_1) \wedge block(B_2)$$

Then, the circumscription of *block* in T, varying nothing, is

$$Circ[T;block;] = T \wedge \forall p\,[T_{[block/p]} \Rightarrow \neg(p < block)].$$

Roughly, this means that *block* is a minimal predicate satisfying T. Computing circumscription is discussed in length in [Lifschitz, 1993] and others, and we do not expand on it here. Using known techniques we can conclude

$$Circ[T;block;] \equiv \forall x\,(block(x) \Leftrightarrow (x = B_1 \vee x = B_2))$$

This means that there are no other blocks in the world other than those mentioned in the original theory T.

We give the preferential semantics for circumscription that was given by [Lifschitz, 1985; McCarthy, 1986; Etherington, 1986] in the following definition.

Definition 3.1 ([Lifschitz, 1985]) *For any two models M and N of a theory T we write $M \leq_{P,Z} N$ if the models M, N differ only in how they interpret predicates from P and Z and if the extension of every predicate from P in M is a subset of its extension in N. We write $M <_{P,Z} N$ if for at least one predicate in P the extension in M is a strict subset of its extension in N.*

We say that a model M of T is *$\leq_{P,Z}$-minimal* if there is no model N such that $N <_{P,Z} M$.

Theorem 3.2 ([Lifschitz, 1985]: Circumscript. Semantics) *Let T be a finite set of sentences. A structure M is a model of $Circ[T;P;Z]$ iff M is a $\leq_{P,Z}$-minimal model of T.*

This theorem allows us to extend the definition of circumscription to set of infinite number of sentences. In those cases, $Circ[T;P;Z]$ is defined as the set of sentences that hold in all the $\leq_{P,Z}$-minimal models of T. Theorem 3.2 implies that this extended definition is equivalent to the syntactic characterization of the original definition (equation (1)) if T is a finite set of sentences. In the rest of this paper, we refer to this extended definition of circumscription, if T is an infinite set of FOL sentences (we will note those cases when we encounter them).

Circumscription satisfied Left Logical Equivalence (LLE): $T \equiv T'$ implies that $Circ[T;P;Z] \equiv Circ[T';P;Z]$. It also satisfies Right Weakening (RW): $Circ[T;P;Z] \models \varphi$ and $\varphi \Rightarrow \psi$ implies that $Circ[T;P;Z] \models \psi$).

3.2 Model Theory

Definition 3.3 *Let M, N be L-structures, for FOL signature L and language $\mathcal{L}$. We say that N is an* elementary extension *of M (or M is an* elementary substructure *of N), written $M \preceq N$, if $U(M) \subseteq U(N)$ and for every $\varphi(\vec{x}) \in \mathcal{L}$ and vector of elements $\vec{a}$ of $U(M)$, $M \models \varphi(\vec{a})$ iff $N \models \varphi(\vec{a})$.*

$f : M \to N$ is an *elementary embedding* if f is an injective (one-to-one) homomorphism from M to N and for every $\varphi(\vec{x}) \in \mathcal{L}$ and vector $\vec{a} = \langle a_1, ..., a_n \rangle$ of elements from $U(M)$ (i.e., $a_1, .., a_n \in U(M)$), $M \models \varphi(\vec{a})$ iff $N \models \varphi(f(a_1), ..., f(a_n))$.

For FOL signatures $L \subseteq L^+$, and for N an L^+-structure, we say that $N \restriction L$ is the *reduct* of N to L, the L-structure with the same universe of elements as N, and the same interpretation as N for those symbols from L^+ that are in L (there is no interpretation for symbols not in L). For A theory T in a language of L^+, let $Cn^L(T)$ be the set of all consequences of T in the language of L.

The following theorem is a model-theoretic property that is analogous to Craig's interpolation theorem (Theorem 2.1).

Theorem 3.4 (See [Hodges, 1997] p.148) *Let L, L^+ be FOL signatures with $L \subseteq L^+$ and T a theory in the language of L^+. Let M be an L-structure. Then, $M \models Cn^L(T)$ if and only if for some model N of T, $M \preceq N \restriction L$ (M is an elementary substructure of the reduct of N to L).*

3.3 Interpolation in Circumscription

In this section we present two interpolation theorems for circumscription. Those theorems hold for both FOL and propositional logic. Roughly speaking, the first (Theorem 3.8) says that if α nonmonotonically entails β (here this means $Circ[\alpha;P;Q] \models \beta$), then there is $\gamma \subseteq \mathcal{L}(\alpha) \cap \mathcal{L}(\beta \cup P)$ such that α classically entails γ ($\alpha \models \gamma$) and γ nonmonotonically entails β ($Circ[\gamma;P;Q] \models \beta$). In the FOL case this γ can be an infinite set of sentences, and we use the extended definition of Circumscription for infinite sets of axioms for this statement.

The second theorem (Theorem 3.11) is similar to the first, with two main differences. First, it requires that $L(\alpha) \subseteq (P \cup Q)$. Second, it guarantees that γ as above (and some other restrictions) exists iff α nonmonotonically entails β. This is in contrast to the first theorem that guarantees only that *if* part. The actual technical details are more fine than those rough statements, so the reader should refer to the actual theorem statements.

In addition to these two theorems, we present another theorem that addresses the case of reasoning from the union of theories (Theorem 3.10). Before we state and prove those theorems, we prove several useful lemmas.

Our first lemma says that if we are given two theories T_1, T_2, and we know the set of sentences that follow from T_2 in the intersection of their languages, then every model of this set of sentences together with T_1 can be extended to a model of $T_1 \cup T_2$.

Lemma 3.5 *Let T_1, T_2 be two theories, with signatures in L_1, L_2, respectively. Let γ be a set of sentences logically equivalent to $Cn^{L_1 \cap L_2}(T_2)$. For every L_1-structure, $\mathcal{M}$, that satisfies $T_1 \cup \gamma$ there is a $(L_1 \cup L_2)$-structure, $\widehat{\mathcal{M}}$, that is a model of $T_1 \cup T_2$ such that $\mathcal{M} \preceq \widehat{\mathcal{M}} \restriction L_1$.*

PROOF Let $\mathcal{M}$ be a L_1-structure that is a model of $T_1 \cup \gamma$. Then $\mathcal{M} \models \gamma$. Noticing that γ is logically equivalent to $Cn^{L_1 \cap L_2}(T_2)$ (by definition of γ), we get that $\gamma \models Cn^{L_1 \cap L_2}(T_2)$. Consequently, $\gamma \models Cn^{L_1}(T_2)$ because $Cn^{L_1 \cap L_2}(T_2) \equiv Cn^{L_1}(Cn^{L_1 \cap L_2}(T_2)) = Cn^{L_1}(T_2)$.

Now we use Theorem 3.4 with $L = L_1$, $L^+ = L_1 \cup L_2$, $M = \mathcal{M}$ and $T = T_1 \cup T_2$. We know that $\mathcal{M} \models T_1 \cup \gamma$. Thus, $M \models T_1 \cup Cn^{L_1}(T_2)$. To use Theorem 3.4 we need to show that $M \models Cn^{L_1}(T_1 \cup T_2)$. We use Craig's interpolation theorem (Theorem 2.1) to show this is indeed the case.

First notice that $Cn^{L_1}(T_1 \cup T_2) \supseteq Cn^{L_1}(T_1 \cup \gamma)$ is true because $T_2 \models \gamma$. We show that $Cn^{L_1}(T_1 \cup T_2) \subseteq Cn^{L_1}(T_1 \cup \gamma)$. Take $\varphi \in Cn^{L_1}(T_1 \cup T_2)$. By definition, $T_1 \cup T_2 \models \varphi$ and $\varphi \in \mathcal{L}(T_1)$. The deduction theorem for FOL implies that $T_2 \models T_1' \Rightarrow \varphi$, for some finite subset $T_1' \subseteq T_1$. Craig's interpolation theorem for FOL implies that there is $\delta \in \mathcal{L}(T_2) \cap \mathcal{L}(T_1' \Rightarrow \varphi) = \mathcal{L}(T_2) \cap \mathcal{L}(T_1)$ such that $T_2 \models \delta$ and $\delta \models T_1' \Rightarrow \varphi$. Thus, $\delta \in Cn^{L_1 \cap L_2}(T_2) \equiv \gamma$. Consequently, $\gamma \models T_1' \Rightarrow \varphi$. Using the deduction theorem again we get that $T_1' \cup \gamma \models \varphi$, implying that $T_1 \cup \gamma \models \varphi$.

Thus, we showed that $Cn^{L_1}(T_1 \cup T_2) = Cn^{L_1}(T_1 \cup \gamma)$. From $M \models T_1 \cup Cn^{L_1}(T_2)$ and $\gamma = Cn^{L_1}(T_2)$ we get that

$M \models Cn^{L_1}(T_1 \cup T_2)$.

Finally, the conditions of Theorem 3.4 for $L = L_1$, $L^+ = L_1 \cup L_2$, $M = \mathcal{M}$ and $T = T_1 \cup T_2$ hold. We conclude that there is a $(L_1 \cup L_2)$-structure, $\widehat{\mathcal{M}}$, that is a model of $T_1 \cup T_2$ such that $\mathcal{M} \preceq \widehat{\mathcal{M}} \restriction L_1$. ■

Our second lemma says that every $<_{P,Q}$-minimal model of T that is also a model of T' is a $<_{P,Q}$-minimal model of $T \cup T'$.

Lemma 3.6 *Let T be a theory and P, Q vectors of nonlogical symbols. If $\mathcal{M} \models Circ[T; P; Q]$ and $\mathcal{M} \models T \cup T'$, then $\mathcal{M} \models Circ[T \cup T'; P; Q]$.*

PROOF Let $\mathcal{M}$ be a model of $T \cup T'$ such that $\mathcal{M} \models Circ[T; P; Q]$. If there is $\mathcal{M}' <_{P,Q} \mathcal{M}$ such that $\mathcal{M}' \models T \cup T'$, then $\mathcal{M}' \models T$ and $\mathcal{M} \not\models Circ[T; P; Q]$. Contradiction. Thus, there is no such $\mathcal{M}'$ and $\mathcal{M} \models Circ[T \cup T'; P; Q]$. ■

The following theorem is central to the rest of our results in this section. It says that when we circumscribe P, Q in $T_1 \cup T_2$ we can replace T_2 by its consequences in $\mathcal{L}(T_1)$, for some purposes and under some assumptions.

Theorem 3.7 *Let T_1, T_2 be two theories and P, Q two vectors of symbols from $L(T_1) \cup L(T_2)$ such that $P \subseteq L(T_1)$. Let γ a set of sentences logically equivalent to $Cn^{L(T_1) \cap L(T_2)}(T_2)$. Then, for all $\varphi \in \mathcal{L}(T_1)$, if $Circ[T_1 \cup T_2; P; Q] \models \varphi$, then $Circ[T_1 \cup \gamma; P; Q] \models \varphi$.*

PROOF We show that for every model of $Circ[T_1 \cup \gamma; P; Q]$ there is a model of $Circ[T_1 \cup T_2; P; Q]$ whose reduct to $L(T_1)$ is an elementary extension of the reduct of the first model to $L(T_1)$.

Let $\mathcal{M}$ be a $L(T_1 \cup T_2)$-structure that is a model of $Circ[T_1 \cup \gamma; P; Q]$. Then, $\mathcal{M} \models T_1 \cup \gamma$. From Lemma 3.5 we know that there is a $(L_1 \cup L_2)$-structure, $\widehat{\mathcal{M}}$, that is a model of T_2 such that $\mathcal{M} \restriction L(T_1) \preceq \widehat{\mathcal{M}} \restriction L(T_1)$.

Thus, $\widehat{\mathcal{M}}$ is a $\leq_{P,Q}$-minimal model of $T_1 \cup \gamma$. To see this, assume otherwise. Then, there is a model $\mathcal{M}'$ for the signature $L(T_1 \cup T_2)$ such that $\mathcal{M}' <_{P,Q} \widehat{\mathcal{M}}$ and $\mathcal{M}' \models T_1 \cup \gamma$. Take $\mathcal{M}''$ such that the interpretation of all the symbols in $L(T_1)$ is exactly the same as that of $\mathcal{M}'$ and such that the interpretation of all symbols in $L(T_2) \setminus L(T_1)$ is exactly the same as that of $\mathcal{M}$. Then, $\mathcal{M}'' \models T_1 \cup \gamma$ because $T_1 \cup \gamma \subseteq \mathcal{L}(T_1)$. Also, $\mathcal{M}'' <_{P,Q'} \mathcal{M}$, for $Q' = Q \cap L(T_1)$ because $P \subseteq L(T_1)$ and $\mathcal{M}, \widehat{\mathcal{M}}$ agree on the interpretation of symbols in $L(T_1)$ ($\mathcal{M} \restriction L(T_1) \preceq \widehat{\mathcal{M}} \restriction L(T_1)$). Thus, $\mathcal{M}'' <_{P,Q} \mathcal{M}$, since $\mathcal{M}'', \mathcal{M}$ agree on all the interpretation of all symbols in $L(T_2) \setminus L(T_1)$. This contradicts $\mathcal{M} \models Circ[T_1 \cup \gamma; P; Q]$, so $\widehat{\mathcal{M}}$ is a $\leq_{P,Q}$-minimal model of $T_1 \cup \gamma$.

Thus, $\widehat{\mathcal{M}} \models Circ[T_1 \cup \gamma; P; Q]$, and $\widehat{\mathcal{M}} \models T_1 \cup T_2$. From Lemma 3.6 we get that $\widehat{\mathcal{M}} \models Circ[T_1 \cup T_2; P; Q]$.

Now, let $\varphi \in \mathcal{L}(T_1)$ such that $Circ[T_1 \cup T_2; P; Q] \models \varphi$. Then every model of $Circ[T_1 \cup T_2; P; Q]$ satisfies φ. Let $\mathcal{M}$ be a model of $Circ[T_1 \cup \gamma; P; Q]$ in the language $\mathcal{L}(T_1 \cup T_2)$. Then there is $\widehat{\mathcal{M}}$ as above, i.e., $\widehat{\mathcal{M}} \models Circ[T_1 \cup T_2; P; Q]$ and $\mathcal{M} \restriction L(T_1) \preceq \widehat{\mathcal{M}} \restriction L(T_1)$. Thus, $\widehat{\mathcal{M}} \models \varphi$. Since $\mathcal{M} \restriction L(T_1) \preceq \widehat{\mathcal{M}} \restriction L(T_1)$, $\mathcal{M} \models \varphi$. Thus every model of $Circ[T_1 \cup \gamma; P; Q]$ is a model of φ. ■

Theorem 3.8 (Interpolation for Circumscription 1) *Let T be a theory, P, Q vectors of symbols, and φ a formula. If $Circ[T; P; Q] \models \varphi$, then there is $\gamma \subseteq \mathcal{L}(T) \cap \mathcal{L}(\varphi \cup P)$ such that*

$$T \models \gamma \quad \textit{and} \quad Circ[\gamma; P; Q] \models \varphi.$$

Furthermore, this γ can be logically equivalent to the consequences of T in $L(T) \cap L(\varphi \cup P)$.

PROOF We use Theorem 3.7 to find this γ. For T, φ as in the statement of the theorem we define T_1, T_2 as follows. We choose T_1 such that $\varphi \in \mathcal{L}(T_1)$ and $P \subseteq L(T_1)$: Let $T_1 = \{\varphi \vee \neg\varphi\} \cup \tau_1$ for τ_1 a set of tautologies such that $L(\tau_1) = P$. We choose T_2 such that it includes T and has a rich enough vocabulary so that $P, Q \subseteq L(T_1) \cup L(T_2)$. Let $T_2 = T \cup \tau_2$, for τ_2 a set of tautologies such that $L(\tau_2) = Q \setminus L(T_1)$. Let $L_1 = L(T_1)$, $L_2 = L(T_2)$.

Theorem 3.7 guarantees that if $P \subseteq L_1$ then γ from that theorem satisfies $Circ[T_1 \cup T_2; P; Q] \models \psi \Rightarrow Circ[T_1 \cup \gamma; P; Q] \models \psi$ for every $\psi \in \mathcal{L}(T_1)$. This implies that for every $\psi \in \mathcal{L}(\{\varphi\} \cup \tau_1)$, $Circ[T; P; Q] \models \psi \Rightarrow Circ[\gamma; P; Q] \models \psi$. In particular, $Circ[\gamma; P; Q] \models \varphi$, and this γ satisfies our current theorem. ■

This theorem does not hold if we require $\gamma \subseteq \mathcal{L}(T) \cap \mathcal{L}(\varphi)$ instead of $\gamma \subseteq \mathcal{L}(T) \cap \mathcal{L}(\varphi \cup P)$. For example, take $\varphi = Q$, $T = \{\neg P \Rightarrow Q\}$, where P, Q are propositional symbols. $Circ[T; P; Q] \models \varphi$. However, every logical consequence of T in $L(\varphi)$ is a tautology. Thus, if the theorem was correct with our changed requirement, γ would be equivalent to $\emptyset$ and $Circ[\gamma; P; Q] \not\models \varphi$.

Theorem 3.9 *Let T_1, T_2 be two theories, P, Q two vectors of symbols from $L(T_1) \cup L(T_2)$ such that $P \subseteq L(T_1)$ and $P \cup Q \supseteq L(T_2)$. Let γ be a set of sentences logically equivalent to $Cn^{L(T_1) \cap L(T_2)}(T_2)$. Then, for all $\varphi \in \mathcal{L}(T_1)$, if $Circ[T_1 \cup \gamma; P; Q] \models \varphi$, then $Circ[T_1 \cup T_2; P; Q] \models \varphi$.*

PROOF We show that every model of $Circ[T_1 \cup T_2; P; Q]$ is also a model of $Circ[T_1 \cup \gamma; P; Q]$. Let $\mathcal{M}$ be a

$L(T_1 \cup T_2)$-structure that is a model of $Circ[T_1 \cup T_2; P; Q]$. Then $\mathcal{M} \models T_1 \cup T_2$, implying that also $\mathcal{M} \models T_1 \cup \gamma$.

Assume that there is $\mathcal{M}' <_{P,Q} \mathcal{M}$ such that $\mathcal{M}' \models T_1 \cup \gamma$. From Lemma 3.5, there is $\bar{\mathcal{M}}'$ such that $\bar{\mathcal{M}}' \models T_1 \cup T_2$ and $\mathcal{M}' \restriction L(T_1) \preceq \bar{\mathcal{M}}' \restriction L(T_1)$. Since $\mathcal{M}', \bar{\mathcal{M}}'$ agree on all the symbols of $L(T_1)$, we get that $\bar{\mathcal{M}}' \leq_{P,Q} \mathcal{M}'$ (because $P \cup Q \supseteq L(T_2)$). Finally, we get that $\bar{\mathcal{M}}' \leq_{P,Q} \mathcal{M}' <_{P,Q} \mathcal{M}$, contradicting the assumption of $\mathcal{M}$ being $\leq_{P,Q}$-minimal satisfying $T_1 \cup T_2$. Thus, $\mathcal{M}$ is a model of $Circ[T_1 \cup \gamma; P; Q]$.

Now, let $\varphi \in \mathcal{L}(T_1)$ such that $Circ[T_1 \cup \gamma; P; Q] \models \varphi$. Then every model of $Circ[T_1 \cup \gamma; P; Q]$ satisfies φ. Let $\mathcal{M}$ be a model of $Circ[T_1 \cup T_2; P; Q]$ in the language $\mathcal{L}(T_1 \cup T_2)$. Then, $\mathcal{M} \models Circ[T_1 \cup \gamma; P; Q]$ and $\mathcal{M} \models \varphi$. Thus, $Circ[T_1 \cup T_2; P; Q] \models \varphi$. ■

From Theorem 3.7 and Theorem 3.9 we get the following theorem.

Theorem 3.10 (Interpolation Between Theories) *Let T_1, T_2 be two theories, P, Q vectors of symbols in $L(T_1) \cup L(T_2)$ such that $P \subseteq L(T_1)$ and $P \cup Q \supset L(T_2)$. Let γ be a set of sentences logically equivalent to $Cn^{L(T_1) \cap L(T_2)}(T_2)$. Then, for every $\varphi \in \mathcal{L}(T_1)$,*

$$Circ[T_1 \cup \gamma; P; Q] \models \varphi \iff Circ[T_1 \cup T_2; P; Q] \models \varphi$$

Theorem 3.11 (Interpolation for Circumscription 2) *Let T be a theory, P, Q vectors of symbols such that $(P \cup Q) \supseteq L(T)$. Let L_2 be a set of nonlogical symbols. Then, there is $\gamma \in \mathcal{L}(T) \cap \mathcal{L}(L_2 \cup P)$ such that $T \models \gamma$ and for all $\varphi \in \mathcal{L}(L_2)$,*

$$Circ[T; P; Q] \models \varphi \iff Circ[\gamma; P; Q] \models \varphi.$$

Furthermore, this γ can be logically equivalent to the consequences of T in $L(T) \cap (L_2 \cup P)$.

PROOF Let T_1 be a set of tautologies such that $L(T_1) = L_2 \cup P$. Also, let $T_2 = T \cup \tau_2$, for τ_2 a set of tautologies such that $L(\tau_2) = Q \setminus L(T_1)$. Let $L_1 = L(T_1)$, $L_2 = L(T_2)$. Theorem 3.10 guarantees that γ from that theorem satisfies $Circ[\gamma; P; Q] \models \psi \iff Circ[T; P; Q] \models \psi$ for every $\psi \in \mathcal{L}_1 = L_2 \cup P$. ■

The theorems we presented are for parallel circumscription, where we minimize all the minimized predicates in parallel without priorities. The case of prioritized circumscription is outside the scope of this paper.

4 Default Logic

In this section we present interpolation theorems for *propositional* default logic. We also assume that the signature of our propositional default theories is finite (this also implies that our theories are finite).

4.1 Reiter's Default Logic: Overview

In Reiter's default logic [Reiter, 1980] one has a set of facts W (in either propositional or FOL) and a set of defaults D (in a corresponding language). Defaults in D are of the form $\frac{\alpha:\beta_1,\ldots,\beta_n}{\delta}$ with the intuition that if α is proved, and $\beta_1, \ldots, \beta_n$ are consistent (throughout the proof), then δ is proved. α is called the *prerequisite*, $pre(d) = \{\alpha\}$; $\beta_1, \ldots, \beta_n$ are the *justifications*, $just(d) = \{\beta_1, \ldots, \beta_n\}$ and δ is the *consequent*, $cons(d) = \{\delta\}$. We use similar notation for sets of defaults (e.g., $cons(D) = \bigcup_{d \in D} cons(d)$). Notice that the justifications are checked for consistency one at a time (and not conjoined).

Take, for example, the following default theory $T = \langle W, D \rangle$:

$$D = \{\frac{bird(x) : fly(x)}{fly(x)}\} \quad W = \{bird(Tweety)\} \qquad (2)$$

Intuitively, this theory says that birds normally fly and that $Tweety$ is a bird.

An *extension* of $\langle W, D \rangle$ is a set of sentences E that satisfies W, follows the defaults in D, and is minimal. More formally, E is an extension if it is minimal (as a set) such that $\Gamma(E) = E$, where we define $\Gamma(S_0)$ to be S, a minimal set of sentences such that

1. $W \subseteq S$; $S = Cn(S)$.
2. For all $\frac{\alpha:\beta_1,\ldots,\beta_n}{\delta} \in D$ if $\alpha \in S$ and $\forall i \neg\beta_i \notin S_0$, then $\delta \in S$.

The following theorem provides an equivalent definition that was shown in [Marek and Truszczyński, 1993; Risch and Schwind, 1994; Baader and Hollunder, 1995]. A set of defaults, $\mathbb{D}$ is grounded in a set of formulae W iff for all $d \in \mathbb{D}$, $pre(d) \in Cn_{Mon(\mathbb{D})}(W)$, where $Mon(\mathbb{D}) = \{\frac{pre(d)}{cons(d)} \mid d \in \mathbb{D}\}$.

Theorem 4.1 (Extensions in Terms of Generating Defaults) *A set of formulae E is an extension of a default theory $\langle W, D \rangle$ iff $E = Cn(W \cup \{cons(d) \mid d \in D'\})$ for a minimal set of defaults $D' \subseteq D$ such that*

1. *D' is grounded in W and*
2. *for all $d \in D$:*

$$d \in D' \text{ iff } pre(d) \in Cn(W \cup cons(D')) \text{ and for all } \psi \in just(d), \neg\psi \notin Cn(W \cup cons(D')).$$

Every minimal set of defaults $D' \subseteq D$ as mentioned in this theorem is said to be a set of *generating defaults*.

Normal defaults are defaults of the form $\frac{\alpha:\beta}{\beta}$. These defaults are interesting because they are fairly intuitive in nature (if we proved α then β is proved unless previously proved inconsistent). We say that a default theory is *normal*, if all of its defaults are normal.

We define $W \mid\sim_D \varphi$ as cautious entailment sanctioned by the defaults in D, i.e., φ follows from every extension of $\langle W, D\rangle$. We define $W \mid\sim^b_D \varphi$ as brave entailment sanctioned by the defaults in D, i.e., φ follows from at least one extension of $\langle W, D\rangle$.

4.2 Interpolation in Default Logic

In this section we present several flavors of interpolation theorems, most of which are stated for cautious entailment.

Theorem 4.2 (Interpolation for Cautious DL 1) *Let $T = \langle W, D\rangle$ be a propositional default theory and φ a propositional formula. If $W \mid\sim_D \varphi$, then there are γ_1, γ_2 such that $\gamma_1 \in \mathcal{L}(W) \cap \mathcal{L}(D \cup \{\varphi\})$, $\gamma_2 \in \mathcal{L}(W \cup D) \cap \mathcal{L}(\varphi)$ and all the following hold:*

$$W \models \gamma_1 \qquad \gamma_1 \mid\sim_D \gamma_2 \qquad \gamma_2 \models \varphi$$
$$W \mid\sim_D \gamma_2 \qquad \gamma_1 \mid\sim_D \varphi$$

PROOF Let γ_1 be the set of consequences of W in $\mathcal{L}(D \cup \{\varphi\}) \cap \mathcal{L}(W)$. Let $\mathbb{E}$ be the set of extensions of $\langle W, D\rangle$ and $\mathbb{E}'$ the set of extensions of $\langle \gamma_1, D\rangle$. We show that every extension $E' \in \mathbb{E}'$ has an extension $E \in \mathbb{E}$ such that $Cn(E' \cup W) = Cn(E)$. This will show that γ_1 is as needed.

Take $E' \in \mathbb{E}'$ and define $E_0 = Cn(E' \cup W)$. We assume that $L(E') \subseteq L(\mathbb{D})$ because otherwise we can take a logically equivalent extension whose sentences are in $\mathcal{L}(\mathbb{D})$. We show that E_0 satisfies the conditions for extensions of $\langle W, D\rangle$:

1. $W \subseteq E_0$,
2. For all $\frac{\alpha:\beta_1,\ldots,\beta_n}{\delta} \in D$, if $\alpha \in E_0$ and $\forall i\ \neg\beta_i \notin E_0$, then $\delta \in E_0$.

The first condition holds by definition of E_0. The second condition holds because every default that is consistent with E_0 is also consistent with E' and vice versa. We detail the second condition below.

For the first direction (every default that is consistent with E_0 is also consistent with E'), let $\frac{\alpha:\beta_1,\ldots,\beta_n}{\delta} \in D$ be such that $\alpha \in E_0$. We show that $\alpha \in E'$.

By definition, $\alpha \in \mathcal{L}(D)$. $\alpha \in E_0$ implies that $E' \cup W \models \alpha$ because $Cn(E' \cup W) = E_0$. Using the deduction theorem for propositional logic we get $W \models E' \Rightarrow \alpha$ (taking E' here to be a finite set of sentences that is logically equivalent to E' in $\mathcal{L}(\mathbb{D})$ (there is such a finite set because we assume that $L(\mathbb{D})$ is finite)). Using Craig's interpolation theorem for propositional logic, there is $\gamma \in \mathcal{L}(W) \cap \mathcal{L}(E' \Rightarrow \alpha)$ such that $W \models \gamma$ and $\gamma \models E' \Rightarrow \alpha$. However, this means that $\gamma_1 \models \gamma$, by the way we chose γ_1. Thus $\gamma_1 \models E' \Rightarrow \alpha$. Since $E' \subseteq \gamma_1$ we get that $E' \models \alpha$. Since $E' = Cn(E')$ we get that $\alpha \in E'$.

The case is similar for δ: if $\delta \in E_0$ then $\delta \in E'$ by the same argument as given above for $\alpha \in E_0 \Rightarrow \alpha \in E'$. Finally, if $\forall i\ \neg\beta_i \notin E_0$ then $\forall i\ \neg\beta_i \notin E'$ because $E' \subseteq E_0$.

The opposite direction (every default that is consistent with E' is also consistent with E_0) is similar to the first one.

Thus, E_0 satisfies those two conditions. However, it is possible that E_0 is not a minimal such set of formulae. If so, Theorem 4.1 implies that there is a strict subset of the generating defaults of E_0 that generate a different extension. However, we can apply this new set of defaults to generate an extension that is smaller than E', contradicting the fact that E' is an extension of $\langle \gamma_1, D\rangle$.

Now, if φ logically follows in all the extensions of $\langle W, D\rangle$ then it must also follow from every extension of $\langle \gamma_1, D\rangle$ together with W. Let $\Lambda = E_1 \vee \ldots \vee E_n$, for $E_1, \ldots, E_n$ the (finite) set of (logically non-equivalent) extensions of $\langle W, D\rangle$ (we have a finite set of those because $L(W) \cup L(D)$ is finite). Then, $\Lambda \models \varphi$. Take $\gamma_2 \in \mathcal{L}(\Lambda) \cap \mathcal{L}(\varphi)$ such that $\Lambda \models \gamma_2$ and $\gamma_2 \models \varphi$, as guaranteed by Craig's interpolation theorem (Theorem 2.1). These γ_1, γ_2 are those promised by the current theorem: $W \models \gamma_1$, $\gamma_2 \models \varphi$, $W \mid\sim_D \gamma_2$, $\gamma_1 \mid\sim_D \gamma_2$ and $\gamma_1 \mid\sim_D \varphi$. ■

Theorem 4.3 (Interpolation for Cautious DL 2) *Let $T = \langle W, D\rangle$ be a propositional default theory and φ a propositional formula. If $W \mid\sim_D \varphi$, then there are $\gamma_1, \gamma_2 \in \mathcal{L}(W) \cap \mathcal{L}(D)$, and all the following hold:*

$$W \models \gamma_1 \qquad \gamma_1 \mid\sim_D \gamma_2$$
$$\{\gamma_2\} \cup W \models \varphi \qquad W \mid\sim_D \gamma_2$$

The proof is similar to the one for Theorem 4.2.

Corollary 4.4 *Let $\langle W, D\rangle$ be a default theory and φ a formula. If $W \mid\sim_D \varphi$, then there is a set of formulae, $\gamma \in \mathcal{L}(W \cup D) \cap \mathcal{L}(\varphi)$ such that $W \mid\sim_D \gamma$ and $\gamma \mid\sim_D \varphi$.*

PROOF Follows immediately from Theorem 4.2 with γ_2 there corresponding to our needed γ. ■

It is interesting to note that we do not get stronger interpolation theorems for prerequisite-free normal default theories. [Imielinski, 1987] provided a modular translation of

normal default theories with no prerequisites into circumscription, but Theorem 3.8 does not lead to better results. In particular, the counter example that we presented after that theorem can be massaged to apply here too.

Theorem 4.5 (Interpolation Between Default Extensions) *Let $\langle W_1, D_1\rangle, \langle W_2, D_2\rangle$ be default theories such that $L(cons(D_2)) \cap L(pre(D_1) \cup just(D_1) \cup W_1) = \emptyset$. Let φ be a formula such that $\varphi \in \mathcal{L}(W_2 \cup D_2)$. If there is an extension E of $\langle W_1 \cup W_2, D_1 \cup D_2\rangle$ in which φ holds, then there is a formula $\gamma \in \mathcal{L}(W_1 \cup D_1) \cap \mathcal{L}(W_2 \cup D_2)$, an extension E_1 of $\langle W_1, D_1\rangle$ such that $Cn(E_1) \cap \mathcal{L}(W_2 \cup D_2) = \gamma$, and an extension E_2 of $\langle W_2 \cup \{\gamma\}, D_2\rangle$ such that $E_2 \models \varphi$.*

PROOF Let $D_1' \subseteq D_1$ be the set of generating defaults of E that belong to D_1. Notice that these defaults are grounded in W_1 because there is no information that may have come from applying the rest of the generating defaults in E (we required that $cons(D_2) \cap (pre(D_1) \cup just(D_1) \cup W) = \emptyset$). Let E_1 be the extension of $\langle W_1, D_1\rangle$ defined using the generating defaults in D_1'.

Let $\gamma \in \mathcal{L}(W_1 \cup D_1) \cap \mathcal{L}(W_2 \cup D_2)$ be the conjunction of the sentences in that language that follow from E_1. Let $D_2' \subseteq D_2$ be the set of generating defaults of E that belong to D_2. Notice that these defaults are grounded in $W_2 \cup \gamma$ because there is no information that D_1' may contribute that is not already in γ (we required that $cons(D_2) \cap (pre(D_1) \cup just(D_1) \cup W) = \emptyset$). Let E_2 be the extension of $\langle W_2 \cup \gamma, D_2\rangle$ defined with the generating defaults in D_2'.

Now, $E_1 \cup E_2 \equiv E$, and γ is the set of sentences that follow from E_1 in $\mathcal{L}(E_1) \cap \mathcal{L}(E_2 \cup \varphi)$. $E_2 \models \varphi$ because of Craig's interpolation theorem (Theorem 2.1) for propositional logic: $E_1 \cup E_2 \models \varphi$ implies that $E_1 \models E_2 \Rightarrow \varphi$, and Craig's interpolation theorem guarantees the existence of $\gamma' \in \mathcal{L}(E_1) \cap \mathcal{L}(E_2 \cup \varphi)$ such that $E_1 \models \gamma'$ and $\gamma' \models E_2 \Rightarrow \varphi$. Thus, $\gamma' \in \gamma$ and $\gamma \models E_2 \Rightarrow \varphi$. This implies $\gamma \cup E_2 \models \varphi$ which implies that $E_2 \models \varphi$. ∎

It is interesting to notice that the reverse direction of this theorem does not hold. For example, if we have two extensions E_1, E_2 as in the theorem statement, it is possible that E_1 uses a default with justification β, but $W_2 \models \neg\beta$. Strengthening the condition of the theorem, i.e., demanding that $L(W_2 \cup cons(D_2)) \cap L(pre(D_1) \cup just(D_1) \cup W_1) = \emptyset$, is not sufficient either. For example, if D_1 includes two defaults $d_1 = \frac{:}{a \Rightarrow \neg\beta}$, and $d_2 = \frac{:\beta}{\varphi}$, $W_1 = \emptyset$, D_2 includes no defaults and $W_2 = \{a\}$ then there is no extension of $\langle W_1 \cup W_2, D_1 \cup D_2\rangle$ that implies φ, for $\varphi = \{c\}$.

Further strengthening the conditions of the theorem gives the following:

Theorem 4.6 (Reverse Direction of Theorem 4.5) *Let $\langle W_1, D_1\rangle, \langle W_2, D_2\rangle$ be default theories such that $L(W_2 \cup cons(D_2)) \cap L(D_1 \cup W_1) = \emptyset$. Let φ be a formula such that $\varphi \in \mathcal{L}(W_2 \cup D_2)$. There is an extension E of $\langle W_1 \cup W_2, D_1 \cup D_2\rangle$ in which φ holds only if there is a formula $\gamma \in \mathcal{L}(W_1 \cup D_1) \cap \mathcal{L}(W_2 \cup D_2)$, an extension E_1 of $\langle W_1, D_1\rangle$ such that $Cn(E_1) \cap \mathcal{L}(W_2 \cup D_2) = \gamma$, and an extension E_2 of $\langle W_2 \cup \{\gamma\}, D_2\rangle$ such that $E_2 \models \varphi$.*

PROOF Let E_1, E_2 be as in the statement of the theorem. Let π_1, π_2 be the sets of defaults applied in E_1, E_2, respectively. Let $E = Cn(E_1 \cup E_2)$, and let $\pi = \pi_1 \cup \pi_2$. We show that E is an extension of $\langle W_1 \cup W_2, D_1 \cup D_2\rangle$ such that $E \models \varphi$ as needed.

First, $E \supseteq W_1 \cup W_2$ by E's definition. For every $d = \frac{\alpha:\beta_1,\ldots,\beta_n}{\gamma_d} \in \pi$, α and γ_d hold because they hold in one of E_1, E_2 ($d \in \pi = \pi_1 \cup \pi_2$). Assume that $\beta_i \in E$ for some $i \leq n$. Then, $E_1 \cup E_2 \models \neg\beta_i$, implying that $E_1 \models E_2 \Rightarrow \neg\beta_i$ (we treat E_1, E_2 here as finite sets of formulae because they are in propositional logic).

If $d \in \pi_1$, then clearly $\neg\beta_i \in \mathcal{L}(W_1 \cup D_1)$. $E_1 \cup \{\beta_i\} \models \neg E_2$. Using Craig's interpolation theorem we get that γ from the theorem's statement satisfies $\{\gamma\} \cup \{\beta_i\} \models \neg E_2$. Consequently, $E_2 \models \gamma \Rightarrow \neg\beta_i$. However, $E_2 = Cn(cons(\pi_2) \cup W_2 \cup \{\gamma\})$, and $L(W_2 \cup cons(D_2)) \cap L(W_1 \cup D_1) = \emptyset$. This means that $L(cons(\pi_2) \cup W_2) \cap L(\gamma \wedge \beta_i) = \emptyset$, implying that $\gamma \models \neg\beta$, contradicting $\neg\beta \notin E_1$.

If $d \in \pi_2$, then $\neg\beta_i \in \mathcal{L}(W_2 \cup D_2)$. Since $E_1 \models E_2 \Rightarrow \neg\beta_i$, we get from Craig's interpolation theorem that $\gamma \models E_2 \Rightarrow \neg\beta_i$. However, $\gamma \in E_2$ by the definition of E_2. Thus, $E_2 \models \neg\beta$, contradicting the fact that d is a default applied in E_2.

In conclusion, $\neg\beta \notin E$. Thus, all the defaults in π are applied in E. It is also simple to see that no other default is applied in E.

If there is a default $d = \frac{\alpha:\beta_1,\ldots,\beta_n}{\gamma_d} \in D_1$ that should apply in E but is not in π, then its preconditions and justifications hold in E. However, this means that α follows from $E_1 \cup E_2$, and $E_2 = Cn(cons(\pi_2) \cup W_2 \cup \{\gamma\})$. Similar to the argument above we get that $E_1 \models \alpha$. Similarly, we get that if $\beta_i \notin E$ then $E_1 \not\models \neg\beta_i$, implying that d should have applied in E_1, contradicting the fact that E_1 is an extension of $\langle W_1, D_1\rangle$.

If there is a default $d = \frac{\alpha:\beta_1,\ldots,\beta_n}{\gamma_d} \in D_2$ that should apply in E but is not in π, then its preconditions and justifications hold in E. A similar argument to the one above shows that it should have applied in E_2 too, contradicting the fact that E_2 is an extension.

Minimality of E follows from that of E_1, E_2. Thus, E is an extension of $\langle W_1 \cup W_2, D_1 \cup D_2\rangle$ as needed. ∎

Corollary 4.7 (Interpolation for Brave DL) *Let $\langle W_1, D_1\rangle, \langle W_2, D_2\rangle$ be default theories such that $L(cons(D_2)) \cap L(pre(D_1) \cup just(D_1) \cup W_1) = \emptyset$. Let φ be a formula such that $\varphi \in \mathcal{L}(W_2 \cup D_2)$. If $W_1 \cup W_2 \mid\sim^b_{D_1 \cup D_2} \varphi$, then there is a formula, $\gamma \in \mathcal{L}(W_1 \cup D_1) \cap \mathcal{L}(W_2 \cup D_2)$, such that $W_1 \mid\sim^b_{D_1} \gamma$ and $W_2 \cup \{\gamma\} \mid\sim^b_{D_2} \varphi$.*

Corollary 4.7 does not hold for the cautious case (where we look at all the extensions and choose φ and γ satisfied by all of them): Let $W_1 = W_2 = \emptyset$, $D_1 = \{\frac{:b}{b}, \frac{:\neg b}{\neg b}\}$ and $D_2 = \{\frac{b:c}{c}, \frac{\neg b:c}{c}\}$. There are two extensions, in both of which c is proved, but $W_1 \not\mid\sim_{D_1} \gamma$ only for $\gamma \equiv TRUE$.

Better interpolation theorems may hold (e.g., theorems that do not depend on $cons(D_2)$, $pre(D_1)$, etc., and provide $\gamma \in \mathcal{L}(\langle W_1, D_1\rangle) \cap \mathcal{L}(\langle W_2, D_2\rangle)$), if we consider the entailment between two default theories ($\langle W_1, D_1\rangle \mid\sim \langle W_2, D_2\rangle$). These are outside the scope of this paper.

Finally, Corollary 4.7 and Theorem 4.5 are similar to the *splitting theorem* of [Turner, 1996], which is provided for default theories with $W = \emptyset$ (there is a modular translation that converts every default theory to one with $W = \emptyset$). We briefly review this result. A *splitting set* for a set of defaults D is a subset A of $L(D)$ such that $pre(D), just(D), cons(D) \subseteq \mathcal{L}(A) \cup \mathcal{L}(L(D) \setminus A)$ and $\forall d \in D\ (cons(d) \notin \mathcal{L}(L(D) \setminus A) \Rightarrow L(d) \subseteq A)$. Let $B = L(D) \setminus A$. The base of D relative to A is $b_A(D) = \{d \in D \mid L(d) \subseteq A\}$. For a set of sentences $X \subseteq \mathcal{L}(A)$, we define

$$e_A(D, X) = \left\{ \frac{\bigwedge(\{a_i\}_{i\le n} \cap \mathcal{L}(B)) : \{b_i\}_{i\le m} \cap \mathcal{L}(B)}{c} \;\middle|\; \begin{array}{l} \frac{\bigwedge_{i=1}^{n} a_i : b_1, \ldots, b_m}{c} \in D \setminus b_A(D), \\ \forall i \le n (a_i \in \mathcal{L}(A) \Rightarrow a_i \in Cn^A(X)), \\ \forall i \le m (\neg b_i \notin Cn^A(X)) \end{array} \right\}$$

Theorem 4.8 ([Turner, 1996]) *Let A be a splitting set for a default theory D over $\mathcal{L}(U)$. A set E of formulae is a consistent extension of D iff $E = Cn^{L(D)}(X \cup Y)$, for some consistent extension X of $b_a(D)$ over $\mathcal{L}(A)$ and Y a consistent extension of $e_A(D, X)$ over $\mathcal{L}(L(D) \setminus A)$.*

Roughly speaking, this theorem finds an extension X of the *base* ($b_A(D)$) and converts $D \setminus b_A(D)$ using this X into a theory $e_A(D, X)$. Then, an extension Y for $e_A(D, X)$ completes the extension for D if $X \cup Y$ is consistent. In contrast, our theorem does not change $D \setminus b_A(D)$, but it is somewhat weaker, in that it only provides a necessary condition for $D \mid\sim^b \varphi$. (however, notice that this *weaker* form is typical for interpolation theorems).

5 Logic Programs

In this section we provide interpolation theorems for logic programs with the stable models semantics. We use the fact the logic programs are a special case of default logic, and the results are straightforward.

An *extended disjunctive logic program* [Gelfond and Lifschitz, 1988; Gelfond and Lifschitz, 1990; Gelfond and Lifschitz, 1991; Przymusiński, 1991] is a set of *rules*. Each rule, r, is written as an expression of the form

$$L_1|...|L_l \leftarrow A_1, ..., A_n, notB_1, ..., notB_m$$

where $L_1, ..., L_l, A_1, ..., A_n, B_1, ..., B_m$ are literals, that is, atomic formulae or their (classic) negations, $L_1, ..., L_l$ are the *head literals*, $head(r)$, $A_1, ..., A_n$ are the *positive subgoals*, $pos(r)$, and $B_1, ..., B_m$ are the *negated subgoals*, $neg(r)$.

A program P is *positive* if none of its rules includes negated subgoals. A set of literals, X, is *closed* under a positive program, P, if, for every rule $r \in P$ such that $pos(r) \subseteq X$, $head(r) \cap X \neq \emptyset$. A set of literals is *logically closed* if it consistent or contains all literals. An *answer set* for a positive program, P is a minimal set of literals that is both closed under P and logically closed.

For an arbitrary logic program, P, and a set of literals, X, we say that X is an *answer set* for a program P if X is an answer set for P^X, where P^X is defined to include a rule r' iff $neg(r') = \emptyset$ and there is $r \in P$ such that $head(r') = head(r)$, $pos(r') = pos(r)$, and $neg(r) \cap X = \emptyset$.

Logic programs with answer-set semantics were shown equivalent to default logic in several ways. For normal rules (rules of the form $A \leftarrow B_1, ..., B_m,\ not\ C_1, ...,\ not\ C_n$, where $A, B_1, ..., B_m, C_1, ..., C_n$ are atoms (i.e., no disjunction or classic negation is allowed)), [Gelfond and Lifschitz, 1991] translated every normal rule of the form $A \leftarrow B_1, ..., B_m,\ not\ C_1, ...,\ not\ C_n$, into a default

$$\frac{B_1 \wedge ... \wedge B_m\ :\ \neg C_1, ..., \neg C_n}{A}.$$

Under this mapping, the stable models of a logic program coincide with the extensions of the corresponding default theory (Facts in the logic program are translated to facts in the default theory, while rules are translated to defaults).

[Sakama and Inoue, 1993] showed that disjunctive logic programs (no classic negation) with the stable model semantics can be translated to prerequisite-free default theories as follows:

1. For a rule $A_1|...|A_l \leftarrow B_1, ..., B_m,\ not\ C_1, ...,\ not\ C_n$ in P, we get

the default

$$\frac{: \neg C_1, ..., \neg C_n}{B_1 \wedge ... \wedge B_m \Rightarrow A_1 \vee ... \vee A_l}$$

2. For each atom A appearing in P, we get the default $\frac{:\neg A}{\neg A}$

Each stable model of P is the set of atoms in some extension of D_P, and the set of atoms in an extension of D_P is a stable model of P (notice that, in general, an extension of D_P can include sentences that are not atoms and are not subsumed by atoms in that extension). [Sakama and Inoue, 1993] provide a similar translation to extended disjunctive logic programs by first translating those into disjunctive logic programs (a literal $\neg A$ is translated to a new symbol, A'), showing that a similar property holds for this class of programs.

We define $P \mid\!\sim \varphi$ as cautious entailment sanctioned from the logic program P, i.e., φ follows from stable model of P. We define $P \mid\!\sim^b \varphi$ as brave entailment sanctioned from the logic program P, i.e., φ follows from at least one stable model of P.

From the last translation above we get the following interpolation theorems.

Theorem 5.1 (Interpolation for Stable Models (Cautious)) *Let P be a logic program and let φ be a formula such that $P \mid\!\sim \varphi$. Then, there is a formula $\gamma \in \mathcal{L}(P) \cap \mathcal{L}(\varphi)$ such that $P \mid\!\sim \gamma$ and $\gamma \models \varphi$.*

PROOF Follows immediately from Theorem 4.2 with γ_2 over there corresponding to our needed γ. ■

Theorem 5.2 (Interpolation for Stable Models (Brave)) *Let P_1, P_2 be logic programs such that $head(P_2) \cap body(P_1) = \emptyset$. Let $\varphi \in \mathcal{L}(P_2)$ be a formula such that $P_1 \cup P_2 \mid\!\sim^b \varphi$. Then, there is a formula $\gamma \in \mathcal{L}(P_1) \cap \mathcal{L}(P_2)$ such that $P_1 \mid\!\sim^b \gamma$ and $\gamma \cup P_2 \mid\!\sim^b \varphi$.*

PROOF Follows directly from the reduction into default logic and Corollary 4.7. ■

The last theorem is similar to the *splitting theorem* of [Lifschitz and Turner, 1994]. This theorem finds an answer set X of the *bottom* (P_1) and converts P_2 using this X into a program P_2'. Then, an answer set Y for P_2' completes the answer set for $P_1 \cup P_2$ if $X \cup Y$ is consistent. In contrast, our theorem does not change P_2, but it is somewhat weaker, in that it does only provides a necessary condition for $P_1 \cup P_2 \mid\!\sim^b \varphi$ (this is the typical form of an interpolation theorem).

6 Summary

We presented interpolation theorems that are applicable to the nonmonotonic systems of circumscription, default logic and Answer Set Programming (a.k.a. Stable Models Semantics). These results are somewhat surprising and revealing in that they show particular structure for the nonmonotonic entailments associated with the different systems. They promise to help in reasoning with larger systems that are based on these nonmonotonic systems.

Several questions remain open. First, γ promised by our theorems is not always finite (in the FOL case). This is in contrast to classical FOL, where the interpolant is always of finite length. What conditions guarantee that it is finite in our setup? We conjecture that this will require the partial order involved in the circumscription to be *smooth*. Second, are there better interpolation theorems for the prioritized case of those systems? Also, what is the shape of the interpolation theorems specific for prerequisite-free seminormal defaults? Further, our results for default logic and logic programs are propositional. How do they extend to the FOL case?

Finally, the theorems for default logic and Logic Programming promise that $\alpha \mid\!\sim_D \beta$ implies the existence of γ such that $\alpha \mid\!\sim_D \gamma$ and $\gamma \mid\!\sim_D \beta$. However, we do not know that the other direction holds, i.e., that the existence of γ such that $\alpha \mid\!\sim_D \gamma$ and $\gamma \mid\!\sim_D \beta$ implies that $\alpha \mid\!\sim_D \beta$. Can we do better than Theorem 4.6 for different cases?

7 Acknowledgments

I wish to thank Esra Erdem, Vladimir Lifschitz and Leora Morgenstern for reading and commenting on different drafts of this manuscript. I also thank the reviewers of NMR'02 for their diligent and detailed comments. This research was supported by DARPA grant N66001-00-C-8018 (RKF program).

References

[Amir and McIlraith, 2000] Eyal Amir and Sheila McIlraith. Paritition-based logical reasoning. In *Principles of Knowledge Representation and Reasoning: Proc. Seventh Int'l Conference (KR '2000)*, pages 389–400. Morgan Kaufmann, 2000.

[Amir, 2000] Eyal Amir. (De)Composition of situation calculus theories. In *Proc. National Conference on Artificial Intelligence (AAAI '00)*, pages 456–463. AAAI Press/MIT Press, 2000.

[Baader and Hollunder, 1995] Franz Baader and Bernard Hollunder. Embedding defaults into terminological

knowledge representation formalisms. *Journal of Automated Reasoning*, 14:149–180, 1995.

[Craig, 1957] William Craig. Linear reasoning. a new form of the herbrand-gentzen theorem. *Journal of Symbolic Logic*, 22:250–268, 1957.

[Darwiche, 1997] Adnan Darwiche. A logical notion of conditional independence. *Artificial Intelligence*, 97(1–2):45–82, 1997.

[Darwiche, 1998] A. Darwiche. Model-based diagnosis using structured system descriptions. *Journal of Artificial Intelligence Research*, 8:165–222, 1998.

[Dechter and Pearl, 1988] Rina Dechter and Judea Pearl. Tree Clustering Schemes for Constraint Processing. In *Proc. National Conference on Artificial Intelligence (AAAI '88)*, 1988.

[Dechter and Rish, 1994] Rina Dechter and Irina Rish. Directional resolution: The davis-putnam procedure, revisited. In *Principles of Knowledge Representation and Reasoning: Proc. Fourth International Conference (KR '94)*, pages 134–145. Morgan Kaufmann, 1994.

[Dechter, 1999] Rina Dechter. Bucket elimination: A unifying framework for reasoning. *Artificial Intelligence*, 113(1–2):41–85, 1999.

[Etherington, 1986] David Etherington. *Reasoning with incomplete Information*. PhD thesis, University of British Columbia, 1986.

[Gabbay *et al.*, 1993] D.M. Gabbay, C.J Hogger, and J.A.Robinson, editors. *Handbook of Logic in Artificial Intelligence and Logic Programming, Volume 3: Nonmonotonic Reasoning and Uncertain Reasoning*. Oxford University Press, Great Britain, 1993.

[Gelfond and Lifschitz, 1988] Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. In *Logic Programming: Proceedings of the Fifth International Conference and Symposium*, pages 1070–1080, 1988.

[Gelfond and Lifschitz, 1990] Michael Gelfond and Vladimir Lifschitz. Logic Program with Classical Negation. In David H. D. Warren and Peter Szeredi, editors, *Proceedings of the 7th Int. Conf. on Logic Programming*, pages 579–597. MIT, June 1990.

[Gelfond and Lifschitz, 1991] Michael Gelfond and Vladimir Lifschitz. Classical Negation in Logic Programs and Disjunctive Databases. *New Generation Computing*, 9, 1991.

[Hodges, 1997] Wilfrid Hodges. *A shorter model theory*. Cambridge University Press, 1997.

[Imielinski, 1987] Tomasz Imielinski. Results on translating defaults to circumscription. *Artificial Intelligence*, 32(1):131–146, April 1987.

[Lifschitz and Turner, 1994] Vladimir Lifschitz and Hudson Turner. Splitting a logic program. In *Proc. Eleventh Int'l Conf. on Logic Programming*, pages 23–37. MIT Press, 1994.

[Lifschitz, 1985] Vladimir Lifschitz. Computing circumscription. In *Proc. of IJCAI-85*, pages 121–127, 1985.

[Lifschitz, 1993] Vladimir Lifschitz. Circumscription. In D.M. Gabbay, C.J Hogger, and J.A Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming, Volume 3: Nonmonotonic Reasoning and Uncertain Reasoning*. Oxford University Press, 1993.

[Marek and Truszczyński, 1993] Viktor M. Marek and Mirek Truszczyński. *Nonmonotonic Logics; Context-Dependent Reasoning*. Springer Verlag, Berlin-Heidelberg-New York, 1st edition, 1993.

[McCarthy, 1980] John McCarthy. Circumscription—A Form of Non-Monotonic Reasoning. *Artificial Intelligence*, 13:27–39, 1980.

[McCarthy, 1986] John McCarthy. Applications of Circumscription to Formalizing Common Sense Knowledge. *Artificial Intelligence*, 28:89–116, 1986.

[McIlraith and Amir, 2001] Sheila McIlraith and Eyal Amir. Theorem proving with structured theories. In *Proc. Seventeenth International Joint Conference on Artificial Intelligence (IJCAI '01)*, pages 624–631. Morgan Kaufmann, 2001.

[Przymusiński, 1991] Teodor Przymusiński. Stable semantics for disjunctive programs. *New Generation Computing*, 9:401–424, 1991.

[Reiter, 1980] Raymond Reiter. A logic for default reasoning. *Artificial Intelligence*, 13 (1–2):81–132, 1980.

[Risch and Schwind, 1994] V. Risch and Camila Schwind. Tableau-based characterization and theorem proving for default logic. *Journal of Automated Reasoning*, 13:223–242, 1994.

[Sakama and Inoue, 1993] Chiaki Sakama and Katsumi Inoue. Relating disjunctive logic programs to default theories. In *Logic Programming and Non-monotonic Reasoning (LPNMR'93)*, pages 266–282, 1993.

[Turner, 1996] Hudson Turner. Splitting a default theory. In *Proc. National Conference on Artificial Intelligence (AAAI '96)*, pages 645–651. AAAI Press, 1996.

Compositionality Results for Stratified Nonmonotone Operators.

David Gilis **Marc Denecker**
Department of Computer Science, K.U.Leuven
Celestijnenlaan 200A, B-3001 Heverlee
Belgium

Abstract

Approximation theory is an extension of Tarski's least fixpoint theory of monotone lattice operators to the generalised case of arbitrary nonmonotone operators. Approximation theory captures intuitions and principles found in several nonmonotonic reasoning systems and describes all main 2-, 3- and 4-valued semantics of logic programming, default logic and autoepistemic logic.

This paper investigates a composition property of lattice operators and their fixpoints. We investigate the circumstances in which an operator defined on a product lattice can be split up in operators on the component lattices, and show how its various fixpoints can be obtained by composing the appropriate fixpoints of the component operators. In the context of a specific nonmonotonic logic, our theorem property translates into compositionality theorems.

1 INTRODUCTION

The formal semantics of nonmonotonic systems is often described in terms of elements of lattices. In the case of logic programming, these lattices are the collections of 2-, 3- or 4-valued interpretations. In the case of autoepistemic and default logic, the lattices are the collection of *possible world sets* and its bilattice. The semantics of a theory in these systems can then be characterised by fixpoints of lattice operators associated to this theory. For example, in this way Fitting proposed an elegant algebraic treatment of all major 2-, 3- and 4-valued semantics of logic programs [Fitting, 1999], that is, the supported-model semantics [Clark, 1978], stable-model semantics [Gelfond and Lifschitz, 1988], Kripke-Kleene semantics [Fitting, 1985; Kunen, 1987] and well-founded semantics [van Gelder *et al.*, 1991].

This algebraic approach was further elaborated in *Approximation theory* [Denecker *et al.*, 2000a; Denecker *et al.*, 2000b; Denecker *et al.*, 2002], an abstract algebraic study of the principles underlying nonmonotone systems. Approximation theory is a fixpoint theory of general nonmonotone operators in a lattice and its bilattice. Central to this approach is the concept of an approximation of a lattice operator O. An approximation is an operator defined on a certain bilattice (the product of the lattice by itself, with two appropriately defined lattice orderings). Using purely algebraic techniques, for an approximation operator for O the notion of the stable operator and the concepts of the Kripke-Kleene, well-founded and stable fixpoints is introduced, and it is shown how they provide information about fixpoints of the operator O. In [Denecker *et al.*, 2000a; Denecker *et al.*, 2000b] is shown that each theory in the above mentioned nonmonotone systems, determines a characteristic operator on its semantic lattice and an approximation on the bilattice and that all existing (2-, 3- and 4-valued) semantics of these systems are described by the various types of fixpoints of these operators. In particular, in the context of logic programming, the operator and its approximation is the 2-valued, respectively 4-valued $\mathcal{T}_P$ operator, and the Kripke-Kleene, well-founded and stable models correspond to the Kripke-Kleene model, well-founded and stable fixpoints of $\mathcal{T}_P$.

Approximation theory has emerged as a powerful tool in investigations of semantics of these logics. It allows to establish a comprehensive semantic treatment for nonmonotonic logics and demonstrates that major nonmonotonic systems are closely related. Also, a striking feature of Approximation theory is the ease with which it can be applied in any context where

semantics emerge as fixpoints of operators. For example, in [Denecker *et al.*, 2001b], approximation theory was applied in the context of aggregate logic programs and a family of semantics Kripke-Kleene, well-founded and the stable-model semantics was obtained. Consequently, studies of approximation theory contribute to the understanding of nonmonotonic reasoning formalisms by eliminating inessential details specific to a particular logic, by simplifying arguments and by finding common principles underlying different nonmonotonic formalisms.

In this paper, we study a *splitting* property of operators and fixpoints. In particular, we investigate the circumstances in which an operator defined on a product lattice can be split up in operators over the component lattices, and show how its various fixpoints can be obtained by composing the appropriate fixpoints of the component operators. In the context of a specific nonmonotonic logic, our theorem property translates into a compositionality theorem. In particular, we will show that our result generalises and abstracts some well-known results about splitting logic programs and answer set programs. Such properties have been investigated in the context of logic programming and answer set programming. For example, in [Lifschitz and Turner, 1994], it was shown that the answer sets of an answer set program satisfying an appropriate stratification condition can be be computed by splitting the program, and composing the answer sets of its components. We will show that our results generalise the theorem by Lifschitz and Turner, in two ways:

- the splitting theorem holds not only for stable models but also well-founded and Kripke-Kleene models, and
- the theorem strips off and makes abstraction of all details specific to answer set programming and thus we obtain a theorem that has its application in other nonmonotonic logics such as aggregate logic programs, autoepistemic and default logic.

The paper is structured as follows. The next two sections give a brief introduction to lattice theory and Approximation theory. Section 4 presents the splitting theorem and some consequences. Section 5 describes the application of the splitting theorem in the context of logic programming and answer set programming and discusses the relationship to the splitting theorem of [Lifschitz and Turner, 1994].

2 LATTICES

A *lattice* is a partially ordered set $\langle L, \leq \rangle$ such that every two element set $\{x, y\} \subseteq L$ has a *least upper bound*, $lub(x, y)$, and a *greatest lower bound*, $glb(x, y)$. A lattice $\langle L, \leq \rangle$ is *complete* if every subset of L has both least upper and greatest lower bounds. Consequently, a complete lattice has a least element ($\bot$) and a greatest element ($\top$).

An *interval* $[x, y]$ in L is the set $\{z \in L | x \leq z \leq y\}$.

An *operator* on a lattice $\langle L, \leq \rangle$ is any function from L to L. An operator O on L is *monotone* if for every pair of elements $x, y \in L$,

$$x \leq y \text{ implies } O(x) \leq O(y).$$

Similarly, an operator O on L is *antimonotone* if for every pair x, y of elements from L,

$$x \leq y \text{ implies } O(y) \leq O(x).$$

Tarski and Knaster [Tarski, 1955] stated *the* basic theorem in the study of fixpoints on operators.

Theorem 1 *Let O be a monotone operator on a complete lattice $\langle L, \leq \rangle$. Then, O has a fixpoint and the set of all fixpoints of O is a complete lattice.*

We denote the least and the greatest fixpoints of the operator O by $lfp(O)$ and $gfp(O)$, respectively.

It is often necessary or more practical not to look at the lattice elements themselves, but to approximate them. We say that an element $z \in L$ is approximated by a pair $(x, y) \in L^2$ if $z \in [x, y]$. Approximations of the form (x, x) are especially interesting. They give an exact description of the element they approximate and so, we refer to them as *exact*. There is a straightforward one-to-one correspondence between L and the set of exact elements of L^2.

The set L^2 can be endowed with two natural orderings. The first of them is a generalization of an ordering $\leq$ from L. We will refer to it as the *lattice* ordering and use the same symbol $\leq$ to denote it. Formally, it is defined by

$$(x, y) \leq (x_1, y_1) \text{ if } x \leq x_1 \text{ and } y \leq y_1.$$

The second ordering, called the *information* ordering, captures the intuition of increased precision of the approximation. This ordering, denoted $\leq_i$, is defined by

$$(x, y) \leq_i (x_1, y_1) \text{ if } x \leq x_1 \text{ and } y_1 \leq y.$$

It is easy to see that L^2 with each of these two orderings induces a complete lattice.

Let us consider an operator A on L^2. Let us denote by A^1 and A^2 the functions from L^2 to L such that

$$A(x,y) = (A^1(x,y), A^2(x,y)).$$

We say that A is *symmetric* if $A^1(x,y) = A^2(y,x)$. Clearly, if an operator $A : L^2 \to L^2$ is symmetric then for every $x \in L$, $A^1(x,x) = A^2(x,x)$.

Finally we define the notion of a direct product on lattices.

Definition 2 *The direct product of two lattices* $\langle L_1, \leq_1 \rangle$ *and* $\langle L_2, \leq_2 \rangle$ *is a structure* $\langle L_1 \otimes L_2, \leq \rangle$, *where* $L_1 \otimes L_2 = \{(a_1,a_2) | a_1 \in L_1, a_2 \in L_2\}$ *and the order* $\leq$ *is defined as* $\forall (a_1,a_2),(b_1,b_2) \in L_1 \otimes L_2 : (a_1,a_2) \leq (b_1,b_2) \Leftrightarrow a_1 \leq b_1 \wedge a_2 \leq b_2$.

Proposition 3 *If* $\langle L_1, \leq_1 \rangle$ *and* $\langle L_2, \leq_2 \rangle$ *are complete lattices, then so is* $\langle L_1 \otimes L_2, \leq \rangle$.

3 APPROXIMATION THEORY

Approximation theory is a methodology to study fixpoints of nonmonotone operators. *Approximating* a nonmonotone operator with a monotone allows the existing fixpoint theory [Fitting, 1999] to be applied in order to derive information on fixpoints of the original operator.

A pair of elements $(u,v) \in L^2$ *approximates* an element $x \in L$ if $u \leq x \leq v$.

Definition 4 *An operator* $A : L^2 \to L^2$ extends *an operator* $O : L \to L$ *if for every* $x \in L$, $A^1(x,x) \leq O(x) \leq A^2(x,x)$.

It is easy to see that if an operator $A : L^2 \to L^2$ extends $O : L \to L$, $A(x,x)$ approximates $O(x)$ for each $x \in L$.

Definition 5 *An operator* $A : L^2 \to L^2$ *is an* approximation *for an operator* $O : L \to L$ *if* A *extends* O *and is* $\leq_i$*-monotone. An operator* $A : L^2 \to L^2$ *is approximating if there exists an operator* O *on* L *such that* A *approximates* O.

Approximations are a usefull tool in the study of fixpoints of operators since each fixpoint of an operator O is approximated by the $\leq_i$-least fixpoint of each of his approximations.

Corollary 6 *Let* $A : L^2 \to L^2$ *be an approximation for an operator* $O : L \to L$. *Then,* A *has a* $\leq_i$*-least fixpoint. This fixpoint approximates every fixpoint of* O.

Thus, the $\leq_i$-least fixpoint is an important concept. It describes the common information of each fixpoint of O. We call the $\leq_i$-least fixpoint of an operator A the *Kripke-Kleene fixpoint* of A (since we can specialise the notion of a $\leq_i$-least fixpoint to Kripke-Kleene semantics in the context of logic programming).

Definition 7 *Given an approximation* A *on* L^2.

Define the lower stable operator $C_A^l : L \to L$ *such that* $C_A^l(y) = lfp(A^1(.,y))$.

Define the upper stable operator $C_A^u : L \to L$ *such that* $C_A^u(x) = lfp(A^2(x,.))$.

Since both $A^1(.,y)$ and $A^2(x,.)$ (for fixed x and y) are monotone, the definitions of $C_A^l(y)$ and $C_A^u(x)$ are sound.

Definition 8 *Define the partial stable operator* $\mathcal{C}_A$ *as*

$$\mathcal{C}_A(x,y) = \left(C_A^l(y), C_A^u(x)\right).$$

Proposition 9 *Let* L *be a complete lattice,* $A : L^2 \to L^2$ *an approximation. Then,* $\mathcal{C}_A$ *is is a* $\leq_i$*-monotone and* $\leq$*-antimonotone operator on* L^2.

Theorem 10 *Let* L *be a complete lattice and* $A : L^2 \to L^2$ *be an approximation. Every fixpoint of the stable operator* $\mathcal{C}_A$ *is a* $\leq$*-minimal fixpoint of* A.

This theorem shows that for an approximator A all fixpoints of $\mathcal{C}_A$ are also fixpoints of A. We will call these *stable fixpoints* of A.

Proposition 9 shows us that $\mathcal{C}_A$ has a $\leq_i$-least fixpoint. We call this the *well-founded* fixpoint of A (β_A).

4 SPLITTING APPROXIMATIONS

In some cases it is possible to derive fixpoints of an operator by looking on the behavior of that operator on certain subsets of its domain. In particular it will be possible under certain conditions to split a lattice L into two components and compose a fixpoint of an approximation $A : L^2 \to L^2$ from the fixpoints of derived operators on both component lattices.

We first state the conditions and show later on that it is possible to calculate fixpoints, the Kripke Kleene fixpoint, stable fixpoints and the well-founded fixpoints out of the approximations on the components.

4.1 DEFINITIONS

Definition 11 *Let* $L = L_b \otimes L_t$ *be a lattice,* $A : L^2 \to L^2$ *an approximation. We say that* A *is* stratified *over*

(L_b, L_t) *if* $\forall u_b, v_b \in L_b : \exists! x_b, y_b \in L_b : \forall u_t, v_t \in L_t : \exists! x_t, y_t \in L_t :$

$$A\big((u_b, u_t), (v_b, v_t)\big) = \big((x_b, x_t), (y_b, y_t)\big).$$

If an approximator is stratified, it will be possible to define two operators on both strata (L_b and L_t). We will even see that these operators are approximations themselves.

Let $L = L_b \otimes L_t$ be a lattice, $A : L^2 \to L^2$ an approximation that is stratified over (L_b, L_t). Since x_b and y_b are independent of the values u_t and v_t, we can define an operator A_b on L_b^2 and for each pair $(u_b, v_b) \in L_b^2$ an operator $A_{(u_b,v_b)}$ on L_t^2, such that

$$A^1\big((u_b, u_t), (v_b, v_t)\big) = \big(A_b^1(u_b, v_b), A^1_{(u_b,v_b)}(u_t, v_t)\big)$$

and

$$A^2\big((u_b, u_t), (v_b, v_t)\big) = \big(A_b^2(u_b, v_b), A^2_{(u_b,v_b)}(u_t, v_t)\big).$$

Theorem 12 *A_b and $A_{(u_b,v_b)}$ $(u_b, v_b \in L_b)$ are approximations.*

From now on we will call A_b the bottom approximation and $A_{(u_b,v_b)}$ a top approximation of A under stratification over (L_b, L_t).

4.2 FIXPOINTS

Theorem 13 *Let $L = L_b \otimes L_t$ be a lattice, $A : L^2 \to L^2$ an approximation that is stratified over (L_b, L_t). Let A_b and $A_{(u_b,v_b)}$ the bottom, resp. top approximations of A under stratification with (L_b, L_t), then*

$\forall u_b, v_b \in L_b, u_t, v_t \in L_t :$

$\big((u_b, u_t), (v_b, v_t)\big)$ *is a fixpoint of* $A \Leftrightarrow$

(u_b, v_b) *is a fixpoint of* A_b *and*

(u_t, v_t) *is a fixpoint of* $A_{(u_b,v_b)}$.

This theorem allows us to construct all fixpoints of an approximation looking only at its parts. It is an obvious consequence that the same correlation exists between the exact fixpoints of A and those of A_b and $A_{(u_b,v_b)}$.

The next theorem shows that the same equivalence holds for Kripke-Kleene fixpoints.

Theorem 14 *Let $L = L_b \otimes L_t$ be a lattice, $A : L^2 \to L^2$ an approximation that is stratified over (L_b, L_t). Let A_b and $A_{(u_b,v_b)}$ the bottom, resp. top approximations of A under stratification with (L_b, L_t), then*

$\forall u_b, v_b \in L_b, u_t, v_t \in L_t :$

$\big((u_b, u_t), (v_b, v_t)\big)$ *is the KK-fixpoint of* $A \Leftrightarrow$

(u_b, v_b) *is the KK-fixpoint of* A_b *and*

(u_t, v_t) *is the KK-fixpoint of* $A_{(u_b,v_b)}$.

Proof:

Let $\big((u_b, u_t), (v_b, v_t)\big)$ be the KK-fixpoint of A, clearly, (u_b, v_b) is a fixpoint of A_b. Suppose it is not the least. We then can find a fixpoint $(u'_b, v'_b) \not\geq_i (u_b, v_b)$ of A_b. However since there exists a fixpoint (u'_t, v'_t) of $A_{(u'_b,v'_b)}$, we have $\big((u'_b, u'_t), (v'_b, v'_t)\big)$ is a fixpoint of A. Since $\big((u_b, u_t), (v_b, v_t)\big) \not\leq_i \big((u'_b, u'_t), (v'_b, v'_t)\big)$, we get a contradiction.

If there exists a $(u'_t, v't) \not\geq_i (u_t, v_t)$ fixpoint of $A_{(u_b,v_b)}$, we see that $\big((u_b, u'_t), (v_b, v'_t)\big) \not\geq_i \big((u_b, u_t), (v_b, v_t)\big)$ is a fixpoint of A.

Finally, if (u_b, v_b) is a KK-fixpoint of A_b and (u_t, v_t) a KK-fixpoint of $A_{(u_b,v_b)}$, we get $\big((u_b, u_t), (v_b, v_t)\big)$, fixpoint of A. A fixpoint of A that is not $\geq_i \big((u_b, u_t), (v_b, v_t)\big)$, would give rise to a contradiction.

□

4.3 STABLE FIXPOINTS

We will now prove a one-to-one relation between the stable fixpoints of an approximation and the stable fixpoints of its parts, thus again enabling the possibility to derive the stable fixpoints of an approximation by looking only at its parts.

First we need the following lemma.

Lemma 15 *Let $L = L_b \otimes L_t$ be a lattice, $A : L^2 \to L^2$ an approximation that is stratified over (L_b, L_t). Let A_b and $A_{(u_b,v_b)}$ the bottom, resp. top approximations of A under stratification with (L_b, L_t), then*

$lfp(A^1(., (y_b, y_t))) = (u_b, u_t)$ *with*

$$u_b = lfp(A_b^1(., y_b)), u_t = lfp(A^1_{(u_b,y_b)}(., y_t)).$$

$lfp(A^2((x_b, x_t), .)) = (v_b, v_t)$ *with*

$$v_b = lfp(A_b^2(x_b, .)), v_t = lfp(A^2_{(x_b,v_b)}(x_t, .))$$

The proof of this lemma is analogous to the proof of theorem 14.

Corollary 16 $C_A(x, y) = \big(C^l_A(y), C^u_A(x)\big)$ *with*

$$C^l_A(y) = \big(C^l_{A_b}(y_b), C^l_{A_{(C^l_{A_b}(y_b), y_b)}}(y_t)\big)$$

$$C^u_A(x) = \big(C^u_{A_b}(x_b), C^u_{A_{(x_b, C^u_{A_b}(x_b))}}(x_t)\big)$$

Theorem 17 *Let $L = L_b \otimes L_t$ be a lattice, $A : L^2 \to L^2$ an approximation that can be stratified over (L_b, L_t). Let A_b and $A_{(u_b,v_b)}$ the bottom, resp. top approximations of A under stratification with (L_b, L_t), then*

$\forall u_b, v_b \in L_b, u_t, v_t \in L_t$:

$((u_b, u_t), (v_b, v_t))$ *is an (exact) fixpoint of* $C_A \Leftrightarrow$

(u_b, v_b) *is an (exact) fixpoint of* C_{A_b} *and*

(u_t, v_t) *is an (exact) fixpoint of* $C_{A_{(u_b,v_b)}}$.

It is easy to prove that we have a similar equivalence between the well-founded fixpoint of A and those of A_b and $A_{(u_b,u_v)}$.

5 SPLITTING APPLIED ON LOGIC PROGRAMS

The possibility to split a logic program and have an equivalence between the answer sets of a disjunctive logic program P and those of derived programs on parts of the alphabet of P was already proven by Lifschitz and Turner [Lifschitz and Turner, 1994]. We will apply our method of stratification on the four-valued immediate consequence operator and show the same results for normal logic programs. As a consequence of stratification we will also get an equivalence between exact fixpoints, stable fixpoints and the well-founded models of the operator on $\mathcal{I}^4_\tau$ and those on $\mathcal{I}^4_{\tau_b}$ and $\mathcal{I}^4_{\tau_t}$, again only on normal logic programs.

We will thus apply the theory on stratification of approximations to the special case of $\mathcal{T}_P$, the four-valued immediate consequence operator of a logic program P.

Let P be a propositional logic program defined on the alphabet τ. The lattice of two-, respectively four-valued τ-interpretations are denoted respectively $\mathcal{I}^2_\tau$, respectively $\mathcal{I}^4_\tau$. There are different equivalent ways of representing interpretations. Two-valued interpretations are often represented as sets of literals or functions of τ to $\mathcal{TWO} = \{\mathbf{t}, \mathbf{f}\}$. Four-valued interpretations are often defined as functions from τ to $\mathcal{FOUR} = \{\mathbf{t}, \mathbf{f}, \mathbf{i}, \mathbf{u}\}$. It is well-known that $\mathcal{FOUR}$ is the bilattice of $\mathcal{TWO}$, by identifying $\mathbf{t} = (\mathbf{t}, \mathbf{t}), \mathbf{f} = (\mathbf{f}, \mathbf{f}), \mathbf{i} = (\mathbf{t}, \mathbf{f}), \mathbf{u} = (\mathbf{f}, \mathbf{t})$. Consequently, elements of $\mathcal{I}^4_\tau$ can be defined alternatively as pairs of two-valued interpretations and $\mathcal{I}^4_\tau$ will be defined below as the bilattice of $\mathcal{I}^2_\tau$.

A *logic program* is a set of rules, where a *rule* r consists of three parts: an atom $head(r)$, a set of positive subgoals $pos(r)$ and a set of negated subgoals $neg(r)$; $At(r) = \{head(r)\} \cup pos(r) \cup neg(r)$. A rule with $head(r) = a$, positive subgoals $L_1, \ldots, L_i$, negative subgoals $L_{i+1}, \ldots, L_n$ is written as

$$a \leftarrow L_1, \ldots, L_i, not\ L_{i+1}, \ldots, not\ L_m.$$

The immediate consequence operator, $T_P : \mathcal{I}^2_\tau \to \mathcal{I}^2_\tau$, of a program P is defined such that $T_P(I) = J$ where $J(p) = \mathbf{t}$ iff some rule $r \in P$ can be found such that $p = head(r), pos(r) \subseteq I$ and $neg(r) \cap I = \phi$. The four-valued immediate consequence operator $\mathcal{T}_P : \mathcal{I}^4_\tau \to \mathcal{I}^4_\tau$, of a program P is defined such that $T_P((I, J)) = (I', J')$ where $I'(p) = \mathbf{t}$ iff some rule $r \in P$ can be found such that $p = head(r), pos(r) \subseteq I$ and $neg(r) \cap J = \phi$, and, $J'(p) = \mathbf{t}$ iff some rule $r \in P$ can be found such that $p = head(r), pos(r) \subseteq J$ and $neg(r) \cap I = \phi$.

In [Denecker *et al.*, 2000a; Denecker *et al.*, 2001a], it was shown that $\mathcal{T}_P$ is an approximation of T_P.

It is possible to define a condition on the structure of a logic program P to make sure its four-valued immediate consequence operator $\mathcal{T}_P$ can be stratified.

We call a *splitting set* for a program P a set $\tau_b \subset \tau$ of atoms such that, for every rule $r \in P$, if $head(r) \in \tau_b$ then $At(r) \subset \tau_b$ as in [Lifschitz and Turner, 1994]. The set of rules $r \in P$ such that $At(r) \subset \tau_b$ is called the bottom of P relative to the splitting set τ_b and is denoted as P_b.

We define the top set of an alphabet τ relative to a splitting set τ_b as $\tau_t = \tau \backslash \tau_b$.

It is easy to see that $\mathcal{I}^2_\tau = \mathcal{I}^2_{\tau_b} \otimes \mathcal{I}^2_{\tau_t}$.

Theorem 18 *Let P be a logic program and τ_b a splitting set for P, then $\mathcal{T}_P$ can be stratified over $(\mathcal{I}^2_{\tau_b}, \mathcal{I}^2_{\tau_t})$.*

It can be proven that the four valued consequence operators of the logic programs on τ_b and τ_t as defined by Lifschitz and Turner coincide with our bottom and top operators.

6 CONCLUSION

In this paper we showed that it is possible to split an approximation on a product lattice into new approximations on the components of this lattice. We have also shown that this is a generalisation of the concept of splitting a logic program as formulated by Lifschitz and Turner [Lifschitz and Turner, 1994].

Since approximation theory is also a generalisation of a wide variaty of nonmonotonic logics such as default logic, autoepistemic logic and aggregate logic programs, we have implicitly proven that splitting is also applicable in a number of other nonmonotonic formalisms.

Futher research will extend the concept of stratification of approximations to any number of strata. Also the consequences of applying our method of stratification in default and autoepistemic logic will have to be thoroughly investigated, as well as the connection with existing splitting/stratification theories (as in [Turner, 1996]).

References

[Balbes and Dwinger, 1974] Raymond Balbes and Philip Dwinger. *Distributive Lattices.* University of Missouri Press, 1974.

[Clark, 1978] Keith L. Clark. Negation as failure. In H. Gallaire and J. Minker, editors, *Logic and data bases*, pages 293–322. Plenum Press, New York-London, 1978.

[Denecker *et al.*, 2000a] Marc Denecker, Victor Marek, and Miroslaw Truszczynski. Approximating operators, stable operators, well-founded fixpoints and applications in nonmonotonic reasoning. In J. Minker, editor, *Logic-based Artificial Intelligence*, chapter 6, pages 127–144. Kluwer Academic Publishers, Boston, 2000.

[Denecker *et al.*, 2000b] Marc Denecker, Victor Marek, and Miroslaw Truszczynski. Uniform semantic treatment of default and autoepistemic logics. In F. Giunchiglia A.G. Cohn and B. Selman, editors, *7'th Int. Conference on Principles of Knowledge Representation and Reasoning (KR2000)*, pages 74–84, Breckenridge, April 2000. Morgan Kaufman.

[Denecker *et al.*, 2001a] Marc Denecker, Maurice Bruynooghe, and Victor Marek. Logic programming revisited: logic programs as inductive definitions. *ACM Transactions on Computational Logic*, 2(4):623–654, October 2001.

[Denecker *et al.*, 2001b] Marc Denecker, Nikolay Pelov, and Maurice Bruynooghe. Ultimate well-founded and stable semantics for logic programs with aggregates. In *International Conference on Logic Programming*, pages 212–226, 2001.

[Denecker *et al.*, 2002] Marc Denecker, Victor Marek, and Miroslaw Truszczynski. Ultimate approximations of operators in commonsense reasoning. In *Proceedings of KR2002. To appear*, Toulouse, France, April 2002.

[Denecker, 2000] Marc Denecker. Extending classical logic with inductive definitions. In C. Baral and M. Truszczynski, editors, *8th Intl. Workshop on Non-Monotonic Reasoning (NMR2000)*, pages 1–15, April 2000.

[Fitting, 1985] Melvin Fitting. A kripke-kleene semantics for logic programs. *Journal of Logic Programming*, 2(4):295–312, 1985.

[Fitting, 1999] Melvin Fitting. Fixpoint semantics for logic programming – a survey. to appear, 1999.

[Gelfond and Lifschitz, 1988] Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. In Robert A. Kowalski and Kenneth Bowen, editors, *Proceedings of the Fifth International Conference on Logic Programming*, pages 1070–1080, Cambridge, Massachusetts, 1988. The MIT Press.

[Ginsberg, 1988] Matt Ginsberg. Multivalued logics: a uniform approach to reasoning in artificial intelligence. *Computational Intelligence*, 4:265–316, 1988.

[Kunen, 1987] Kenneth Kunen. Negation in logic programming. *Journal of Logic Programming*, 4(4):289–308, 1987.

[Lifschitz and Turner, 1994] Vladimir Lifschitz and Hudson Turner. Splitting a logic program. In *International Conference on Logic Programming*, pages 23–37, 1994.

[Tarski, 1955] Alfred Tarski. Lattice-theoretic fixpoint theorem and its applications. *Pacific Journal of Mathematics*, 5:285–309, 1955.

[Turner, 1996] Hudson Turner. Splitting a default theory. In Howard Shrobe and Ted Senator, editors, *Proceedings of the Thirteenth National Conference on Artificial Intelligence and the Eighth Innovative Applications of Artificial Intelligence Conference*, pages 645–651, Menlo Park, California, 1996. AAAI Press.

[van Gelder *et al.*, 1991] Allen van Gelder, Kenneth Ross, and John S. Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM*, 38(3):620–650, 1991.

A Polynomial Translation of Logic Programs with Nested Expressions into Disjunctive Logic Programs: Preliminary Report*

David Pearce
European Commission,
DG Information Society – F1
BU33 3/58, Rue de le Loi 200,
B-1049 Brussels
David.Pearce@cec.eu.int

Vladimir Sarsakov, Torsten Schaub†
Institut für Informatik, Universität Potsdam,
Postfach 90 03 27, D–14439 Potsdam, Germany
sarsakov@rz.uni-potsdam.de,
torsten@cs.uni-potsdam.de

Hans Tompits, Stefan Woltran
Institut für Informationssysteme 184/3,
Technische Universität Wien,
Favoritenstraße 9–11, A–1040 Vienna, Austria
[tompits,stefan]@kr.tuwien.ac.at

Abstract

Nested logic programs have recently been introduced in order to allow for arbitrarily nested formulas in the heads and the bodies of logic program rules under the answer sets semantics. Nested expressions can be formed using conjunction, disjunction, as well as the negation as failure operator in an unrestricted fashion. This provides a very flexible and compact framework for knowledge representation and reasoning. Previous results show that nested logic programs can be transformed into standard (unnested) disjunctive logic programs in an elementary way, applying the negation as failure operator to body literals only. This is of great practical relevance since it allows us to evaluate nested logic programs by means of off-the-shelf disjunctive logic programming systems, like DLV. However, it turns out that this straightforward transformation results in an exponential blow-up in the worst-case, despite the fact that complexity results indicate that there is a polynomial translation among both formalisms. In this paper, we take up this challenge and provide a polynomial translation of logic programs with nested expressions into disjunctive logic programs. Moreover, we show that this translation is modular and (strongly) faithful. We have implemented both the straightforward as well as our advanced transformation; the resulting compiler serves as a front-end to DLV and is publicly available on the Web.

1 Introduction

Lifschitz, Tang, and Turner [23] recently extended the answer set semantics [12] to a class of logic programs in which arbitrarily nested formulas, formed from literals using negation as failure, conjunction, and disjunction, constitute the heads and bodies of rules. These so-called *nested logic programs* generalise the well-known classes of *normal*, *generalised*, *extended*, and *disjunctive logic programs*, respectively. Despite their syntactically much more restricted format, the latter classes are well recognised as important tools for knowledge representation and reasoning. This is reflected by the fact that several practicably relevant applications have been developed recently using these types of programs (cf., e.g., [21, 3, 11, 16]), which in turn is largely fostered by the availability of efficient solvers for the answer set semantics, most notably DLV [8, 9] and Smodels [25].

In this paper, we are interested in utilising these highly performant solvers for interpreting nested logic programs. We address this problem by providing a translation of nested

* This work was partially supported by the Austrian Science Fund (FWF) under grants P15068-INF and N Z29-INF. The authors would like to thank Agata Ciabattoni for pointing out some relevant references.

† Affiliated with the School of Computing Science at Simon Fraser University, Burnaby, Canada.

logic programs into disjunctive logic programs. In contrast to previous work, our translation is guaranteed to be polynomial in time and space, as suggested by related complexity results [30]. More specifically, we provide a translation, σ, from nested logic programs into disjunctive logic programs possessing the following properties:

- σ maps nested logic programs over an alphabet $\mathcal{A}_1$ into disjunctive logic programs over an alphabet $\mathcal{A}_2$, where $\mathcal{A}_1 \subseteq \mathcal{A}_2$;
- the size of $\sigma(\Pi)$ is polynomial in the size of Π;
- σ is *faithful*, i.e., for each program Π over alphabet $\mathcal{A}_1$, there is a one-to-one correspondence between the answer sets of Π and sets of form $I \cap \mathcal{A}_1$, where I is an answer set of $\sigma(\Pi)$; and
- σ is *modular*, i.e., $\sigma(\Pi \cup \Pi') = \sigma(\Pi) \cup \sigma(\Pi')$, for each program Π, Π'.

Moreover, we have implemented translation σ, serving as a front-end for the logic programming system DLV.

The construction of σ relies on the introduction of new *labels*, abbreviating subformula occurrences. This technique is derived from *structure-preserving normal form translations* [34, 31], frequently employed in the context of automated deduction (cf. [1] for an overview). We use here a method adapted from a structure-preserving translation for intuitionistic logic as described in [24].

Regarding the faithfulness of σ, we actually provide a somewhat stronger condition, referred to as *strong faithfulness*, expressing that, for any programs Π and Π' over alphabet $\mathcal{A}_1$, there is a one-to-one correspondence between the answer sets of $\Pi \cup \Pi'$ and sets of form $I \cap \mathcal{A}_1$, where I is an answer set of $\sigma(\Pi) \cup \Pi'$. This condition means that we can add to a given program Π *any nested program* Π' and still recover the answer sets of the combined program $\Pi \cup \Pi'$ from $\sigma(\Pi) \cup \Pi'$; in particular, for any nested logic program Π, we may choose to translate, in a semantics-preserving way, only an arbitrary *program part* $\Pi_0 \subseteq \Pi$ and leave the remaining part $\Pi \setminus \Pi_0$ unchanged. For instance, if Π_0 is already a disjunctive logic program, we do not need to translate it again into another (equivalent) disjunctive logic program. Strong faithfulness is closely related to the concept of *strong equivalence* [22] (see below).

In order to have a sufficiently general setting for our purposes, we base our investigation on *equilibrium logic* [26], a generalisation of the answer set semantics for nested logic programs. Equilibrium logic is a form of minimal-model reasoning in the *logic of here-and-there*, which is intermediate between classical logic and intuitionistic logic (the logic of here-and-there is also known as *Gödel's three-valued logic* in view of [14]). As shown in [26, 27, 22], logic programs can be viewed as a special class of formulas in the logic of here-and-there such that, for each program Π, the answer sets of Π are given by the equilibrium models of Π, where the latter Π is viewed as a set of formulas in the logic of here-and-there.

The problem of implementing nested logic programs has already been addressed in [30], where (linear-time constructible) encodings of the basic reasoning tasks associated with this language into quantified Boolean formulas are described. These encodings provide a straightforward implementation for nested logic programs by appeal to off-the-shelf solvers for quantified Boolean formulas (like, e.g., the systems proposed in [4, 10, 13, 19, 20, 32]). Besides the encodings into quantified Boolean formulas, a further result of [30] is that nested logic programs possess the same worst-case complexity as disjunctive logic programs, i.e., the main reasoning tasks associated with nested logic programs lie at the second level of the polynomial hierarchy. From this result it follows that nested logic programs can in turn be efficiently reduced to disjunctive logic programs. Hence, given such a reduction, solvers for the latter kinds of programs, like, e.g., DLV or `Smodels`, can be used to compute the answer sets of nested logic programs. The main goal of this paper is to construct a reduction of this type.

Although results by Lifschitz, Tang, and Turner [23] (together with transformation rules given in [18]) provide a method to translate nested logic programs into disjunctive ones, that approach suffers from the drawback of an exponential blow-up of the resulting disjunctive logic programs in the worst case. This is due to the fact that the "language-preserving" nature of that translation relies on distributivity laws yielding an exponential increase of program size whenever the given program contains rules whose heads are in disjunctive normal form or whose bodies are in conjunctive normal form, and the respective expressions are not simple disjunctions or conjunctions of literals. Our translation, on the other hand, is always polynomial in the size of its input program.

Finally, we mention that structure-preserving normal form translations in the logic of here-and-there are also studied, yet in much more general settings, by Baaz and Fermüller [2] as well as by Hähnle [15]; there, whole classes of finite-valued Gödel logics are investigated. Unfortunately, these normal form translations are not suitable for our purposes, because they do not enjoy the particular form of programs required here.

2 Preliminaries

We deal with propositional languages and use the logical symbols $\top$, $\bot$, $\neg$, $\vee$, $\wedge$, and $\rightarrow$ to construct formulas in

the standard way. We write $\mathcal{L}_\mathcal{A}$ to denote a language over an alphabet $\mathcal{A}$ of *propositional variables* or *atoms*. Formulas are denoted by Greek lower-case letters (possibly with subscripts). As usual, *literals* are formulas of form v or $\neg v$, where v is some variable or one of $\top, \bot$.

Besides the semantical concepts introduced below, we also make use of the semantics of classical propositional logic. By a (*classical*) *interpretation*, I, we understand a set of variables. Informally, a variable v is true under I iff $v \in I$. The truth value of a formula ϕ under interpretation I, in the sense of classical propositional logic, is determined in the usual way.

2.1 Logic Programs

The central objects of our investigation are logic programs with nested expressions, introduced by Lifschitz *et al.* [23]. These kinds of programs generalise normal logic programs by allowing bodies and heads of rules to contain arbitrary Boolean formulas. For reasons of simplicity, we deal here only with languages containing one kind of negation, however, corresponding to default negation. The extension to the general case where strong negation is also permitted is straightforward and proceeds in the usual way.

We start with some basic notation. A formula whose sentential connectives comprise only $\wedge$, $\vee$, or $\neg$ is called an *expression*. A rule, r, is an ordered pair of form

$$H(r) \leftarrow B(r),$$

where $B(r)$ and $H(r)$ are expressions. $B(r)$ is called the *body* of r and $H(r)$ is the *head* of r. We say that r is a *generalised disjunctive rule* if $B(r)$ is a conjunction of literals and $H(r)$ is a disjunction of literals; r is a *disjunctive rule* iff it is a generalised disjunctive rule containing no negated atom in its head; finally, if r is a rule containing no negation at all, then r is called *basic*. A *nested logic program*, or simply a *program*, Π, is a finite set of rules. Π is a *generalised disjunctive logic program* iff it contains only generalised disjunctive rules. Likewise, Π is a *disjunctive logic program* iff Π contains only disjunctive rules, and Π is *basic* iff each rule in Π is basic. We say that Π is a program *over* alphabet $\mathcal{A}$ iff all atoms occurring in Π are from $\mathcal{A}$. The set of all atoms occurring in program Π is denoted by $var(\Pi)$. We use $NLP_\mathcal{A}$ to denote the class of all nested logic programs over alphabet $\mathcal{A}$; furthermore, $DLP_\mathcal{A}$ stands for the subclass of $NLP_\mathcal{A}$ containing all disjunctive logic programs over $\mathcal{A}$; and $GDLP_\mathcal{A}$ is the class of all generalised disjunctive logic programs over $\mathcal{A}$. Further classes of programs are introduced in Section 4.

In what follows, we associate to each rule r a corresponding formula $\hat{r} = B(r) \rightarrow H(r)$ and, accordingly, to each program Π a corresponding set of formulas $\hat{\Pi} = \{\hat{r} \mid r \in \Pi\}$.

Let Π be a basic program over $\mathcal{A}$ and $I \subseteq \mathcal{A}$ a (classical) interpretation. We say that I is a *model* of Π iff it is a model of the associated set $\hat{\Pi}$ of formulas. Furthermore, given an (arbitrary) program Π over $\mathcal{A}$, the *reduct*, Π^I, of Π with respect to I is the basic program obtained from Π by replacing every occurrence of an expression $\neg\psi$ in Π which is not in the scope of any other negation by $\bot$ if ψ is true under I, and by $\top$ otherwise. I is an *answer set* (or *stable model*) of Π iff it is a minimal model (with respect to set inclusion) of the reduct Π^I. The collection of all answer sets of Π is denoted by $AS_\mathcal{A}(\Pi)$.

Two logic programs, Π_1 and Π_2, are *equivalent* iff they possess the same answer sets. Following Lifschitz *et al.* [22], we call Π_1 and Π_2 *strongly equivalent* iff, for every program Π, $\Pi_1 \cup \Pi$ and $\Pi_2 \cup \Pi$ are equivalent.

2.2 Equilibrium Logic

Equilibrium logic is an approach to nonmonotonic reasoning that generalises the answer set semantics for logic programs. We use this particular formalism because it offers a convenient logical language for dealing with logic programs under the answer set semantics. It is defined in terms of the logic of here-and-there, which is intermediate between classical logic and intuitionistic logic. Equilibrium logic was introduced in [26] and further investigated in [27]; proof theoretic studies of the logic can be found in [29, 28].

Generally speaking, the logic of here-and-there is an important tool for analysing various properties of logic programs. For instance, as shown in [22], the problem of checking whether two logic programs are strongly equivalent can be expressed in terms of the logic of here-and-there (cf. Proposition 2 below).

The semantics of the logic of here-and-there is defined by means of two worlds, H and T, called "here" and "there". It is assumed that there is a total order, $\leq$, defined between these worlds such that $\leq$ is reflexive and $H \leq T$. As in ordinary Kripke semantics for intuitionistic logic, we can imagine that in each world a set of atoms is verified and that, once verified "here", an atom remains verified "there".

Formally, by an *HT-interpretation*, $\mathcal{I}$, we understand an ordered pair $\langle I_H, I_T\rangle$ of sets of atoms such that $I_H \subseteq I_T$. We say that $\mathcal{I}$ is an HT-interpretation *over* $\mathcal{A}$ if $I_T \subseteq \mathcal{A}$. The set of all HT-interpretations over $\mathcal{A}$ is denoted by $INT_\mathcal{A}$. An HT-interpretation $\langle I_H, I_T\rangle$ is *total* if $I_H = I_T$.

The truth value, $\nu_\mathcal{I}(w, \phi)$, of a formula ϕ at a world $w \in \{H, T\}$ in an HT-interpretation $\mathcal{I} = \langle I_H, I_T\rangle$ is recursively defined as follows:

1. if $\phi = \top$, then $\nu_\mathcal{I}(w, \phi) = 1$;

2. if $\phi = \bot$, then $\nu_{\mathcal{I}}(w, \phi) = 0$;

3. if $\phi = v$ is an atom, then $\nu_{\mathcal{I}}(w, \phi) = 1$ if $v \in I_w$, otherwise $\nu_{\mathcal{I}}(w, \phi) = 0$;

4. if $\phi = \neg\psi$, then $\nu_{\mathcal{I}}(w, \phi) = 1$ if, for every world u with $w \leq u$, $\nu_{\mathcal{I}}(u, \psi) = 0$, otherwise $\nu_{\mathcal{I}}(w, \phi) = 0$;

5. if $\phi = (\phi_1 \wedge \phi_2)$, then $\nu_{\mathcal{I}}(w, \phi) = 1$ if $\nu_{\mathcal{I}}(w, \phi_1) = 1$ and $\nu_{\mathcal{I}}(w, \phi_2) = 1$, otherwise $\nu_{\mathcal{I}}(w, \phi) = 0$;

6. if $\phi = (\phi_1 \vee \phi_2)$, then $\nu_{\mathcal{I}}(w, \phi) = 1$ if $\nu_{\mathcal{I}}(w, \phi_1) = 1$ or $\nu_{\mathcal{I}}(w, \phi_2) = 1$, otherwise $\nu_{\mathcal{I}}(w, \phi) = 0$;

7. if $\phi = (\phi_1 \rightarrow \phi_2)$, then $\nu_{\mathcal{I}}(w, \phi) = 1$ if for every world u with $w \leq u$, $\nu_{\mathcal{I}}(u, \phi_1) = 0$ or $\nu_{\mathcal{I}}(u, \phi_2) = 1$, otherwise $\nu_{\mathcal{I}}(w, \phi) = 0$.

We say that ϕ is *true under* $\mathcal{I}$ *in* w iff $\nu_{\mathcal{I}}(w, \phi) = 1$, otherwise ϕ is *false under* $\mathcal{I}$ *in* w. An HT-interpretation $\mathcal{I} = \langle I_H, I_T \rangle$ *satisfies* ϕ, or $\mathcal{I}$ is an *HT-model* of ϕ, iff $\nu_{\mathcal{I}}(H, \phi) = 1$. If ϕ is true under any HT-interpretation, then ϕ is *valid in the logic of here-and-there*, or simply *HT-valid*.

Let S be a set of formulas. An HT-interpretation $\mathcal{I}$ is an HT-model of S iff $\mathcal{I}$ is an HT-model of each element of S. We say that $\mathcal{I}$ is an HT-model of a *program* Π iff $\mathcal{I}$ is an HT-model of $\hat{\Pi} = \{B(r) \rightarrow H(r) \mid r \in \Pi\}$.

Two sets of formulas are *equivalent in the logic of here-and-there*, or *HT-equivalent*, iff they possess the same HT-models. Two formulas, ϕ and ψ, are HT-equivalent iff the sets $\{\phi\}$ and $\{\psi\}$ are HT-equivalent.

It is easily seen that any HT-valid formula is valid in classical logic, but the converse does not always hold. For instance, $p \vee \neg p$ and $\neg\neg p \rightarrow p$ are valid in classical logic but not in the logic of here-and-there as the pair $\langle \emptyset, \{p\} \rangle$ is not an HT-model for either of these formulas.

Equilibrium logic can be seen as a particular type of reasoning with minimal HT-models. Formally, an *equilibrium model* of a formula ϕ is a total HT-interpretation $\langle I, I \rangle$ such that (i) $\langle I, I \rangle$ is an HT-model of ϕ, and (ii) for every proper subset J of I, $\langle J, I \rangle$ is not an HT-model of ϕ.

The following result establishes the close connection between equilibrium models and answer sets, showing that answer sets are actually a special case of equilibrium models:

Proposition 1 ([26, 22]) *For any program* Π, I *is an answer set of* Π *iff* $\langle I, I \rangle$ *is an equilibrium model of* $\hat{\Pi}$.

Moreover, HT-equivalence was shown to capture the notion of strong equivalence between logic programs:

Proposition 2 ([22]) *Let* Π_1 *and* Π_2 *be programs, and let* $\hat{\Pi}_i = \{B(r) \rightarrow H(r) \mid r \in \Pi_i\}$, *for* $i = 1, 2$. *Then,* Π_1 *and* Π_2 *are strongly equivalent iff* $\hat{\Pi}_1$ *and* $\hat{\Pi}_2$ *are equivalent in the logic of here-and-there.*

Recently, de Jongh and Hendriks [5] have extended Proposition 2 by showing that for nested programs strong equivalence is characterised precisely by equivalence in all intermediate logics lying between here-and-there (upper bound) and the logic KC of weak excluded middle (lower bound) which is axiomatised by intuitionistic logic together with the schema $\neg\varphi \vee \neg\neg\varphi$.

We require the following additional concepts. By an *HT-literal*, l, we understand a formula of form v, $\neg v$, or $\neg\neg v$, where v is a propositional atom or one of $\top$, $\bot$. Furthermore, a formula is in *here-and-there negational normal form*, or *HT-NNF*, if it is made up of HT-literals, conjunctions and disjunctions. Likewise, we say that a *program* is in HT-NNF iff all heads and bodies of rules in the program are in HT-NNF.

Following [23], every expression ϕ can effectively be transformed into an expression ψ in HT-NNF possessing the same HT-models as ϕ. In fact, we have the following property:

Proposition 3 *Every expression* ϕ *is HT-equivalent to an expression* $\nu(\phi)$ *in HT-NNF, where* $\nu(\phi)$ *is constructible in polynomial time from* ϕ, *satisfying the following conditions, for each expression* φ, ψ:

1. $\nu(\varphi) = \varphi$, *if* φ *is an HT-literal;*
2. $\nu(\neg\neg\neg\varphi) = \nu(\neg\varphi)$;
3. $\nu(\varphi \circ \psi) = \nu(\varphi) \circ \nu(\psi)$, *for* $\circ \in \{\wedge, \vee\}$;
4. $\nu(\neg(\varphi \wedge \psi)) = \nu(\neg\varphi) \vee \nu(\neg\psi)$;
5. $\nu(\neg(\varphi \vee \psi)) = \nu(\neg\varphi) \wedge \nu(\neg\psi)$.

3 Faithful Translations

Next, we introduce the general requirements we impose on our desired translation from nested logic programs into disjunctive logic programs. The following definition is central:

Definition 1 *Let* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *be two alphabets such that* $\mathcal{A}_1 \subseteq \mathcal{A}_2$, *and, for* $i = 1, 2$, *let* $S_i \subseteq NLP_{\mathcal{A}_i}$ *be a class of nested logic programs closed under unions.*[1] *Then, a function* $\rho : S_1 \rightarrow S_2$ *is*

[1] A class S of sets is closed under unions providing $A, B \in S$ implies $A \cup B \in S$.

1. polynomial *iff, for all programs* $\Pi \in S_1$, *the time required to compute* $\rho(\Pi)$ *is polynomial in the size of* Π;

2. faithful *iff, for all programs* $\Pi \in S_1$,

$$AS_{\mathcal{A}_1}(\Pi) = \{I \cap \mathcal{A}_1 \mid I \in AS_{\mathcal{A}_2}(\rho(\Pi))\};$$

3. strongly faithful *iff, for all programs* $\Pi \in S_1$ *and all programs* $\Pi' \in NLP_{\mathcal{A}_1}$,

$$AS_{\mathcal{A}_1}(\Pi \cup \Pi') = \{I \cap \mathcal{A}_1 \mid I \in AS_{\mathcal{A}_2}(\rho(\Pi) \cup \Pi')\};$$

and

4. modular *iff, for all programs* $\Pi_1, \Pi_2 \in S_1$,

$$\rho(\Pi_1 \cup \Pi_2) = \rho(\Pi_1) \cup \rho(\Pi_2).$$

In view of the requirement that $\mathcal{A}_1 \subseteq \mathcal{A}_2$, the general functions considered here may introduce new atoms. Clearly, if the given function is polynomial, the number of newly introduced atoms is also polynomial. Faithfulness guarantees that we can recover the stable models of the input program from the translated program. Strong faithfulness, on the other hand, states that we can add to a given program Π *any nested logic program* Π' and still retain, up to the original language, the semantics of the combined program $\Pi \cup \Pi'$ from $\rho(\Pi) \cup \Pi'$. Finally, modularity enforces that we can translate programs rule by rule.

It is quite obvious that any strongly faithful function is also faithful. Furthermore, strong faithfulness of function ρ implies that, for a given program Π, we can translate *any program part* Π_0 of Π whilst leaving the remaining part $\Pi \setminus \Pi_0$ unchanged, and determine the semantics of Π from $\rho(\Pi_0) \cup (\Pi \setminus \Pi_0)$. As well, for any function of form $\rho : NLP_{\mathcal{A}} \to NLP_{\mathcal{A}}$, strong faithfulness of ρ is equivalent to the condition that Π and $\rho(\Pi)$ are strongly equivalent, for any $\Pi \in NLP_{\mathcal{A}}$. Hence, strong faithfulness generalises strong equivalence.

Following [17, 18], we say that a function ρ as in Definition 1 is *PFM*, or that ρ is a *PFM-function*, iff it is polynomial, faithful, and modular. Analogously, we call ρ *PSM*, or a *PSM-function*, iff it is polynomial, strongly faithful, and modular.

It is easy to see that the composition of two PFM-functions is again a PFM-function; and likewise for PSM-functions. Furthermore, since any PSM-function is also PFM, in the following we focus on PSM-functions. In fact, in the next section, we construct a function $\sigma : NLP_{\mathcal{A}_1} \to DLP_{\mathcal{A}_2}$ (where $\mathcal{A}_2$ is a suitable extension of $\mathcal{A}_1$) which is PSM.

Next, we discuss some sufficient conditions guaranteeing that certain classes of functions are strongly faithful. We start with the following concept.

Definition 2 *Let* $\rho : NLP_{\mathcal{A}_1} \to NLP_{\mathcal{A}_2}$ *be a function such that* $\mathcal{A}_1 \subseteq \mathcal{A}_2$, *and let* $INT_{\mathcal{A}_i}$ *be the class of all HT-interpretations over* $\mathcal{A}_i$ $(i = 1, 2)$.

Then, the function $\alpha_\rho : INT_{\mathcal{A}_1} \times NLP_{\mathcal{A}_1} \to INT_{\mathcal{A}_2}$ *is called a* ρ-associated HT-embedding *iff, for each HT-interpretation* $\mathcal{I} = \langle I_H, I_T \rangle$ *over* $\mathcal{A}_1$, *each* $\Pi \in NLP_{\mathcal{A}_1}$, *and each* $w \in \{H, T\}$, $J_w \cap \mathcal{A}_1 = I_w$ *and* $J_w \setminus \mathcal{A}_1 \subseteq var(\rho(\Pi))$, *where* $\alpha_\rho(\mathcal{I}, \Pi) = \langle J_H, J_T \rangle$.

Furthermore, for any $G \subseteq INT_{\mathcal{A}_1}$ *and any* $\Pi \in NLP_{\mathcal{A}_1}$, *we define* $\alpha_\rho(G, \Pi) = \{\alpha_\rho(\mathcal{I}, \Pi) \mid \mathcal{I} \in G\}$.

Intuitively, a ρ-associated HT-embedding transforms HT-interpretations over the input alphabet $\mathcal{A}_1$ of ρ into HT-interpretations over the output alphabet $\mathcal{A}_2$ of ρ such that the truth values of the atoms in $\mathcal{A}_1$ are retained. The following definition strengthens these kinds of mappings:

Definition 3 *Let* ρ *be as in Definition 2, and let* α_ρ *be a* ρ*-associated HT-embedding. We say that* α_ρ *is a* ρ-associated HT-homomorphism *if, for any* $\mathcal{I}, \mathcal{I}' \in INT_{\mathcal{A}_1}$ *and any* $\Pi \in NLP_{\mathcal{A}_1}$, *the following conditions hold:*

1. $\mathcal{I}$ *is an HT-model of* Π *iff* $\alpha_\rho(\mathcal{I}, \Pi)$ *is an HT-model of* $\rho(\Pi)$;

2. $\mathcal{I}$ *is total iff* $\alpha_\rho(\mathcal{I}, \Pi)$ *is total;*

3. *if* $\mathcal{I} = \langle I_H, I_T \rangle$ *and* $\mathcal{I}' = \langle I'_H, I'_T \rangle$ *are HT-models of* Π, *then* $I_H \subset I'_H$ *and* $I_T = I'_T$ *holds precisely if* $J_H \subset J'_H$ *and* $J_T = J'_T$, *for* $\alpha_\rho(\mathcal{I}, \Pi) = \langle J_H, J_T \rangle$ *and* $\alpha_\rho(\mathcal{I}', \Pi) = \langle J'_H, J'_T \rangle$; *and*

4. *an HT-interpretation* $\mathcal{J}$ *over* $var(\rho(\Pi))$ *is an HT-model of* $\rho(\Pi)$ *only if* $\mathcal{J} \in \alpha_\rho(INT_{\mathcal{A}_1}, \Pi)$.

Roughly speaking, ρ-associated HT-homomorphisms retain the relevant properties of HT-interpretations for being equilibrium models with respect to transformation ρ. More specifically, the first three conditions take semantical and set-theoretical properties into account, respectively, whilst the last one expresses a specific "closure condition". The inclusion of the latter requirement is explained by observation that the first three conditions alone are not sufficient to exclude the possibility that there may exist some equilibrium model $\mathcal{I}$ of Π such that $\alpha_\rho(\mathcal{I}, \Pi)$ is not an equilibrium model of $\rho(\Pi)$. The reason for this is that the set $\alpha_\rho(INT_{\mathcal{A}_1}, \Pi)$, comprising the images of all HT-interpretations over $\mathcal{A}_1$ under α_ρ with respect to program Π, does, in general, not cover *all* HT-interpretations over $var(\rho(\Pi))$. Hence, for a general ρ-associated HT-embedding $\alpha_\rho(\cdot, \cdot)$, there may exist some HT-model of $\rho(\Pi)$ which is not included in $\alpha_\rho(INT_{\mathcal{A}_1}, \Pi)$ preventing $\alpha_\rho(\mathcal{I}, \Pi)$ from being an equilibrium model of $\rho(\Pi)$ albeit $\mathcal{I}$ is an equilibrium model of Π. The addition of the last

condition in Definition 3, however, excludes this possibility, ensuring that all relevant HT-interpretations required for checking whether $\alpha_\rho(\mathcal{I}, \Pi)$ is an equilibrium model of $\rho(\Pi)$ are indeed considered. The following result can be shown:

Lemma 1 *For any function $\rho : NLP_{\mathcal{A}_1} \to NLP_{\mathcal{A}_2}$ with $\mathcal{A}_1 \subseteq \mathcal{A}_2$, if there is some ρ-associated HT-homomorphism, then ρ is faithful.*

From this, we obtain the following property:

Theorem 1 *Under the circumstances of Lemma 1, if ρ is modular and there is some ρ-associated HT-homomorphism, then ρ is strongly faithful.*

We make use of the last result for showing that the translation from nested logic programs into disjunctive logic programs, as discussed next, is PSM.

4 Main Construction

In this section, we show how logic programs with nested expressions can be efficiently mapped to disjunctive logic programs, preserving the semantics of the respective programs. Although results by Lifschitz *et al.* [23] already provide a reduction of nested logic programs into disjunctive ones (by employing additional transformation steps as given in [18]), that method is exponential in the worst case. This is due to the fact that the transformation relies on distributive laws, yielding an exponential increase of program size whenever the given program contains rules whose heads are in disjunctive normal form or whose bodies are in conjunctive normal form, and the respective expressions are not simple disjunctions or conjunctions of HT-literals.

To avoid such an exponential blow-up, our technique is based on the introduction of new atoms, called *labels*, abbreviating subformula occurrences. This method is derived from structure-preserving normal form translations [34, 31], which are frequently applied in the context of automated reasoning (cf., e.g., [2, 15] for general investigations about structure-preserving normal form translation in finite-valued Gödel logics, and [6, 7] for proof-theoretical issues of such translations for classical and intuitionistic logic). In contrast to theorem proving applications, where the main focus is to provide translations which are satisfiability (or, alternatively, validity) equivalent, here we are interested in somewhat stronger equivalence properties, viz. in the *reconstruction of the answer sets* of the original programs from the translated ones, which involves also an adequate handling of additional minimality criteria.

The overall structure of our translation can be described as follows. Given a nested logic program Π, we perform the following steps:

1. For each $r \in \Pi$, transform $H(r)$ and $B(r)$ into HT-NNF;
2. translate the program into a program containing only rules with conjunctions of HT-literals in their bodies and disjunctions of HT-literals in their heads;
3. eliminate double negations in bodies and heads; and
4. transform the resulting program into a disjunctive logic program, i.e., make all heads negation free.

Steps 1 and 3 are realised by using properties of logic programs as described in [23]; Step 2 represents the central part of our construction; and Step 4 exploits a procedure due to Janhunen [18].

In what follows, for any alphabet $\mathcal{A}$, we define the following new and disjoint alphabets:

- a set $\mathcal{A}_\mathbf{L} = \{\mathbf{L}_\phi \mid \phi \in \mathcal{L}_\mathcal{A}\}$ of labels; and
- a set $\bar{\mathcal{A}} = \{\overline{p} \mid p \in \mathcal{A}\}$ of atoms representing negated atoms.

Furthermore, $NLP^{nnf}_\mathcal{A}$ is the class of all nested logic programs over $\mathcal{A}$ which are in HT-NNF, and $GDLP^{ht}_\mathcal{A}$ is the class of all programs over $\mathcal{A}$ which are defined like generalised logic programs, except that HT-literals may occur in rules instead of ordinary literals.

We assume that for each of the above construction stages, Step i is realized by a corresponding function $\sigma_i(\cdot)$ ($i = 1, \ldots, 4$). The overall transformation is then described by the composed function $\sigma = \sigma_4 \circ \sigma_3 \circ \sigma_2 \circ \sigma_1$, which is a mapping from the set $NLP_\mathcal{A}$ of all programs over $\mathcal{A}$ into the set $DLP_{\mathcal{A}^*}$ of all disjunctive logic program over $\mathcal{A}^* = \mathcal{A} \cup \mathcal{A}_\mathbf{L} \cup \bar{\mathcal{A}}$. More specifically,

$$\sigma_1 : NLP_\mathcal{A} \to NLP^{nnf}_\mathcal{A}$$

translates any nested logic program over $\mathcal{A}$ into a nested program in HT-NNF. Translation

$$\sigma_2 : NLP^{nnf}_\mathcal{A} \to GDLP^{ht}_{\mathcal{A} \cup \mathcal{A}_\mathbf{L}}$$

takes these programs and transforms their rules into simpler ones as described by Step 2, introducing new labels. These rules are then fed into mapping

$$\sigma_3 : GDLP^{ht}_{\mathcal{A} \cup \mathcal{A}_\mathbf{L}} \to GDLP_{\mathcal{A} \cup \mathcal{A}_\mathbf{L}},$$

yielding generalised disjunctive logic programs. Finally,

$$\sigma_4 : GDLP_{\mathcal{A} \cup \mathcal{A}_\mathbf{L}} \to DLP_{\mathcal{A}^*}$$

outputs standard disjunctive logic programs.

As argued in the following, each of these functions is PSM; hence, the overall function $\sigma = \sigma_4 \circ \sigma_3 \circ \sigma_2 \circ \sigma_1$ is PSM as well.

We continue with the technical details, starting with σ_1.

For the first step, we use the procedure $\nu(\cdot)$ from Proposition 3 to transform heads and bodies of rules into HT-NNF.

Definition 4 *The function $\sigma_1 : NLP_{\mathcal{A}} \to NLP_{\mathcal{A}}^{nnf}$ is defined by setting*

$$\sigma_1(\Pi) = \{\nu(H(r)) \leftarrow \nu(B(r)) \mid r \in \Pi\},$$

for any $\Pi \in NLP_{\mathcal{A}}$.

Since, for each expression ϕ, $\nu(\phi)$ is constructible in polynomial time and ϕ is HT-equivalent to $\nu(\phi)$ (cf. Proposition 3), the following result is immediate:

Lemma 2 *The translation σ_1 is PSM.*

The second step is realised as follows:

Definition 5 *The function $\sigma_2 : NLP_{\mathcal{A}}^{nnf} \to GDLP_{\mathcal{A} \cup \mathcal{A}_{\mathbf{L}}}^{ht}$ is defined by setting, for any $\Pi \in NLP_{\mathcal{A}}^{nnf}$,*

$$\sigma_2(\Pi) = \{\mathbf{L}_{H(r)} \leftarrow \mathbf{L}_{B(r)} \mid r \in \Pi\} \cup \gamma(\Pi),$$

where $\gamma(\Pi)$ is constructed as follows:

1. *for each HT-literal l occurring in Π, add the two rules*

$$\mathbf{L}_l \leftarrow l \quad \text{and} \quad l \leftarrow \mathbf{L}_l;$$

2. *for each expression $\phi = (\phi_1 \wedge \phi_2)$ occurring in Π, add the three rules*

$$\mathbf{L}_\phi \leftarrow \mathbf{L}_{\phi_1} \wedge \mathbf{L}_{\phi_2}, \quad \mathbf{L}_{\phi_1} \leftarrow \mathbf{L}_\phi, \quad \mathbf{L}_{\phi_2} \leftarrow \mathbf{L}_\phi;$$

   and

3. *for each expression $\phi = (\phi_1 \vee \phi_2)$ occurring in Π, add the three rules*

$$\mathbf{L}_{\phi_1} \vee \mathbf{L}_{\phi_2} \leftarrow \mathbf{L}_\phi, \quad \mathbf{L}_\phi \leftarrow \mathbf{L}_{\phi_1}, \quad \mathbf{L}_\phi \leftarrow \mathbf{L}_{\phi_2}.$$

This definition is basically an adaption of a structure-preserving normal form translation for intuitionistic logic, as described in [24].

It is quite obvious that σ_2 is modular and, for each $\Pi \in NLP_{\mathcal{A}}^{nnf}$, we have that $\sigma_2(\Pi)$ is constructible in polynomial time. In order to show that σ_2 is strongly faithful, we define a suitable HT-homomorphism as follows.

Sublemma 1 *Let σ_2 be the translation defined above, and let $\sigma_2^* : NLP_{\mathcal{A}} \to NLP_{\mathcal{A} \cup \mathcal{A}_{\mathbf{L}}}$ result from σ_2 by setting $\sigma_2^*(\Pi) = \sigma_2(\Pi)$ if $\Pi \in NLP_{\mathcal{A}}^{nnf}$ and $\sigma_2^*(\Pi) = \Pi$ if $\Pi \in NLP_{\mathcal{A}} \setminus NLP_{\mathcal{A}}^{nnf}$.*

Then, the function $\alpha_{\sigma_2^*} : INT_{\mathcal{A}} \times NLP_{\mathcal{A}} \to INT_{\mathcal{A} \cup \mathcal{A}_{\mathbf{L}}}$, defined as

$$\alpha_{\sigma_2^*}(\mathcal{I}, \Pi) = \langle I_H \cup \lambda_H(\mathcal{I}, \Pi), I_T \cup \lambda_T(\mathcal{I}, \Pi) \rangle,$$

is a σ_2^*-associated HT-homomorphism, where

$$\lambda_w(\mathcal{I}, \Pi) = \{\mathbf{L}_\phi \in \mathcal{A}_{\mathbf{L}} \cap var(\sigma_2^*(\Pi)) \mid \nu_{\mathcal{I}}(w, \phi) = 1\}$$

if $\Pi \in NLP_{\mathcal{A}}^{nnf}$, and $\lambda_w(\mathcal{I}, \Pi) = \emptyset$ otherwise, for any $w \in \{H, T\}$ and any HT-interpretation $\mathcal{I} = \langle I_H, I_T \rangle$ over $\mathcal{A}$.

Hence, according to Theorem 1, σ_2^* is strongly faithful. As a consequence, σ_2 is strongly faithful as well. Thus, the following holds:

Lemma 3 *The function σ_2 is PSM.*

For Step 3, we use a method due to Lifschitz *et al.* [23] for eliminating double negations in heads and bodies of rules. The corresponding function σ_3 is defined as follows:

Definition 6 *Let $\sigma_3 : GDLP_{\mathcal{A} \cup \mathcal{A}_{\mathbf{L}}}^{ht} \to GDLP_{\mathcal{A} \cup \mathcal{A}_{\mathbf{L}}}$ be the function obtained by replacing, for each given program $\Pi \in GDLP_{\mathcal{A} \cup \mathcal{A}_{\mathbf{L}}}^{ht}$, each rule $r \in \Pi$ of form*

$$\phi \vee \neg\neg p \leftarrow \psi \quad \text{by} \quad \phi \leftarrow \psi \wedge \neg p,$$

as well as each rule of form

$$\phi \leftarrow \psi \wedge \neg\neg q \quad \text{by} \quad \phi \vee \neg q \leftarrow \psi,$$

where ϕ and ψ are expressions and $p, q \in \mathcal{A}$.

As shown in [23], performing replacements of the above type results in programs which are strongly equivalent to the original programs. In fact, it is easy to see that such replacements yield transformed programs which are strongly faithful to the original ones. Since these transformations are clearly modular and constructible in polynomial time, we obtain that σ_3 is PSM.

Lemma 4 *The function σ_3 is PSM.*

Finally, we eliminate remaining negations possibly occurring in the heads of rules. To this end, we employ a procedure due to Janhunen [18].

Definition 7 *Let $\sigma_4 : GDLP_{\mathcal{A} \cup \mathcal{A}_{\mathbf{L}}} \to DLP_{\mathcal{A} \cup \mathcal{A}_{\mathbf{L}} \cup \bar{\mathcal{A}}}$ be the function defined by setting, for any program $\Pi \in GDLP_{\mathcal{A} \cup \mathcal{A}_{\mathbf{L}}}$,*

$$\sigma_4(\Pi) = \bar{\Pi} \cup \{\bot \leftarrow (p \wedge \overline{p}),\ \overline{p} \leftarrow \neg p \mid \neg p \text{ occurs in the head of some rule in } \Pi\},$$

where $\overline{\Pi}$ results from Π by replacing each occurrence of a literal $\neg p$ in the head of a rule in Π by $\overline{p}$.

Janhunen showed that replacements of the above kind lead to a transformation which is PFM. As a matter of fact, since his notion of faithfulness is somewhat stricter than ours, the results in [18] actually imply that, for any $\Pi, \Pi' \in GDLP_{\mathcal{A} \cup \mathcal{A}_\mathbf{L}}$, $AS_{\mathcal{A} \cup \mathcal{A}_\mathbf{L}}(\Pi \cup \Pi')$ is given by

$$\{I \cap (\mathcal{A} \cup \mathcal{A}_\mathbf{L}) \mid I \in AS_{\mathcal{A} \cup \mathcal{A}_\mathbf{L} \cup \bar{\mathcal{A}}}(\sigma_4(\Pi) \cup \Pi')\}.$$

However, we need a stronger condition here, viz. that the above equation holds for any $\Pi \in GDLP_{\mathcal{A} \cup \mathcal{A}_\mathbf{L}}$ and any $\Pi' \in NLP_{\mathcal{A} \cup \mathcal{A}_\mathbf{L}}$. We show this by appeal to Theorem 1.

Sublemma 2 *Let σ_4 be the translation defined above, and let $\sigma_4^* : NLP_{\mathcal{A} \cup \mathcal{A}_\mathbf{L}} \to NLP_{\mathcal{A} \cup \mathcal{A}_\mathbf{L} \cup \bar{\mathcal{A}}}$ result from σ_4 by setting $\sigma_4^*(\Pi) = \sigma_4(\Pi)$ if $\Pi \in GDLP_{\mathcal{A} \cup \mathcal{A}_\mathbf{L}}$ and $\sigma_4^*(\Pi) = \Pi$ if $\Pi \in NLP_{\mathcal{A} \cup \mathcal{A}_\mathbf{L}} \setminus GDLP_{\mathcal{A} \cup \mathcal{A}_\mathbf{L}}$.*

Then, the function $\alpha_{\sigma_4^*} : INT_{\mathcal{A} \cup \mathcal{A}_\mathbf{L}} \times NLP_{\mathcal{A} \cup \mathcal{A}_\mathbf{L}} \to INT_{\mathcal{A} \cup \mathcal{A}_\mathbf{L} \cup \bar{\mathcal{A}}}$, defined as

$$\alpha_{\sigma_4^*}(\mathcal{I}, \Pi) = \langle I_H \cup \kappa(\mathcal{I}, \Pi), I_T \cup \kappa(\mathcal{I}, \Pi) \rangle,$$

is a σ_4^*-associated HT-homomorphism, where

$$\kappa(\mathcal{I}, \Pi) = \{\overline{p} \mid \neg p \text{ occurs in the head of some rule in } \Pi \text{ and } p \notin I_T\}$$

if $\Pi \in GDLP_{\mathcal{A} \cup \mathcal{A}_\mathbf{L}}$, and $\kappa(\mathcal{I}, \Pi) = \emptyset$ otherwise, for any HT-interpretation $\mathcal{I} = \langle I_H, I_T \rangle$ over $\mathcal{A} \cup \mathcal{A}_\mathbf{L}$.

Observe that, in contrast to the definition of function $\alpha_{\sigma_2^*}$ from Sublemma 1, here the same set of newly introduced atoms is added to both worlds. As before, we obtain that σ_4^* is strongly faithful, and hence that σ_4 is strongly faithful as well.

Lemma 5 *The function σ_4 is PSM.*

Summarising, we obtain our main result, which is as follows:

Theorem 2 *Let $\sigma_1, \ldots, \sigma_4$ be the functions defined above. Then, the composed function $\sigma = \sigma_4 \circ \sigma_3 \circ \sigma_2 \circ \sigma_1$, mapping nested logic programs over alphabet $\mathcal{A}$ into disjunctive logic programs over alphabet $\mathcal{A} \cup \mathcal{A}_\mathbf{L} \cup \bar{\mathcal{A}}$, is polynomial, strongly faithful, and modular.*

Since strong faithfulness implies faithfulness, we get the following corollary:

Corollary 1 *For any nested logic program Π over $\mathcal{A}$, the answer sets of Π are in a one-to-one correspondence to the answer sets of $\sigma(\Pi)$, determined by the following equation:*

$$AS_{\mathcal{A}}(\Pi) = \{I \cap \mathcal{A} \mid I \in AS_{\mathcal{A}^*}(\sigma(\Pi))\},$$

where $\mathcal{A}^* = \mathcal{A} \cup \mathcal{A}_\mathbf{L} \cup \bar{\mathcal{A}}$.

We conclude with a remark concerning the construction of function σ_2. As pointed out previously, this mapping is based on a structure-preserving normal form translation for intuitionistic logic, as described in [24]. Besides the particular type of translation used here, there are also other, slightly improved structure-preserving normal form translations in which fewer rules are introduced, depending on the polarity of the corresponding subformula occurrences. However, although such optimised methods work in monotonic logics, they are not sufficient in the present setting. For instance, in a possible variant of translation σ_2 based on the polarity of subformula occurrences, instead of introducing all three rules for an expression ϕ of form $(\phi_1 \wedge \phi_2)$, only $\mathbf{L}_\phi \leftarrow \mathbf{L}_{\phi_1} \wedge \mathbf{L}_{\phi_2}$ is used if ϕ occurs in the body of some rule, or both $\mathbf{L}_{\phi_1} \leftarrow \mathbf{L}_\phi$ and $\mathbf{L}_{\phi_2} \leftarrow \mathbf{L}_\phi$ are used if ϕ occurs in the head of some rule, and analogous manipulations are performed for atoms and disjunctions. Applying such an encoding to $\Pi = \{p \leftarrow;\ q \leftarrow;\ r \vee (p \wedge q) \leftarrow\}$ over $\mathcal{A}_0 = \{p, q, r\}$ yields a translated program possessing two answer sets, say S_1 and S_2, such that $S_1 \cap \mathcal{A}_0 = \{p, q\}$ and $S_2 \cap \mathcal{A}_0 = \{p, q, r\}$, although only $\{p, q\}$ is an answer set of Π.

5 Conclusion

We have developed a translation of logic programs with nested expressions into disjunctive logic programs. We have proven that our translation is polynomial, strongly faithful, and modular. This allows us to utilise off-the-shelf disjunctive logic programming systems for interpreting nested logic programs. In fact, we have implemented our translation as a front end for the system DLV [8, 9]. The corresponding compiler is implemented in Prolog and can be downloaded from the Web at URL

`http://www.cs.uni-potsdam.de/~torsten/nlp`.

Our technique is based on the introduction of new atoms, abbreviating subformula occurrences. This method has its roots in structure-preserving normal form translations [34, 31], which are frequently used in automated deduction. In contrast to theorem proving applications, however, where the main focus is to provide satisfiability (or, alternatively, validity) preserving translations, we are concerned with much stronger equivalence properties, involving additional minimality criteria, since our goal is to *reconstruct* the answer sets of the original programs from the translated ones.

With the particular labeling technique employed here, our translation avoids the risk of an exponential blow-up in the worst-case, faced by a previous approach of Lifschitz *et al.* [23] due to the usage of distributivity laws. However, this is not to say that our translation is *always* the better choice. As in classical theorem proving, it is rather a matter of experimental studies under which circumstances

which approach is the more appropriate one. To this end, besides the implementation of our structural translation, we have also implemented the distributive translation into disjunctive logic programs in order to conduct experimental results. These experiments are subject to current research.

Also, we have introduced the concept of *strong faithfulness*, as a generalisation of (standard) faithfulness and strong equivalence. This allows us, for instance, to translate, in a semantics-preserving way, arbitrary program parts and leave the remaining program unaffected.

References

[1] M. Baaz, U. Egly, and A. Leitsch. Normal Form Transformations. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume I, chapter 5, pages 273–333. Elsevier Science B.V., 2001.

[2] M. Baaz and C. G. Fermüller. Resolution-Based Theorem Proving for Many-Valued Logics. *Journal of Symbolic Computation*, 19(4):353–391, 1995.

[3] C. Baral and C. Uyan. Declarative Specification and Solution of Combinatorial Auctions Using Logic Programming. In *Proceedings of the Sixth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR-01)*, volume 2173 of *Lecture Notes in Artificial Intelligence*, pages 186–199, Berlin, 2001. Springer Verlag.

[4] M. Cadoli, A. Giovanardi, and M. Schaerf. An Algorithm to Evaluate Quantified Boolean Formulae. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*, pages 262–267, Menlo Park, 1998. AAAI Press.

[5] D. de Jongh and L. Hendriks. Characterization of Strongly Equivalent Logic Programs in Intermediate Logics. Technical report, 2001. Preprint at `http://turing.wins.uva.nl/~lhendrik/`.

[6] U. Egly. On Different Structure-Preserving Translations to Normal Form. *Journal of Symbolic Computation*, 22(2):121–142, 1996.

[7] U. Egly. On Definitional Transformations to Normal Form for Intuitionistic Logic. *Fundamenta Informaticae*, 29(1,2):165–201, 1997.

[8] T. Eiter, N. Leone, C. Mateis, G. Pfeifer, and F. Scarcello. A Deductive System for Non-Monotonic Reasoning. In *Proceedings of the Fourth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR-97)*, volume 1265 of *Lecture Notes in Artificial Intelligence*, pages 363–374, Berlin, 1997. Springer Verlag.

[9] T. Eiter, N. Leone, C. Mateis, G. Pfeifer, and F. Scarcello. The KR System dlv: Progress Report, Comparisons and Benchmarks. In *Proceedings of the Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR-98)*, pages 406–417. Morgan Kaufmann Publishers, 1998.

[10] R. Feldmann, B. Monien, and S. Schamberger. A Distributed Algorithm to Evaluate Quantified Boolean Formulas. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-00)*, pages 285–290. AAAI Press, 2000.

[11] M. Gelfond, M. Balduccini, and J. Galloway. Diagnosing Physical Systems in A-Prolog. In *Proceedings of the Sixth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR-01)*, volume 2173 of *Lecture Notes in Artificial Intelligence*, pages 213–225, Berlin, 2001. Springer Verlag.

[12] M. Gelfond and V. Lifschitz. Classical Negation in Logic Programs and Disjunctive Databases. *New Generation Computing*, 9:365–385, 1991.

[13] E. Giunchiglia, M. Narizzano, and A. Tacchella. QUBE: A System for Deciding Quantified Boolean Formulas Satisfiability. In *Proceedings of the First International Joint Conference on Automated Reasoning (IJCAR-01)*, volume 2083 of *Lecture Notes in Computer Science*, pages 364–369, 2001.

[14] K. Gödel. Zum intuitionistischen Aussagenkalkül. *Anzeiger der Akademie der Wissenschaften in Wien*, pages 65–66, 1932.

[15] R. Hähnle. Short Conjunctive Normal Forms in Finitely Valued Logics. *Journal of Logic and Computation*, 4(6):905–927, 1994.

[16] K. Heljanko and I. Niemelä. Bounded LTL Model Checking with Stable Models. In *Proceedings of the Sixth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR-01)*, volume 2173 of *Lecture Notes in Artificial Intelligence*, pages 200–212, Berlin, 2001. Springer Verlag.

[17] T. Janhunen. On the Intertranslatability of Autoepistemic, Default and Priority Logics, and Parallel Circumscription. In *Proceedings of the Sixth European Workshop on Logics in Artificial Intelligence (JELIA-98)*, volume 1489 of *Lecture Notes in Artificial Intelligence*, pages 216–232, Berlin, 1998. Springer Verlag.

[18] T. Janhunen. On the Effect of Default Negation on the Expressiveness of Disjunctive Rules. In *Proceedings*

of the Sixth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR-01), volume 2173 of *Lecture Notes in Artificial Intelligence*, pages 93–106, Berlin, 2001. Springer Verlag.

[19] H. Kleine-Büning, M. Karpinski, and A. Flögel. Resolution for Quantified Boolean Formulas. *Information and Computation*, 117(1):12–18, 1995.

[20] R. Letz. Advances in Decision Procedures for Quantified Boolean Formulas. In *Proceedings of the IJCAR-01 Workshop on Theory and Applications of Quantified Boolean Formulas*, Technical Report DII 12/01, Università degli Studi di Siena, Dipartimento di Ingegneria dell'Informazione, pages 55–64, 2001.

[21] V. Lifschitz. Answer Set Planning. In *Proceedings of the Sixteenth International Conference on Logic Programming (ICLP-99)*, volume 1730 of *Lecture Notes in Artificial Intelligence*, pages 23–37, Berlin, 1999. Springer Verlag.

[22] V. Lifschitz, D. Pearce, and A. Valverde. Strongly Equivalent Logic Programs. *ACM Transactions on Computational Logic*, 2(4):526–541, 2001.

[23] V. Lifschitz, L. Tang, and H. Turner. Nested Expressions in Logic Programs. *Annals of Mathematics and Artificial Intelligence*, 25(3-4):369–389, 1999.

[24] G. Mints. Resolution Strategies for the Intuitionistic Logic. In B. Mayoh, E. Tyugu, and J. Penjaam, editors, *Constraint Programming: NATO ASI Series*, pages 282–304. Springer Verlag, 1994.

[25] I. Niemelä and P. Simons. Smodels: An Implementation of the Stable Model and Well-Founded Semantics for Normal Logic Programs. In *Proceedings of the Fourth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR-97)*, volume 1265 of *Lecture Notes in Artificial Intelligence*, pages 420–429, Berlin, 1997. Springer Verlag.

[26] D. Pearce. A New Logical Characterisation of Stable Models and Answer Sets. In *Non-Monotonic Extensions of Logic Programming*, pages 57–70. Springer Verlag, 1997.

[27] D. Pearce. From Here to There: Stable Negation in Logic Programming. In *What is Negation?* Kluwer, 1999.

[28] D. Pearce, I. de Guzmán, and A. Valverde. A Tableau Calculus for Equilibrium Entailment. In *Proceedings of the Ninth International Conference on Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX-00)*, volume 1847 of *Lecture Notes in Artificial Intelligence*, pages 352–367, Berlin, 2000. Springer Verlag.

[29] D. Pearce, I. de Guzmán, and A. Valverde. Computing Equilibrium Models Using Signed Formulas. In *Proceedings of the First International Conference on Computational Logic (CL-00)*, volume 1861 of *Lecture Notes in Computer Science*, pages 688–702, Berlin, 2000. Springer Verlag.

[30] D. Pearce, H. Tompits, and S. Woltran. Encodings for Equilibrium Logic and Logic Programs with Nested Expressions. In *Progress in Artificial Intelligence: Proceedings of the Tenth Portuguese Conference on Artificial Intelligence (EPIA-01)*, volume 2258 of *Lecture Notes in Artificial Intelligence*, pages 306–320, Berlin, 2001. Springer Verlag.

[31] D. A. Plaisted and S. Greenbaum. A Structure Preserving Clause Form Translation. *Journal of Symbolic Computation*, 2(3):293–304, 1986.

[32] J. Rintanen. Improvements to the Evaluation of Quantified Boolean Formulae. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 1192–1197. Morgan Kaufmann Publishers, 1999.

[33] J. Siekmann and G. Wrightson, editors. *Automation of Reasoning: Classical Papers in Computational Logic 1967–1970*, volume 2. Springer-Verlag, 1983.

[34] G. Tseitin. On the Complexity of Proofs in Propositional Logics. *Seminars in Mathematics*, 8, 1970. Reprinted in [33].

Logic Programming with Ordered Disjunction

Gerhard Brewka
Universität Leipzig
Institut für Informatik
Augustusplatz 10-11
04109 Leipzig, Germany
brewka@informatik.uni-leipzig.de

Abstract

Logic programs with ordered disjunction (*LPODs*) combine ideas underlying Qualitative Choice Logic [5] and answer set programming. Logic programming under answer set semantics is extended with a new connective called ordered disjunction. The new connective allows us to represent alternative, ranked options for problem solutions in the heads of rules: $A \times B$ intuitively means: if possible A, but if A is not possible then at least B. The semantics of logic programs with ordered disjunction is based on a preference relation on answer sets. *LPODs* are useful for applications in design and configuration and can serve as a basis for qualitative decision making.

1 Introduction

In a recent paper [5] a propositional logic called Qualitative Choice Logic (*QCL*) was introduced. The logic contains a new connective $\times$ representing ordered disjunction. Intuitively, $A \times B$ stands for: if possible A, but if A is impossible then (at least) B. This connective allows context dependent preferences to be represented in a simple and elegant fashion. As a simple example consider the preferences for booking a hotel for a conference. Assume the most preferred option is to be within walking distance from the conference site, the second best option is to have transportation provided by the hotel, the third best is public transportation. This can simply be represented as

$$walking \times hotel\text{-}transport \times public\text{-}transport$$

From a description of available hotels, a disjunction expressing that one of the hotels must be picked, and the above formula *QCL* is able to derive the hotel which satisfies best the given preferences (if there is more than one such hotel a corresponding disjunction is concluded).

The semantics of the logic is based on degrees of satisfaction of a formula in a classical model. The degrees, intuitively, measure disappointment and induce a preference relation on models. Consequence is defined in terms of most preferred models. It is argued in that paper that there are numerous useful applications, e.g. in configuration and design.

In this paper we want to combine ideas underlying *QCL* with logic programming. More precisely, we want to investigate logic programs based on rules with ordered disjunction in the heads. We call such programs logic programs with ordered disjunction (*LPODs*).

The semantical framework in which the investigation will be carried out is that of answer set semantics [11]. Logic programs under answer set semantics have emerged as a new promising programming paradigm dubbed answer set programming. There are numerous interesting AI applications of answer set programming, for instance in planning [14] and configuration [21]. One of the reasons for this success is the availability of highly efficient systems for computing answer sets like *smodels* [15] and *dlv* [10].

We think it is worthwile to investigate simple representations of context dependent preferences in the answer set programming paradigm. Our combination of ideas from *QCL* and answer set programming will lead to an approach which is less expressive than *QCL* in one respect: the syntax of *LPODs* restricts the appearance of ordered disjunction to the head of rules. On the other hand, we inherit from answer set programming the nonmonotonic aspects which are due to default negation. This allows us to combine default knowledge with knowledge about preferences and desires in a simple and elegant way.

The basic intuition underlying our approach can be described as follows: we will use the ordered disjunctions in rule heads to select some of the answer sets of a program as the preferred ones. Consider a program containing the rule

$$A \times B \leftarrow C$$

If S_1 is an answer set containing C and A and S_2 is an answer set containing C and B but not A, then - ceteris paribus (other things being equal) - S_1 is preferred over S_2. Of course, we have to give precise meaning to the ceteris paribus phrase. Intuitively ceteris paribus is to be read as S_1 and S_2 satisfy the other rules in the program equally well.

We will show that under certain conditions reasoning from most preferred answer sets yields optimal problem solutions. In more general decision making settings the preference relation on answer sets provides a basis for best possible choices given a specific decision strategy.

We will restrict our discussion in this paper to propositional programs. However, as usual in answer set programming, we admit rule schemata containing variables bearing in mind that these schemata are just convenient representations for the set of their ground instances.

The rest of the paper is organized as follows. In the next section we introduce syntax and semantics of *LPODs*. We define the degree of satisfaction of a rule in an answer set and show how to use the degrees to determine a preference relation on answer sets. Conclusions are defined as the literals true in all preferred answer sets. Sect. 3 discusses some simple examples and potential applications. Sect. 4 investigates implementation issues. Sect. 5 shows how *LPODs* can serve as a basis for a qualitative decision theory. The last section discusses related work and concludes.

2 Logic programs with ordered disjunction

Logic programming with ordered disjunction is an extension of logic programming with two kinds of negation (default and strong negation) [11]. The new connective $\times$ representing ordered disjunction is allowed to appear in the head of rules only. A (propositional) *LPOD* thus consists of rules of the form

$$C_1 \times \ldots \times C_n \leftarrow A_1, \ldots, A_m, \text{not } B_1, \ldots, \text{not } B_k$$

where the C_i, A_j and B_l are ground literals.

The intuitive reading of the rule head is: if possible C_1, if C_1 is not possible then C_2, ..., if all of $C_1, \ldots, C_{n-1}$ are not possible then C_n. The literals C_i are called choices of the rule. Extended logic programs with two negations are a special case where $n = 1$ for all rules. As usual we omit $\leftarrow$ whenever $m = 0$ and $k = 0$, that is, if the rule is a fact. Moreover, rules of the form $\leftarrow$ *body* (constraints) are used as abbreviations for $p \leftarrow$ *body*, not p for some p not appearing in the rest of the program. The effect is that no answer sets containing *body* exist.

Before defining the semantics of *LPODs* a few observations are in order. As already mentioned in the introduction we want to use the ranking of literals in the head of rules to select some of the answer sets of a program as the preferred ones. But what are the answer sets of a program among which to make this selection?

Since ordered disjunction is a particular prioritized form of disjunction it seems like a natural idea to base the semantics of *LPODs* on one of the standard semantics for disjunctive logic programs, for instance Gelfond and Lifschitz's semantics [11].

Unfortunately, this doesn't work. The problem is that most of the semantics for disjunctive logic programs have minimality built in. For instance, according to Gelfond and Lifschitz, S is an answer set of a disjunctive logic program P iff S is a minimal set of literals which is logically closed, and closed under the S-reduct of P. The S-reduct of P is obtained from P by (1) deleting all rules r from P such that not B_j in the body of r and $B_j \in S$, and (2) deleting all default negated literals from the remaining rules. A set of literals S is closed under a rule r if one of the literals in the head of r is in S whenever the body is true in S (see [11] for the details).

In this approach answer sets are minimal: if S_1 and S_2 are answer sets of a disjunctive program P and $S_1 \subseteq S_2$, then $S_2 \subseteq S_1$.

Minimality is not always wanted for *LPODs*. Consider the following two facts:

1) $A \times B \times C$
2) $B \times D$

The single best way of satisfying both ordered disjunctions is obviously to make A and B true, that is, we would expect $\{A, B\}$ to be the single preferred answer set of this simple *LPOD*. However, since B is sufficient to satisfy both disjunctions, the set $\{A, B\}$ is not even an answer set of the corresponding disjunctive logic program (where $\times$ is replaced by $\vee$) according to the

semantics of [11]: the built in minimality precludes sets containing both A and B from consideration.

We thus have to use a semantics which is not minimal. Indeed, there is such a semantics, the possible models semantics proposed by Sakama and Inoue [18]. It is based on so-called split programs, that is, disjunction free programs which contain arbitrary subsets of single head rules obtained from disjunctive rules by deleting all but one alternatives in the head.

Unfortunately, also this semantics is inadequate, this time for opposite reasons: it admits too many literals in answer sets. Consider the disjunctive logic program

1) $A \vee B \vee C$

There are seven split programs corresponding to the nonempty subsets of the literals of the fact. The split program containing the facts A, B, C generates the possible model where A, B, C is true.

Let us replace disjunction by ordered disjunction in this formula. According to our intuitive discussion we want to read the rule as "if possible A, if this is not possible then B, and if also B is not possible then C". Under this reading models containing more than one of the literals in the head do not seem justified on the basis of a single rule (they may be justified by different rules, though).

For this reason we will not allow cases where a single rule of the original program gives rise to more than one rule in the split program. There is a further complication: consider the program:

1) $A \times B \times C$
2) A

We do not want to obtain $\{A, B\}$ as an answer set from the split program consisting of these 2 atomic facts since again this does not correspond to the intuitive reading of the first rule (B only if A is not possible). We therefore have to use slightly more complicated rules in split programs.

Definition 1 *Let $r = C_1 \times \ldots \times C_n \leftarrow body$ be a rule. For $k \leq n$ we define the kth option of r as*

$$r^k = C_k \leftarrow body, \text{not } C_1, \ldots, \text{not } C_{k-1}.$$

Definition 2 *Let P be an LPOD. P' is a split program of P if it is obtained from P by replacing each rule in P by one of its options.*

Here is a simple example. Let P consist of the rules

1) $A \times B \leftarrow \text{not } C$
2) $B \times C \leftarrow \text{not } D$

We obtain 4 split programs

$A \leftarrow \text{not } C$	$A \leftarrow \text{not } C$
$B \leftarrow \text{not } D$	$C \leftarrow \text{not } D, \text{not } B$
$B \leftarrow \text{not } C, \text{not } A$	$B \leftarrow \text{not } C, \text{not } A$
$B \leftarrow \text{not } D$	$C \leftarrow \text{not } D, \text{not } B$

Split programs do not contain ordered disjunction. We thus can define:

Definition 3 *Let P be an LPOD. A set of literals A is an answer set of P if it is a consistent answer set of a split program P' of P.*

We exclude inconsistent answer sets from consideration since they do not represent possible problem solutions. In the example above we obtain 3 answer sets: $\{A, B\}, \{C\}, \{B\}$. Note that one of the answer sets is a proper subset of another answer set. On the other hand, none of the rules in the original *LPOD* sanctions more than one literal in any of the answer sets, as intended.

Not all of the answer sets satisfy our most intended options. Clearly, $\{B, A\}$ gives us the best options for both rules, whereas $\{C\}$ gives only the second best option for 2) and $\{B\}$ the second best option for 1). To distinguish between more and less intended answer sets we introduce the degree of satisfaction of a rule in an answer set:

Definition 4 *Let S be an answer set of an LPOD P. S satisfies the rule*

$$C_1 \times \ldots \times C_n \leftarrow A_1, \ldots, A_m, \text{not } B_1, \ldots, \text{not } B_k$$

- *to degree 1 if $A_j \notin S$, for some j, or $B_i \in S$, for some i,*
- *to degree j $(1 \leq j \leq n)$ if all $A_j \in S$, no $B_i \in S$, and $j = min\{r \mid C_r \in S\}$.*

Proposition 1 *If A is an answer set of P then A satisfies all rules of P to some degree.*[1]

Proof: Let r be a rule of P. If S is an answer set of P, then there is a split program P' such that S is an answer set of P'. Let r^i be the rule in P' generated

[1]The other direction of the proposition does obviously not hold. For example, the set $\{A\}$ satisfies the rule $B \leftarrow \text{not } A$, but is not an answer set for the program consisting of this single rule.

from r. Since S is an answer set of P' either the body of r^i is satisfied in S and thus C_i is contained in S, in which case r is satisfied to degree i or smaller, or the body of r^i is not satisfied in S, in which case r is satisfied to degree 1 in S, or there is a better choice than C_k, $k < i$, in S and r is satisfied to degree k. □

We use the degrees of satisfaction of a rule to define a preference relation on answer sets. There are different ways of doing this. For instance, we could simply add up the satisfaction degrees of all rules and prefer those answer sets where the total sum is minimal. Although this may be reasonable in certain applications, this approach makes quite strong assumptions about the commensurability of choices in different rule heads. In [5] a lexicographic ordering of models based on the number of premises satisfied to a particular degree was proposed. This lexicographic ordering has a highly syntactic flavour. Therefore, we will use here a somewhat more cautious preference relation (in the sense that fewer answer sets are considered better than others) based on set inclusion of the rules satisfied to certain degrees:

Definition 5 *For a set of literals S, let $S^i(P)$ denote the set of rules in P satisfied by S to degree i. Let S_1 and S_2 be answer sets of an LPOD P. S_1 is preferred to S_2 ($S_1 > S_2$) iff there is i such that $S_2^i(P) \subset S_1^i(P)$, and for all $j < i$, $S_1^j(P) = S_2^j(P)$.*

Definition 6 *A set of literals S is a preferred answer set of an LPOD P iff S is an answer set of P and there is no answer set S' of P such that $S' > S$.*

Definition 7 *A literal l is a conclusion of an LPOD P iff l is contained in all preferred answer sets of P.*

Consider again the program

1) $A \times B \leftarrow \text{not } C$
2) $B \times C \leftarrow \text{not } D$

As discussed before we obtain the 3 answer sets: $S_1 = \{A, B\}$, $S_2 = \{C\}$ and $S_3 = \{B\}$. S_1 satisfies both rules with degree 1, $\{C\}$ satisfies 1) to degree 1 but 2) to degree 2. $\{B\}$ satisfies 1) to degree 2 and 2) to degree 1. The single preferred answer set is thus S_1, as intended, and A and B are the conclusions of the program.

3 Examples

LPODs allow us - like normal logic programs - to express incomplete and defeasible knowledge through the use of default negation. In addition, they provide means to represent preferences among intended properties of problem solutions. Moreover, these preferences may depend on the current context.

In this section we discuss several examples illustrating potential uses of *LPODs*. The first example is about how to spend a free afternoon. You like to go to the beach, but also to the cinema. Normally you prefer the cinema over the beach, unless it is hot (which is the exception in the area where you live, except during the summer). If it is hot the beach is preferred over the cinema. In summer it is normally hot, but there are exceptions. If it rains the beach is out of question. This information can be represented using the following rules:

1) $cinema \times beach \leftarrow \text{not } hot$
2) $beach \times cinema \leftarrow hot$
3) $hot \leftarrow \text{not } \neg hot, summer$
4) $\neg beach \leftarrow rain$

Without further information about the weather we obtain the single preferred answer set $S_1 = \{cinema\}$. There is no information that it might be hot, so rule 1) will determine the preferences. S_1 satisfies all rules to degree 1.

Now assume the fact *summer* is additionally given. In this case we obtain $S_2 = \{summer, hot, beach\}$ as the single preferred answer set. Again this answer set satisfies all rules to degree 1.

Next assume that, in addition to *summer* also the literal $\neg hot$ is given. The single preferred answer set now is $S_3 = \{summer, \neg hot, cinema\}$. All rules are satisfied to degree 1.

Finally, assume the additional facts are *summer* and *rain*. Now the single preferred answer set (and in fact the single answer set) is

$$S_4 = \{summer, rain, hot, \neg beach, cinema\}.$$

Note that this time it is not possible to satisfy all rules to degree 1: rule 2) is satisfied to degree 2 only. As often in real life, there are situations where the best options simply do not work out.

We think that *LPODs* are very well suited for representing problems where a certain choice has to be made or, more generally, where a number of components have to be chosen for a certain configuration task. The general idea would be to have

- for each component a set of rules describing its properties,
- rules describing which components are needed for

the configuration to be complete; this may depend on other components chosen,

- rules describing intended properties of the solution we want to generate. The involved preferences may be context dependent, and
- a description of the case at hand.

In each case default knowledge can be used to describe what is normally the case. Consider the problem of configuring a menu. The menu should consist of a *starter*, a *main* course, a *dessert* and a *beverage*. As a *starter* you prefer *soup* over *salad*. As *main* course *fish*, *beef* and *lasagne* are possible (this is all you are able to cook) and your preferences are in this order. Of course, if the visitor is *vegetarian* the first two (as well as the *soup*) are out of the question. In case of *beef* you prefer *red* wine over *white* wine over mineral *water*, otherwise the order between wines is reversed. Only *ice−coffee* and *tiramisu* is available as a *dessert*. If *tiramisu* is chosen, then an extra *coffee* is necessary. You prefer *espresso* over *cappucino*.

The possible components thus are *soup*, *salad*, *fish*, *beef*, *lasagne*, *ice−coffee*, *tiramisu*, *espresso*, *cappucino*, *red*, *white* and *water*. The following properties of the components are relevant:

$\neg vegetarian \leftarrow beef$ $\quad alcohol \leftarrow white$
$\neg vegetarian \leftarrow fish$ $\quad alcohol \leftarrow red$
$\neg vegetarian \leftarrow soup$

The needed components are

$starter$ $\quad beverage$
$main$ $\quad coffee \leftarrow tiramisu$
$dessert$

The preferences are as follows:

$soup \times salad \leftarrow starter$
$fish \times beef \times lasagne \leftarrow main$
$red \times white \times water \leftarrow beverage, beef$
$white \times red \times water \leftarrow beverage, \text{not } beef$
$espresso \times cappuccino \leftarrow coffee$
$ice{-}coffee \leftarrow \text{not } tiramisu, dessert$
$tiramisu \leftarrow \text{not } ice{-}coffee, dessert$

Now, given a description of the case at hand, e.g. whether the visitor is vegetarian or not, drinks alcohol or not, likes fish etc. the preferred answer sets will determine a menu which satisfies the preferences as much as possible. The last two rules are necessary to make sure that one of the desserts is picked. For the other courses this is implicit in the specified preferences. In the language of [16] these rules can be represented as the cardinality constraint rule $1\{ice{-}coffee, tiramisu\}1 \leftarrow dessert$. Combinations of *LPOD*s and such constraints are a topic of further research.

4 Computation

The first question to ask is whether *LPOD*s can simply be reduced to standard logic programs with two kinds of negation. In that case standard answer set programming techniques would be sufficient for computing consequences of *LPOD*s. We will show that a seemingly natural translation does not yield the intended answer sets.

Definition 8 *The pseudo-translation $trans(r)$ of a rule*

$$r = C_1 \times \ldots \times C_n \leftarrow body$$

is the collection of rules

$$\begin{array}{ll} C_1 & \leftarrow body, \text{not } -C_1 \\ C_2 & \leftarrow body, \text{not } -C_2, -C_1 \\ \ldots & \\ C_{n-1} & \leftarrow body, \text{not } -C_{n-1}, -C_1, \ldots, -C_{n-2} \\ C_n & \leftarrow body, -C_1, \ldots, -C_{n-1} \end{array}$$

where $-C$ is the complement of C, that is $\neg C$ if C is an atom and C' if $C = \neg C'$. The pseudo-translation $trans(P)$ of an LPOD P is

$$trans(P) = \bigcup_{r \in P} trans(r)$$

The pseudo-translation creates for each option C_i in the head of r a rule with head C_i which has the negation of the better options as additional body literals. In addition, the rule is made defeasible by adding the default negation of the complement of C_i to the body. There is an exception: the rule generated for the last option is not made defeasible this way since at least one of the options must be true whenever the body of the original rule is true.

Although this translation seems natural it does not work. Consider the following example:

1) $a \times b$
2) $p \leftarrow \text{not } p, a$

The single preferred answer set is $\{b\}$. The pseudo-translation is

1) $a \leftarrow \text{not } \neg a$
2) $b \leftarrow \neg a$
3) $p \leftarrow \text{not } p, a$

The resulting program has no answer set. In fact, we can prove the following proposition:

Proposition 2 *There is no translation trans from LPODs to extended logic programs (without ordered disjunction) such that for each program P the preferred answer sets of P and the answer sets of $trans(P)$ coincide.*

Proof: The proposition follows from the fact that preferred answer sets of *LPODs* are not necessarily subset minimal. Consider the program $a \times b$; $c \times b \leftarrow a$; $\neg c$. The preferred answer sets are $S_1 = \{b, \neg c\}$ and $S_2 = \{a, b, \neg c\}$. Clearly, $S_1 \subset S_2$. There is thus no extended logic program with these answer sets. □

Of course, this does not exclude the possibility of translations to programs containing some extra atoms. This is a topic of further study.

For the computation of preferred answer sets we suggest to search through the space of split programs. Fortunately, the split programs can be ordered according to the options they contain. Let us first introduce some notation. For an *LPOD* P and a split program P' of P we let

$P^k_{P'} = \{r \in P \mid k$ smallest integer such that $r^k \in P'\}$.

Now consider the following preference relation on split programs:

Definition 9 *Let P_1 and P_2 be two split programs of the LPOD P. We define $P_1 > P_2$ iff there is a k such that $P^k_{P_2} \subset P^k_{P_1}$ and for all $j < k$, $P^j_{P_1} = P^j_{P_2}$.*

The split programs of P form a lattice whose top element is the program with all the best options and whose bottom element is the program with all the worst options. The *lub* of two programs picks for each rule in P the best option contained in one of the programs, the *glb* picks the worst option.

Among the split programs we are interested in consistent ones which are $>$-maximal. A program is consistent if it possesses at least one consistent answer set.

Definition 10 *A split program P' of P is called P-optimal iff P' is consistent, and there is no consistent split program P'' of P such that $P'' > P'$.*

We are able to prove the following proposition:

Proposition 3 *If S is a preferred answer set of an LPOD P then there is a P-optimal split program P' of P such that S is an answer set of P'.*

Proof: Sketch: Let S be a preferred answer set of P. Construct a split program P' as follows: for each rule $r = C_1 \times \ldots \times C_n \leftarrow body \in P$ choose r^1 if *body* is false in S and r^k if body is true in S and k is the smallest integer such that $C_k \in S$. The split program obtained this way has S as an answer set and can be shown to be optimal. □

We can thus use the preference ordering on split programs to search for preferred answer sets, starting from the best split program. Whenever a split program possesses a consistent answer set we can eliminate all programs below from further consideration.

5 Decision Making using LPODs

In Section 3 we discussed several examples illustrating the notions underlying LPODs. The examples were chosen in such a way that the most preferred answer sets in each case provided the best solutions to the problem at hand. Later in this section we will analyze why this worked for the chosen examples.

In more general decision making settings it is not sufficient to consider the most preferred answer sets only since this amounts to an extremely optimistic view about how the world will behave (this view is sometimes called wishful thinking). As is well-known in decision theory, for realistic models of decision making it is necessary to clearly distinguish what is under the control of the agent (and thus may constitute the agent's decision) from what is not. We will do this by distinguishing a subset of the literals in a program as decision literals.

In this section we describe a general methodology for qualitative decision making based on *LPODs*. The basic idea is to use *LPODs* to describe possible actions or decisions and their consequences, states of the world and desired outcomes. The representation of desires induces, through ordered disjunction, a preference ordering on answer sets representing their desirability. Based on this preference ordering an ordering on possible decisions can be defined based on some decision strategy.

Let us describe the necessary steps more precisely:

1. Among the literals in the logical language distinguish a set of decision literals C. C is the set of literals the agent can decide upon. It's the agent's decision which makes them true. A decision is a consistent subset of C.
2. Represent the different alternative decisions which can be made by the agent. This can be done using standard answer set programming techniques. Note that certain options may lead to additional

choices that need to be made.

3. Represent the different alternative states of the world. Again standard answer set programming techniques apply.
4. Represent relationships between and consequences of different alternatives.
5. Represent desired properties. This is where ordered disjunction comes into play. Of course, desires may be context-dependent.
6. Use the preference relation on answer sets derived form the satisfaction degrees of rules to induce a preference relation on possible decisions. Of course, there are different ways to do this corresponding to different attitudes of the agent towards risk.
7. Pick one of the most preferred decisions.

We will use Savage's famous rotten egg example [19] to illustrate this methodology. An agent is preparing an omelette. 5 fresh eggs are already in the omelette. There is one more egg. It is uncertain whether this egg is fresh or rotten. The agent can

- add it to the omelette which means the whole omelette may be wasted, or
- throw it away, which means one egg may be wasted, or
- put it in a cup, check whether it is ok or not and put it to the omelette in the former case, throw it away in the latter. In any case, a cup has to be washed if this option is chosen.

In this example, the decision literals correspond to the three possible actions, that is C is the set of literals built from $\{in{-}omelette, in{-}cup, throw{-}away\}$. Here are the rules which generate the possible decisions and states of the world:

$$in{-}omelette \leftarrow \text{not}\, in{-}cup, \text{not}\, throw{-}away$$
$$in{-}cup \leftarrow \text{not}\, in{-}omelette, \text{not}\, throw{-}away$$
$$throw{-}away \leftarrow \text{not}\, in{-}cup, \text{not}\, in{-}omelette$$
$$rotten \leftarrow \text{not}\, fresh$$
$$fresh \leftarrow \text{not}\, rotten$$

For our example it is not necessary to specify that the different actions and states of the egg are mutually exclusive. It is guaranteed by the rules that only one of the exclusive options is contained in an answer set.

We next define the effects of the different choices:

$$5{-}omelette \leftarrow throw{-}away$$
$$6{-}omelette \leftarrow fresh, in{-}omelette$$
$$0{-}omelette \leftarrow rotten, in{-}omelette$$
$$6{-}omelette \leftarrow fresh, in{-}cup$$
$$5{-}omelette \leftarrow rotten, in{-}cup$$
$$\neg wash \leftarrow \text{not}\, in{-}cup$$
$$wash \leftarrow in{-}cup$$

For the different omelettes we must state that they are mutually inconsistent. We omit the 6 rules necessary for representing this. They are of the form $\neg x{-}omelette \leftarrow y{-}omelette$ with $x \neq y$. We finally represent our desires:

$$\neg wash \times wash$$
$$6{-}omelette \times 5{-}omelette \times 0{-}omelette$$

This logic program has the following 6 answer sets

$$S_1 = \{6{-}omelette, \neg wash, fresh, in{-}omelette\}$$
$$S_2 = \{0{-}omelette, \neg wash, rotten, in{-}omelette\}$$
$$S_3 = \{6{-}omelette, wash, fresh, in{-}cup\}$$
$$S_4 = \{5{-}omelette, wash, rotten, in{-}cup\}$$
$$S_5 = \{5{-}omelette, \neg wash, fresh, throw{-}away\}$$
$$S_6 = \{5{-}omelette, \neg wash, rotten, throw{-}away\}$$

The preference relation among answer sets is as follows: S_1 is the single maximally preferred answer set. S_5 and S_6 are preferred to S_2 and S_4 but incomparable to S_3. S_3 is preferred to S_4 but incomparable to S_5, S_6 and S_2. S_2 and S_4 are incomparable. Fig. 1 illustrates these relationships:

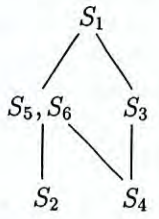


Fig. 1: Preferences among answer sets

Reasoning from maximally preferred answer sets in the example would yield $in{-}omelette$ as the alternative chosen by the agent. It is obvious that this amounts to an extremely optimistic attitude towards decision making which in the example amounts to assuming the egg will be fresh.

A pessimistic decision maker might choose the action whose worst outcome is most tolerable. In the example the answer sets containing $throw{-}away$, that is S_5 and S_6, are preferred to the least preferred answer set containing $in{-}omelette$, S_2, and to the least preferred answer set containing $in{-}cup$, S_4. Thus, a pessimistic decision maker would choose $throw{-}away$.

An extremely cautious strategy would prefer choice C_1 over choice C_2 if the least preferred answer set(s) containing C_1 are preferred to the most preferred answer set(s) containing C_2. This is a very strong requirement and in the egg example no action is preferred to another one according to this strategy.

Finally, we can compare answer sets statewise (where the state in the example is *rain* or $\neg rain$). C_1 is preferred over C_2 if for each state T the least preferred answer set(s) containing $C_1 \cup T$ are preferred to the most preferred answer set(s) containing $C_2 \cup T$.

Intuitively, S_2 in our example seems far less desirable than S_4 and both S_5 and S_6 less desirable than S_3. This is not reflected in our preference relation on answer sets. To express this it is necessary to represent preferences between sets of literals rather than single literals. Within our framework this can be done by introducing new atoms representing conjunctions of literals. However, it would probably be more elegant to apply orderd disjunction directly to sets of literals (read as the conjunction of these literals). Extending LPODs in such a way is straightforward.

Another natural idea would be to use numerical penalties. We can use integers for this and write, say:

$\neg wash\text{-}cup \times wash\text{-}cup$ (1)
$6\text{-}omelette \times 5\text{-}omelette$ (5) $\times$ $0\text{-}omelette$ (50)

The overall penalty for an answer set S is obtained by adding up the penalties for all rules, where the penalty of $c_1 \times c_2(n_2) \times \ldots \times c_k(n_k) \leftarrow body$ is 0 if *body* is not satisfied in S or $c_1 \in S$, n_j otherwise, where j is the smallest integer such that $c_j \in S$. The preference relation among answer sets is obtained through their overall penalty. In the example we would obtain the following overall penalties:

$S_1 : 0$	$S_3 : 1$	$S_5 : 5$
$S_6 : 5$	$S_4 : 6$	$S_2 : 50$

Choices could then be ordered on the basis of the average penalties of answer sets they contain. This strategy would thus choose $in\text{-}cup$.

Of course, many alternative strategies can be thought of. A further investigation is beyond the scope of this paper and left for future work.

Every approach to qualitative decision making has to combine preferences among outcomes of choices with a treatment of uncertainty. In our approach the preferences are described through ordered disjunction. But what about the uncertainty? Different possible states of the world are represented as different answer sets. As usual in nonmonotonic reasoning states of the world which are unnormal in some respect are totally disregarded (this is what McCarthy called jumping to conclusions). All states which have to be taken into account are considered plausible. Further distinctions between the generated answer sets are not possible. For instance, it is not possible to express, say, that *fresh* is more probable than *rotten* in the omelette example. If, however the possibility of *rotten* is negligeable and *fresh* is true by default we can make sure that only answer sets containing *fresh* are generated by using adequate rules. Our general qualitative attitude towards uncertainty can thus be described as: states are either negligeable or plausible; in the latter case no assumption about the degree of plausibility is made.

We are now in a position to analyze why the examples in Sect. 3 which were based on reasoning from most preferred answer sets worked out properly. The reason is that in these examples only one answer set for the different possible choices (which were left implicit) is generated. This means that optimistic, pessimistic and other kinds of LPOD based decision making coincide. In general, this is possible whenever there is enough knowledge to guarantee a single plausible state for each case at hand (as in the cinema example), or whenever all relevant literals are under the control of the agent (as in the cooking example).

6 Conclusion

In this paper we introduced a new connective to logic programming. This connective - called ordered disjunction - can be used to represent context dependent preferences in a simple and elegant way. Logic programming with ordered disjunction has interesting applications, in particular in design and configuration, and it can serve as a basis for qualitative decision models.

There are numerous papers introducing preferences to logic programming. For an overview of some of these approaches see the discussion in [6] or the more recent [20]. Only few of these proposals allow for context dependent preferences. Such preferences are discussed for instance in [4, 6]. The representation of the preferences in these papers is based on the introduction of names for rules, the explicit representation of the preference relation among rules in the logical language, and a sophisticated reformulation of the central semantic notion (answer set, extension, etc.) with a highly self-referential flavour. Alternative approaches [7, 12] are based on compilation techniques and make heavy use of meta-predicates in the logical language. Nothing like this is necessary in our approach. All we have

to do is use the degree of satisfaction of a rule to define a preference relation on answer sets directly.

Our approach is closely related to work in qualitative decision theory, for an overview see [8]. Poole [17] aims at a combination of logic and decision theory. His approach incorporates quantitative utilities whereas our preferences are qualitative. Interestingly, Poole uses a logic *without* disjunction whereas we *enhance* disjunction. In [3] a graphical representation, somewhat reminiscent of Bayes nets, for conditional preferences among feature values under the *ceteris paribus* principle is proposed, together with corresponding algorithms. *LPODs* are more general and offer means to reason defeasibly. Several models of qualitative decision making based on possibility theory are described in [9, 1]. They are based on certainty and desirability rankings. Some of them make strong commensurability assumptions with respect to these rankings. In a series of papers [13, 22], originally motivated by [2], the authors propose viewing conditional desires as constraints on utility functions. Intuitively, $D(a|b)$ stands for: the b-worlds with highest utility satisfy a. Our interpretation of ranked options is very different. Rather than being based on decision theory our approach can be viewed as giving a particular interpretation to the ceteris paribus principle.

In future work we plan to investigate application methodologies for logic programming with ordered disjunction. An answer set programming methodology for configuration tasks has been developed in a number of papers by Niemelä and colleagues at Helsinki University of Technology [21, 16]. We plan to study possibilities of combining this methodology with *LPODs*. Of course, the discussion of qualitative decision models in this paper was very preliminary. We plan to work this out in more detail in a separate paper. Finally, we are studying possibilities of implementing *LPODs* on top of the smodels system.

Acknowledgements

I would like to thank Salem Benferhat, Richard Booth, Tomi Janhunen, Ilkka Niemelä, Tommi Syrjänen and Leon van der Torre for helpful comments.

References

[1] S. Benferhat, D. Dubois, H. Fargier, and H. Prade. Decision, nonmonotonic reasoning and possibilistic logic. In J. Minker, editor, *Logic-Based Artificial Intelligence*. Kluwer Academic Publishers, 2000.

[2] C. Boutilier. Towards a logic for qualitative decision theory. In *Proc. Principles of Knowledge Representation and Reasoning, KR-94*, pages 75–86. Morgan Kaufmann, 1994.

[3] C. Boutilier, R.I. Brafman, H.H. Hoos, and D. Poole. Reasoning with conditional ceteris paribus preference statements. In *Proc. Uncertainty in Artificial Intelligence, UAI-99*, 1999.

[4] G. Brewka. Well-founded semantics for extended logic programs with dynamic preferences. *Journal of Artificial Intelligence Research*, 4:19–36, 1996.

[5] G. Brewka, S. Benferhat, and D. Le Berre. Qualitative choice logic. In *Proc. Principles of Knowledge Representation and Reasoning, KR-02*. Morgan Kaufmann, 2002. available as Ifl-Report under www.informatik.uni-leipzig.de/ brewka.

[6] G. Brewka and T. Eiter. Preferred answer sets for extended logic programs. *Artificial Intelligence*, 109:297–356, 1999.

[7] J. Delgrande, T. Schaub, and H. Tompits. Logic programs with compiled preferences. In *Proc. European Conference on Artificial Intelligence, ECAI-00*, pages 464–468. IOS Press, 2000.

[8] J. Doyle and R.H. Thomason. Background to qualitative decision theory. *AI Magazine*, 20(2):55–68, 1999.

[9] D. Dubois, D. Le Berre, H. Prade, and R. Sabbadin. Using possibilistic logic for modelingqualitative decision: Atms-based algorithms. *Fundamenta Informaticae*, 34:1–30, 1999.

[10] T. Eiter, N. Leone, C. Mateis, G. Pfeifer, and F. Scarcello. The kr system dlv: Progress report, comparisons and benchmarks. In *Proc. Principles of Knowledge Representation and Reasoning, KR-98*. Morgan Kaufmann, 1998.

[11] M. Gelfond and V. Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9:365–385, 1991.

[12] B. Grosof. Diplomat: Compiling prioritized rules into ordinary logic programs for e-commerce applications (intelligent systems demonstration abstract). In *Proc. AAAI-99*, 1999.

[13] J. Lang. Conditional desires and utilities - an alternative logical approach to qualitative decision theory. In *Proc. 12th European Conference on Artificial Intelligence, ECAI-96*, pages 318–322. Wiley and Sons, 1996.

[14] V. Lifschitz. Answer set programming and plan generation. available under www.cs.utexas.edu/users/vl/papers.html, 2001.

[15] I. Niemelä and P. Simons. Efficient implementation of the stable model and well-founded semantics for normal logic programs. In *Proc. 4th Intl. Conference on Logic Programming and Nonmonotonic Reasoning*. Springer Verlag, 1997.

[16] I. Niemelä and P. Simons. Extending the smodels system with cardinality and weight constraints. In J. Minker, editor, *Logic-Based Artificial Intelligence*. Kluwer Academic Publishers, 2000.

[17] D. Poole. The independent choice logic for modelling multiple agents under uncertainty. *Artificial Intelligence Journal*, 94(1-2):7–56, 1997.

[18] C. Sakama and K. Inoue. An alternative approach to the semantics of disjunctive logic programs and deductive databases. *Journal of Automated Reasoning*, 13:145–172, 1994.

[19] L. Savage. *The foundations of Statistics*. Dover, New York, 1954.

[20] T. Schaub and K. Wang. A comparative study of logic programs with preference. In *Proc. Intl. Joint Conference on Artificial Intelligence, IJCAI-01*, 2001.

[21] T. Soininen. *An Approach to Knowledge Representation and Reasoning for Product Configuration Tasks*. PhD thesis, Helsinki University of Technology, Finland, 2000.

[22] L. van der Torre and E. Weydert. Parameters for utilitarian desires in a qualitative decision theory. *Applied Intelligence*, 14:285–301, 2001.

Optimality Theory through Default Logic

Ph. Besnard*
IRIT-CNRS
Université Paul Sabatier
118 route de Narbonne
F-31062 Toulouse cedex
France
besnard@irit.fr

R. E. Mercer
Computer Science Department
Middlesex College
University of Western Ontario
London Ont. N6A 5B7
Canada
mercer@csd.uwo.ca

T. Schaub
Institut für Informatik
Universität Potsdam
Postfach 900327
D-14439 Potsdam
Germany
torsten@uni-potsdam.de

Abstract

Optimality Theory is an approach to linguistic problems which is based on rules with exceptions, resorting to a ranking among rules to resolve conflicts arising from competing rules.

In such a way, dealing with linguistic problems amounts to applying rules with exceptions: That is reasoning. A related issue is then about a formalization of the logic at work. An immediate candidate is Default Logic which is dedicated to reasoning from rules with exceptions. Moreover, there are versions of default logic with priorities.

We show that Default Logic is well-suited as a specification language capturing Optimality Theory and suggests that implementations of default logic can be applied to run experiments with grammatical interaction in the sense of Optimality Theory and beyond.

1 Introduction

Optimality Theory is a grammatical architecture that was invented in phonology [Prince & Smolensky 1993] but managed to spread into the other subdisciplines of linguistics quite successfully. In its standard version, Optimality Theory (cf [Kager 1999] for instance) is a representational rather than a derivational account of grammatical facts: It comprises of a set of grammatical constraints that evaluate the quality of candidate structures (or, say, representations), but it does not care how these candidate structures are generated.

In this respect, Optimality Theory only needs a component that decides which structures are compared with each other. The grammatical description of Optimality Theory is thus anchored with an input component. Inputs could be strings of sounds (in phonology), sets of morphemes (in morphology) or predicate-argument structures (in syntax). They are subjected to a GEN component that generates the candidate set on the basis of the input by very general grammatical processes. The candidate set is passed on to the EVAL component (EVAL stands for evaluation) that is in charge of selecting the optimal candidate according to the language at hand, using the grammatical constraints.

Optimality Theory assumes that the grammatical constraints are simple and universal (all languages work with the same set of constraints): "syllables have an onset", "sentences have a subject" are examples of what could be a constraint.

The grammatical constraints may imply incompatible requirements for certain structures. E.g., objects should follow the verb (compare *John loves Mary* with **John Mary loves*) but questions should begin with the question word (*how did she say this* vs. **she said this how*). For an object question, the two principles make different predictions (*what did she say* vs. **she said what*), and we see that the conflict between the two principles is resolved in favor of the question principle.

In Optimality Theory, the grammatical constraints are organized in a hierarchy. When two options compete, the one with the better violation profile wins: A candidate structure S is grammatical if and only if there is no competitor S' such that the highest constraint on which S and S' differ incurs less violations from S' than from S.

Conflict resolution is thus lexicographic: The numbers of violations of a candidate with respect to each constraint form a vector (constraints are considered in decreasing order).

*Contributed while at Institut für Informatik, Universität Potsdam.

Here is an example. The highest constraint is "the question word occurs first", the next highest constraint is "the verb group comes second", and the lowest constraint is "any non-subject item occurs after the subject". Consider the candidates: (1) **where she is now?*, (2) **she is where now?*, (3) *where is she now?*, (4) **where is now she?* The first constraint rules out (2) that is the only candidate to violate it, and then similarly for (1) with respect to the second constraint. The last constraint is violated thrice by (4) but only twice by (3) that is thus the best candidate (the fact that (2) does not violate the last constraint is irrelevant: (2) was already out of the competition).

In Optimality Theory, this is often visualized in a two-dimensional table as follows. Rewriting the candidates (1) to (4) as x_1 to x_4 and abbreviating the three constraints (from higher to lower) by c_1 to c_3, we obtain the configuration depicted in Table 1:

Table 1: A constraint tableau in Optimality Theory (optimal candidate: x_3).

	c_1	c_2	c_3
x_1: **where she is now?*		*	
x_2: **she is where now?*	*		
x_3: *where is she now?*			**
x_4: **where is now she?*			***

The ranking among the contraints is reflected by their decreasing importance from left to right. With the exception of grey cells, Table 1 displays to what extent each candidate (dis)agrees with each constraint. E.g., the violation of c_1 by x_2 is indicated by $*$ while the triple violation of c_3 by x_4 is represented by $***$. Grey cells denote data that are not taken into account (for instance, the cell $x_2 \times c_3$ is grey to reflect the aforementioned fact that no matter how well (2) fares with respect to the last constraint it is irrelevant because (2) is out by virtue of the first constraint).

Constraints in Optimality Theory turn out to be rules with exceptions: They are universal but they are *not* universally valid (e.g., there are syllables in English that have no onset). Indeed, Optimality Theory provides a methodology to apply rules with exceptions. As Default Logic was motivated by the need to deal with such rules, it seems natural to investigate whether Optimality Theory conforms with Default Logic.

Actually, Optimality Theory appears as reasoning non-monotonically not only about optimal candidates but also about candidates that fail to be optimal. Indeed, applying constraints in Optimality Theory leads to identify candidates as being suboptimal.

Unexpectedly enough, focusing on suboptimal candidates turns out to yield a direct characterization using defaults with priorities. This is interesting because the intuitive approach in Optimality Theory is more about the so-called winners than the so-called losers. Yet, losers and winners can easily be related in the obvious way using Default Logic.

2 Interaction in a linear hierarchy of constraints

Consider the case where the candidates involve no calculation: The list of candidates in final form $a, b, \ldots$ is available and the status (whether[1] absolute or relative to any other candidate) of each candidate with respect to every constraint is decided.

2.1 Harmonic parallel approach

Logic relations and defaults

The status of candidates with respect to constraints is to be encoded by means of $c_i defeats(a, b)$ relations over candidates and suboptimality (i.e., failure for a candidate to be a correct output) is encoded as $suboptimal(a)$.

The relation $c_i defeats(a, b)$ captures the case of non-binary constraints as well as the case of binary constraints (cf page 69 onwards in [Prince & Smolensky 1993]).

That suboptimality is determined from the status of the candidates is rendered through defaults:

$$(c_i) \quad \frac{c_i defeats(a, b) \ : \ \neg suboptimal(a)}{suboptimal(b)}$$

that should be read (after [Prince & Smolensky 1993] on page 74) as follows:

if b is less harmonic than a with respect to the constraint c_i then b is suboptimal unless a is itself suboptimal

While the set of all candidates is simply assumed to be finite, the set of constraints is assumed to be both finite and totally ordered: $C = \{c_1, c_2, \ldots\}$ where c_1 ranks highest then c_2 and so on. So, the above defaults for Optimality Theory are ordered accordingly:

$$c_i < c_j \quad \text{iff} \quad j \geq i \text{ and } i \neq j$$

[1] Depending on each particular constraint.

Hence, let us consider default logic with priorities (adapted from [Brewka 1994]):

Definition 1 *Let $(W, D, <)$ be a default theory where $<$ is a total order over D. Define*

- $E_0 = W$
- *for $n \geq 0$,*

$$E_{n+1} = \mathrm{Th}(E_n \cup \{cons(\delta)\})$$

where δ ranks highest among all defaults $\frac{\alpha:\beta}{\gamma}$ in D that satisfy the conditions[2] $\alpha \in E_n$ and $\neg\beta \notin E_n$ while $\gamma \notin \mathrm{Th}(E_n)$; if no such default exist, $\{cons(\delta)\} = \emptyset$.

Then, $E = \cup_{n\geq 0} E_n$ is the extension of $(W, D, <)$.

A correct output is then any candidate for which there exists an extension in which it is *not* proven suboptimal. [3]

Example 1 *The language is a dialect of Berber and the item to be parsed is /txznt/. Two constraints (the one denoted c_1 ranks higher than the one denoted c_2) are examined:*

(c_1) ONS *(Syllables must have onsets)*

(c_2) HNUC *(High sonority nucleus is more harmonic)*

For this example, [Prince & Smolensky 1993] consider the following three candidates:

.txź.ńt.
.tx́.zńt.
.t́x.zńt.

With a and b thus ranging over the set of candidates $\{.txź.ńt., .tx́.zńt., .t́x.zńt.\}$, the defaults are:

$$(c_1)\ \frac{c_1 defeats(a,b)\ :\ \neg suboptimal(a)}{suboptimal(b)}$$

(requiring $a \neq b$ yields six such defaults)[4]

and

$$(c_2)\ \frac{c_2 defeats(a,b)\ :\ \neg suboptimal(a)}{suboptimal(b)}$$

(requiring $a \neq b$ yields six such defaults)[4]

Taking .txź.ńt. to be less behaved than the other two candidates regarding ONS,

$$c_1 defeats(.tx́.zńt., .txź.ńt.) \in W,$$

$$c_1 defeats(.t́x.zńt., .txź.ńt.) \in W,$$

and taking .t́x.zńt. to be less behaved than .tx́.zńt. regarding HNUC,

$$c_2 defeats(.tx́.zńt., .t́x.zńt.) \in W,$$

the resulting default theory has exactly one extension:

$$\mathrm{Th}(W \cup \{suboptimal(.txź.ńt.), suboptimal(.t́x.zńt.)\})$$

The only candidate which is not proven suboptimal in the extension is .tx́.zńt., this is the unique optimal candidate (the correct output).

The outcome would be the same, should $c_1 defeats(.t́x.zńt., .txź.ńt.)$ not be in W. This is an instance of the property which ensures that those $c_i defeats(a, \cdot)$ items are enough to get the correct output(s), where a ranges over the optimal candidate(s) while c_i ranges over all constraints such that $i > j$ where j is the highest ranking constraint violated by the optimal candidate(s).

It would also make no difference if HNUC *had .t́x.zńt. less behaved than .txź.ńt.*

Although the underlying idea is the same, using $c_i defeats$ relations is more general than the mark approach in [Prince & Smolensky 1993] (cf page 68 onwards) which can be captured (as $c_i defeats$ can be defined in virtually any way in W) as follows.

$$\forall a, b, m_a, m_b \quad Ons(a, m_a) \wedge Ons(b, m_b) \wedge m_a \succ_{Ons} m_b \rightarrow c_1 defeats(a, b)$$

$$\forall a, b, m_a, m_b \quad FM(m_a) \succ_{Ons} FM(m_b) \rightarrow m_a \succ_{Ons} m_b$$

$$\forall a, b, m_a, m_b \quad (FM(m_a) \approx_{Ons} FM(m_b) \wedge Rest(m_a) \succ_{Ons} Rest(m_b)) \rightarrow m_a \succ_{Ons} m_b$$

$Ons(c, m_c)$ means that ONS assigns a list m_c of violation marks to every candidate c. Then, $FM(m_a) \succ_{Ons}$

[2] It is alos possible to require for δ to be such that $\neg jus(\delta') \notin \mathrm{Th}(E_n \cup \{cons(\delta)\})$ for all defaults δ' selected prior to δ. (i.e., for some $m < n$, δ' ranking highest such that $\alpha' \in E_m$ and $\neg\beta' \notin E_m$ while $\gamma' \notin \mathrm{Th}(E_m)$).

[3] The reader should not worry about matters of uniqueness and non-uniqueness, which are unproblematic even though the existence of exactly one extension for $(W, D, <)$ does not preclude the existence of more than one optimal output. By contrast, it is an absolute requirement that the order of defaults be total and this means that *instances* of the *same* constraint c_i have to be ordered (or $c_i defeats(a, b)$ must be antisymmetric, as well as irreflexive of course).

[4] An arbitrary total order is assumed among these six defaults (cf the previous footnote).

$FM(m_b)$ always holds when m_a is empty while m_b is not. Also, $FM(m_a) \approx_{Ons} FM(m_b)$ always holds when m_a and m_b are both non-empty. Lastly, $Rest(m_c)$ denotes the list m_c deprived from its first mark.

$$\forall a, b, m_a, m_b \quad Hnuc(a, m_a) \wedge Hnuc(b, m_b) \wedge m_a \succ_{Hnuc} m_b \rightarrow c_2 defeats(a, b)$$

$$\forall a, b, m_a, m_b \quad FM(m_a) \succ_{Hnuc} FM(m_b) \rightarrow m_a \succ_{Hnuc} m_b$$

$$\forall a, b, m_a, m_b \quad (FM(m_a) \approx_{Hnuc} FM(m_b) \wedge Rest(m_a) \succ_{Hnuc} Rest(m_b)) \rightarrow m_a \succ_{Hnuc} m_b$$

Similarly to ONS, this assumes HNUC ([Prince & Smolensky 1993] page 72) to provide:

- a list of marks m_c sorted from most to least sonorous[5] for every candidate c
  (as is required in the first formula)
- an assessment for every pair of marks where one is more sonorous than the other
  (as is required in the second formula)
- an assessment for every pair of equally sonorous marks
  (as is required in the third formula)

All this assumes further that W contains the usual axioms for equality (for instance, $Rest(Rest(Rest(m_c)))$ must be provably equal with the list obtained by deleting the first three marks in m_c). There are other formulations, getting rid of FM and $Rest$, such that the axioms of equality can be dispensed with.

As to a different kind of an example, consider NONFINALITY in a version where it is a binary constraint that does not apply multiply ([Prince & Smolensky 1993] page 43)

$$\forall a, b \quad Nonfinality(a) \wedge \neg Nonfinality(b) \rightarrow c_i defeats(a, b)$$[6]

where i is the rank of NONFINALITY and for every candidate c, $Nonfinality(c)$ holds when the head foot of the prosodic word is not final.

2.2 Harmonic sequential approach

Now, it is no longer the case that the candidates are available right from the start. They are to form step by step, some of them getting discarded even before they develop to final form (this is the difference with the parallel approach).

Logic relations and defaults

Generation of (partial) candidates is encoded by means of the $gen(a, b)$ relation indicative of a derivation step from a to b. That a representation in the derivation currently counts as a candidate is encoded by means of $current(a)$. Steps in generation are rendered through the default[7]

$$(Gen) \quad \frac{current(a) \wedge gen(a, b) \ : \ \neg suboptimal(b)}{current(b)}$$

that reads

if a counts as a candidate and there is a step in derivation turning a into b
then b counts as a candidate unless b is suboptimal

Importantly, this default[8] ranks lower than all the ones representing constraints:

$$(c_i) \quad \frac{c_i defeats(a, b) \ : \ \neg suboptimal(a)}{suboptimal(b)}$$

These defaults are exactly as in the parallel approach[9] (thus making the sequential approach a generalization of it).

$$\forall x \ input(x) \rightarrow current(x) \in W$$

If that is desired, a more detailed formulation is possible where changed elements are explicited as it only takes including in W the following formulas for all relevant a, b, c:

$$change(a, e_a)$$

$$(gen(a, b) \wedge gen(a, c) \wedge change(b, e_b) \wedge change(c, e_c) \wedge c_i defeats(e_b, e_c)) \rightarrow c_i defeats(b, c)$$

The former formulas indicate that e_a is the changed element in a and the latter formulas state that defeat among candidates is ruled by defeat among changed elements.

[5] Harmonic, if we identify the mark with the nucleus it stands for (p. 72 [Prince & Smolensky 1993]).

[6] This formula makes $c_i defeats(a, b)$ to be antisymmetric. Also, if all candidates fail NONFINALITY then $c_i defeats(a, b)$ holds for no a and b.

[7] Singular is used here, although improperly, as the specific values of a and b are unimportant.

[8] Again, singular is improperly used. As a further motive, it can be pointed out that the highest ranking among all defaults of that form ranks lower than any default representing a constraint.

[9] Even though they need *not* obey the same order: see footnote 49 in [Prince & Smolensky 1993].

Digression. A subtlety is that antisymmetry is required for $c_i defeats$ if its arguments are changed elements (i.e., $c_i defeats(e_b, e_c)$ and $c_i defeats(e_c, e_b)$ are incompatible). Such a requirement is *not* needed for $c_i defeats$ if its arguments are partial candidates (that is, of concern here is symmetry through $c_i defeats(b, c)$ and $c_i defeats(c, b)$). How can this be? Well, b and c differ from each other by at least one changed element. Now, one of these must occur before the other and therefore *rules out* the other candidate.

Example 2 *In the same dialect of Berber, the item to be parsed is /ratlult/ so that*

$$input(ratlult) \in W$$

The following formulas are also in W:

$$gen(ratlult, \{rA\}tlult)$$
$$gen(\{rA\}tlult, \{rA\}t\{lu\}lt)$$
$$gen(\{rA\}tlult, \{rAt\}lult)$$

Distinctively from the parallel approach (cf [Prince & Smolensky 1993]), the ordering of constraints[10] *is:*

(c_1) –Cod *(Syllables do not have codas)*

(c_2) Hnuc *(Higher sonority nucleus is more harmonic)*

such that

$$c_2 defeats(\{rAt\}lult, \{rA\}t\{lu\}lt) \in W$$

$$c_1 defeats(\{rA\}t\{lu\}lt, \{rAt\}lult) \in W$$

The resulting default theory has exactly one extension:

$$\mathrm{Th}(W \cup X)$$

where X consists of the following formulas

$$current(\{rAt\}lult)$$
$$suboptimal(\{rAt\}lult)$$
$$current(\{rA\}t\{lu\}lt)$$

The above data are partial, they only describe an initial part of the process of selecting the correct output. Yet, they suffice to indicate that $\{rA\}t\{lu\}lt$ is the line to be developed but not $\{rAt\}lult$. Should more data be taken into account, the result goes further.

[10] Where –Cod is introduced on page 34 in [Prince & Smolensky 1993].

If the changed elements are to be explicited, the following formulas have to be in W:

$$gen(ratlult, \{rA\}tlult) \rightarrow change(\{rA\}tlult, \{rA\})$$

$$gen(\{rA\}tlult, \{rA\}t\{lu\}lt) \rightarrow change(\{rA\}t\{lu\}lt, \{lu\})$$

$$gen(\{rA\}tlult, \{rAt\}lult) \rightarrow change(\{rAt\}lult, \{rAt\})$$

$$c_2 defeats(\{rAt\}, \{lu\})$$

$$c_1 defeats(\{lu\}, \{rAt\})$$

$$c_2 defeats(\{rAt\}, \{lu\}) \rightarrow c_2 defeats(\{rAt\}lult, \{rA\}t\{lu\}lt)$$

$$c_1 defeats(\{lu\}, \{rAt\}) \rightarrow c_1 defeats(\{rA\}t\{lu\}lt, \{rAt\}lult)$$

Of course, the outcome is exactly the same.

3 Interaction in an arbitrary hierarchy of constraints

3.1 Harmonic parallel approach

Footnote 31 in [Prince & Smolensky 1993] acknowledges the possibility that a grammar should recognize nonranking pairs of constraints (although the authors make it clear that they found no evidence of crucial nonranking). Non-linear hierarchies of constraints may further seem invited from the view stated on page 88 in [Prince & Smolensky 1993], that a category of constraints dominated by others fixed in superordinate position may be relatively ranked in any dominance order in a particular language.[11]

Logic relations and defaults

Logic relations and defaults are still as described above. The set of constraints is still finite but need no longer to be totally ordered: Therefore, it is still the case that $c_i < c_j$ for some c_i and c_j in $C = \{c_1, c_2, \ldots\}$ but this no longer relates to $j \geq i$. Accordingly, let us now consider default logic with non-linear priorities (adapted from [Baader & Hollunder 1995]):

Definition 2 *Let $(W, D, <)$ be a default theory where $<$ is a partial order over D. Then, E is an extension for $(W, D, <)$ iff $E = \cup_{n \geq 0} E_n$ such that*

- $E_0 = W$

[11] Not: ... are relatively ranked in some dominance order

- *for* $n \geq 0$,

$$E_{n+1} = \mathrm{Th}(E_n \cup Cons(\{\delta_1, \ldots, \delta_k\}))$$

where $\delta_1 \ldots \delta_k$ *have priority*[12] *among all defaults in* D *that are active*[13] *in* E_n

An example [Prince & Smolensky 1993] not requiring a linear hierarchy of constraints:

Example 3 *The language is Latin. Three constraints are considered:*

(c_1) FTBIN *(Feet are binary at some level of analysis)*

(c_2) LX≈PR *(A member of a certain morphological category corresponds to a* PRWD*)*

(c_3) NONFINALITY *(The head foot of the prosodic word is not final)*

Consider the following three candidates:

$$.a.qua.$$
$$(\acute{a})\, L$$
$$(\acute{a}.qua)$$

With u *and* v *ranging over the set of candidates* $\{.a.qua., (\acute{a})\, L, (\acute{a}.qua)\}$*, defaults are:*

$$(c_1)\ \frac{c_1 defeats(u,v)\ :\ \neg suboptimal(u)}{suboptimal(v)}$$

$$(c_2)\ \frac{c_2 defeats(u,v)\ :\ \neg suboptimal(u)}{suboptimal(v)}$$

$$(c_3)\ \frac{c_3 defeats(u,v)\ :\ \neg suboptimal(u)}{suboptimal(v)}$$

c_3 *ranks lowest of all but neither* c_1 *ranks higher than* c_2 *nor* c_2 *ranks higher than* c_1*. The other data are*

$$\forall u\ c_1 defeats(u, (\acute{a})\, L) \in W$$
$$\forall u\ c_2 defeats(u, .a.qua.) \in W$$
$$\forall u\ c_3 defeats(u, (\acute{a}.qua)) \in W$$

The outcome is that the default theory at hand has a single extension

$$E = \mathrm{Th}(W \cup \{suboptimal(\acute{a})\, L), suboptimal(.a.qua.)\})$$

The optimal candidate is $(\acute{a}.qua)$*, there was no need for a linear ordering of constraints.*

The existence of a unique optimal candidate coincides with the existence of a greatest element in the induced[14] ordering.

[12] The precise definition depends on what variant of default logic is selected.

[13] Again, there is a choice of definitions here and $Cons(\{\delta_1, \ldots, \delta_k\}) = \emptyset$ by convention when no default is active.

[14] Induced by c_i's ordering.

4 Logic programming

As for harmonic parallelism, the case of a linear hierarchy of constraints is efficiently tackled using logic programming, in a guise e.g. [Delgrande, Schaub, & Tompits 2000] where preferences can be expressed so that preemption among constraints is resolved:

```
suboptimal(b) :-
            name(i(a,b)),
            c-i-defeats(a,b),
            not suboptimal(a).
```

For all $j < k$ (i.e., constraint j ranks higher than constraint k), one need the clauses:

```
(j(_,_) < k(_,_)).
```

Example 4 *(Back to the Berber example.) The predicate* `missing-ons(c,r)` *assigns the missing onsets in the candidate* c *to the variable* r *whereas* `s-nuclei(c,r)` *assigns the nuclei in the candidate* c *(sorted from most to least sonorous) to the variable* r*.*

```
c-1-defeats(a,b) :-
                missing-ons(a,p),
                missing-ons(b,q),
                better-ons(p,q).
c-2-defeats(a,b) :-
                s-nuclei(a,p),
                s-nuclei(b,q),
                better-hnuc(p,q).
better-ons([],[z|t]).
better-ons([x|r],[y|s]) :-
                          better-ons(r,s).
better-hnuc([x|r],[y|s]) :-
                           more-sonorous(x,y).
better-hnuc([x|r],[x|s]) :-
                           better-hnuc(r,s).
```

Entering the data,

```
missing-ons(".txZ.Nt.",[2nd-syll]).
missing-ons(".tX.zNt.",[]).
missing-ons(".Tx.zNt.",[]).
s-nuclei(".txZ.Nt.",["N","Z"]).
s-nuclei(".tX.zNt.",["N","X"]).
s-nuclei(".Tx.zNt.",["N","T"]).
more-sonorous("X","T").
```

the outcome is that the goal `suboptimal(".txZ.Nt.")` *succeeds as does the goal* `suboptimal(".Tx.zNt.")` *while the goal* `suboptimal(".tX.zNt.")` *fails, meaning that* `".tX.zNt."` *is optimal.*

5 Summary of method

The method introduced here can be described as follows:

1. *Include in W the status of candidates with respect to constraints:*

$$C_\lambda \to c_i defeats(\varphi, \sigma)$$
$$\vdots$$

where the conditions C_λ are basically tautological in the parallel approach[15]
—in which case the above formulas then simplify to $c_i defeats(\varphi, \sigma)$.

2. *Include in D the effect of constraints towards suboptimality:*

$$\frac{c_i defeats(\varphi, \sigma) \ : \ \neg suboptimal(\varphi)}{suboptimal(\sigma)}$$

3. *Specify the ordering $<$ between constraints.*

4. *In view of the resulting default theory $(W, D, <)$, look for any candidate c such that $suboptimal(c)$ is not in an extension. Such a candidate is a correct output.*

As we have seen, the above scheme can be extended when dealing with special cases such as the sequential approach and so on:

$$gen(\chi, \kappa)$$
$$\vdots$$
$$change(\kappa, \epsilon_\kappa)$$
$$\vdots$$
$$gen(\varphi, \sigma) \wedge gen(\varphi, \rho) \wedge change(\sigma, \epsilon_\sigma) \wedge change(\rho, \epsilon_\rho) \wedge c_i defeats(\epsilon_\sigma, \epsilon_\rho) \to c_i defeats(\sigma, \rho)$$
$$\vdots$$

Expressing Optimality Theory in default logic without priority is also possible, with defaults that are no more complex than above. However, modularity as well as independence of defaults from constraint ranking are lost.

Acknowledgements

This research has been partially supported by the German Science Foundation (DFG) under grant FOR 375/1-1, TP C.

[15] Reduplication of the mark approach is a good example showing that the conditions C_λ can take various forms, all of them amount to tautologies.

References

[Baader & Hollunder 1995]
F. Baader and B. Hollunder.
Priorities on defaults with prerequisites. *Journal of Automated Reasoning* 15:41–68, 1995.

[Brewka 1994] G. Brewka.
Adding priorities and specificity to default logic. In L. Pereira and D. Pearce, editors, *4th European Workshop on Logics in Artificial Intelligence (JELIA-94)*, pp. 247–260, 1994.

[Delgrande, Schaub, & Tompits 2000]
J. Delgrande, T. Schaub, and H. Tompits.
A compiler for logic programs with preferences. 2000.
http://www.cs.uni-potsdam.de/~torsten/plp

[Delgrande, Schaub, & Tompits 2000]
J. Delgrande, T. Schaub, and H. Tompits.
A compiler for ordered logic programs. *8th Workshop on Non-Monotonic Reasoning (NMR-2000)*, Breckenridge, CO, 2000.

[Delgrande, Schaub, & Tompits 2000]
J. Delgrande, T. Schaub, and H. Tompits.
Logic programs with compiled preferences. *14th European Conference on Artificial Intelligence (ECAI-00)*, Berlin, Germany, 2000.

[Kager 1999] R. Kager.
Optimality Theory. Cambridge Textbooks in Linguistics. Cambridge Univ. Press, 1999.

[Prince & Smolensky 1993]
A. Prince and P. Smolensky.
Optimality Theory. Tech Report, Rutgers U., New Brunswick (NJ), and Computer Science Department, U. of Colorado, Boulder. 1993.

Defeasible Logic with Dynamic Priorities

Grigoris Antoniou
Dept. Of Computer Science, Univ. of Bremen, Germany
ga@tzi.de

Abstract

Defeasible logic is a nonmonotonic reasoning approach based on rules and priorities. Its design supports efficient implementation, and it shows promise to be successfully deployed in applications.

So far only static priorities have been used, provided by an external superiority relation. In this paper we show how dynamic priorities can be integrated, where priority information is obtained from the deductive process itself. Dynamic priorities have been studied for other related reasoning systems, such as default logic and argumentation. We define a proof theory, study its formal properties, and provide an argumentation semantics.

1 INTRODUCTION

Defeasible reasoning is a nonmonotonic reasoning approach in which the gaps due to incomplete information are closed through the use of defeasible rules that are usually appropriate. Defeasible logics were introduced and developed by Nute over several years (Nute 1994). These logics perform defeasible reasoning, where a conclusion supported by a rule might be overturned by the effect of another rule. Roughly, a proposition p can be defeasibly proved ($+\partial p$) only when a rule supports it, and it has been demonstrated that no applicable rule supports $\neg p$; this demonstration makes use of statements $-\partial q$ which mean intuitively that an attempt to prove q defeasibly has failed finitely. These logics also have a monotonic reasoning component, and a priority on rules.

This family of approaches has recently attracted considerable interest. Its use in various application domains has been advocated, including the modelling of regulations and business rules (Grosof et al. 1999), modelling of contracts (Reeves et al. 1999), legal reasoning (Prakken 1997) and agent negotiations (Governatori et al. 2000). Also, defeasible reasoning (in the form of courteous logic programs (Grosof 1997)) provides the foundation for IBM's Business Rules Markup Language and for planned W3C activities. Therefore defeasible reasoning is arguably one of the most successful subareas in nonmonotonic reasoning as far as applications and integration to mainstream IT is concerned.

Recent theoretical work on defeasible logics has: (i) established some relationships to other logic programming approaches without negation as failure (Antoniou, Billington and Maher 2000); (ii) analysed the formal properties of these logics (Antoniou et al. 2001; Maher 2001), and (iii) has delivered efficient implementations (Maher et al. 2000).

So far defeasible logics had *fixed priorities*, given in the form of an external superiority relation. But *dynamic priorities*, where priorities are derived within the logic, have been considered in related areas, such as for extended logic programs (Brewka 1996; Brewka and Eiter 1999) and argumentation (Prakken and Sartor 1997). Reasons for considering dynamic priorities include the following:

- *Case-based priorities*: Sometimes preference of a rule over another is conditional on a property holding. In such cases it is natural to write $a \rightarrow r_1 > r_2$ and apply the priority information whenever a has been derived.
- *Defeasible priorities*: Sometimes case-based priorities do not apply always, but will be subject to "attacks" from other arguments. This happens, for example, in the legal domain, when different prioritisation principles can be applied (specificity, hierarchy, recency). In such cases the prin-

ciples should be stated conditionally using defeasible rules, and may be defeated by stronger principles.

- We expect dynamic priorities to be particularly useful for intelligent agents working on the Web. Such agents will frequently need to make decisions about the reliability of information, especially to resolve conflicts among different sources.

In this paper we show how dynamic priorities can be incorporated in the framework of defeasible logics. We consider two variations: one in which cyclic priorities may emerge, and one in which such cycles are avoided. We study both approaches, and derive various theoretical results concerning their proof theory and semantics.

2 THE APPROACH

2.1 A LANGUAGE FOR DEFEASIBLE REASONING

A defeasible theory (a knowledge base in defeasible logic) consists of three different kinds of knowledge: strict rules, defeasible rules, and a superiority relation. (Fuller versions of defeasible logic also have facts and defeaters, but (Antoniou et al. 2001) shows that they can be simulated by the other ingredients).

Strict rules are rules in the classical sense: whenever the premises are indisputable (e.g. facts) then so is the conclusion. An example of a strict rule is "Emus are birds". Written formally:

$$emu(X) \rightarrow bird(X).$$

Defeasible rules are rules that can be defeated by contrary evidence. An example of such a rule is "Birds typically fly"; written formally:

$$bird(X) \Rightarrow flies(X).$$

The idea is that if we know that something is a bird, then we may conclude that it flies, *unless there is other, not inferior, evidence suggesting that it may not fly.*

In the defeasible logics defined so far in the literature, an external, acyclic *superiority relation* among rules is used to define priorities among rules, that is, where one rule may override the conclusion of another rule. For example, given the defeasible rules

$$\begin{array}{lrl} r: & bird(X) & \Rightarrow flies(X) \\ r': & brokenWing(X) & \Rightarrow \neg flies(X) \end{array}$$

which contradict one another, no conclusive decision can be made about whether a bird with broken wings can fly. But if we introduce a superiority relation $>$ with $r' > r$, with the intended meaning that r' is strictly stronger than r, then we can indeed conclude that the bird cannot fly.

2.2 FORMAL DEFINITION

In this paper we restrict attention to essentially propositional defeasible logic. Rules with free variables are interpreted as rule schemas, that is, as the set of all ground instances; in such cases we assume that the Herbrand universe is finite. We assume that the reader is familiar with the notation and basic notions of propositional logic. If q is a literal, $\sim q$ denotes the complementary literal (if q is a positive literal p then $\sim q$ is $\neg p$; and if q is $\neg p$, then $\sim q$ is p).

Rules are defined over a *language* (or *signature*) Σ, the set of propositions (atoms) and labels that may be used in the rule.

A *rule* $r : A(r) \hookrightarrow C(r)$ consists of its unique *label* r, its *antecedent* $A(r)$ ($A(r)$ may be omitted if it is the empty set) which is a finite set of literals, an arrow $\hookrightarrow$ (which is a placeholder for concrete arrows to be introduced in a moment), and its *head* (or *consequent* or *conclusion*) $C(r)$ which is a literal. In writing rules often we omit set notation for antecedents. There are two kinds of rules, each represented by a different arrow. Strict rules use $\rightarrow$ and defeasible rules use $\Rightarrow$.

Given a set R of rules, we denote the set of all strict rules in R by R_s. $R[q]$ denotes the set of rules in R with consequent q.

A *defeasible theory* D is a finite set of rules R.

2.3 DEFEASIBLE LOGIC WITH DYNAMIC PRIORITIES

The approach we take is as follows:

- For every rule r we associate a unique label $label(r)$.
- In the language language we introduce a binary predicate $\succ$ which applies to rule labels. The intuitive meaning of $label(r_1) \succ label(r_2)$ is that rule r_1 is stronger than rule r_2.
- Literals including $\succ$ may appear anywhere in a rule: as antecedents or as consequents, negated or not negated. For example, we can write:

$$a \Rightarrow label(r_1) \succ label(r_2)$$
$$\neg label(r_1) \succ label(r_2) \Rightarrow c$$

- A defeasible theory is just a set of rules. We don't use external priority information, but only the priorities derived from the theory.

We consider two variants of defeasible logic with dynamic priorities. The first one is simpler and allows cyclic priority information to be derived (thus resulting in the possibility of inconsistencies). The second approach checks for potential cycles in the priorities, and rules them out.

According to the first approach, it is the user's responsibility to ensure that cycles do not occur (just as it is the programmer's responsibility to avoid dividing by zero). The second approach prevents cycles from occurring, in anticipation of its use in distributed environments, where cycles can easily occur when putting together knowledge from various sources.

3 DDL1: DYNAMIC DEFEASIBLE LOGIC WITHOUT CYCLE CHECKING

3.1 PROOF THEORY

A *conclusion* of a defeasible theory D consists of a sign, a tag, and a literal. The + sign denotes provability, and the − sign denotes finite failure to prove. The tags we use are those introduced for the family of defeasible logics in (Antoniou et al. 2000): Δ denotes strict provability, and ∂ denotes defeasible provability. More tags will be introduced soon. So, a conclusion $+\partial q$ is intended to mean that q is defeasibly provable in D. And $-\Delta$ means that q is provably not strictly provable.

Provability is defined below. It is based on the concept of a *derivation* (or *proof*) in $D = R$. A derivation is a finite sequence $P = P(1), \ldots, P(n)$ of tagged literals satisfying the following conditions. The conditions are essentially inference rules phrased as conditions on proofs.$P(1..i)$ denotes the initial part of the sequence P of length i.

$+\Delta$: If $P(i+1) = +\Delta q$ then
$\exists r \in R_s[q]\ \forall a \in A(r) : +\Delta a \in P(1..i)$

That means, to prove $+\Delta q$ we need to establish a proof for q using strict rules only. This is a deduction in the classical sense – no proofs for the negation of q need to be considered (in contrast to defeasible provability below, where opposing chains of reasoning must be taken into account, too).

$-\Delta$: If $P(i+1) = -\Delta q$ then
$\forall r \in R_s[q]\ \exists a \in A(r) : -\Delta a \in P(1..i)$

The definition of $-\Delta$ is the so-called *strong negation* of $+\Delta$: normal negation rules like De-Morgan rules are applied to the definition, + is replaced by −, and vice versa. Therefore in the following we may omit giving inference conditions of both + and −.

$+\partial$: If $P(i+1) = +\partial q$ then either
(1) $+\Delta q \in P(1..i)$ or
(2) (2.1) $\exists r \in R[q]\ \forall a \in A(r) : +\partial a \in P(1..i)$ and
(2.2) $-\Delta \sim q \in P(1..i)$ and
(2.3) $\forall s \in R[\sim q]$ either
(2.3.1) $\exists a \in A(s) : -\partial a \in P(1..i)$, or
(2.3.2) $\exists t \in R[q]$ such that
$\forall a \in A(t) + \partial a \in P(1..i)$ and
$+\partial(label(t) \succ label(s)) \in P(1..i)$

Intuitively: to show that q is provable defeasibly we have two choices: (1) We show that q is already definitely provable; or (2) we need to argue using the defeasible part of D as well. In particular, we require that there must be a strict or defeasible rule with head q which can be applied (2.1). But now we need to consider possible "counterattacks", that is, reasoning chains in support of $\sim q$. To be more specific: to prove q defeasibly we must show that $\sim q$ is not definitely provable (2.2). Also (2.3) we must consider the set of all rules which are not known to be inapplicable and which have head $\sim q$. Essentially each such rule s attacks the conclusion q. For q to be provable, each such rule s must have been established as non-applicable (2.3.1). Alternatively, there is an applicable rule with head q which is stronger than the rule s, according to the priority information derived so far.

We note that the only difference to the corresponding inference condition in the "classical" defeasible logic (Antoniou et al. 2001) is the last line: we use the priority derived so far, instead of an external superiority relation.

The inference condition $-\partial$ is defined in a similar way, as the strong negation of the $+\partial$ inference condition.

Also we note that the conclusion q may be a literal $label(r_1) \succ label(r_2)$.

Example 3.1 Consider the theory

$r_1 :\Rightarrow p$ $\quad$ $r_3 :\Rightarrow q$
$r_2 : p \Rightarrow r_3 \succ r_4$ $\quad$ $r_4 :\Rightarrow \neg q$

(in examples the notation $r : A \Rightarrow C$ implies that r is the unique label of the rule). We can derive $+\partial p, +\partial(r_3 \succ r_4)$, and $+\partial q$.

3.2 AMBIGUITY PROPAGATION

In (Antoniou et al. 2000) we discussed the property of ambiguity propagation, noting that the original defeasible logic was ambiguity blocking. A preference for ambiguity blocking or ambiguity propagating behaviour is one of the properties of non-monotonic inheritance nets over which intuitions can clash (Touretzky et al. 1987). Ambiguity propagation results in fewer conclusions being drawn, which might make it preferable when the cost of an incorrect conclusion is high. For these reasons an ambiguity propagating version of DL is of interest.

The solution to achieve ambiguity propagation behaviour is to separate the invalidation of a counterargument from the derivation of $-\partial$ tagged literals. We do so by introducing a third level of provability (besides definite and defeasible provability), called *support* and denoted by $\int$. Intuitively, a literal p is supported if there is a chain of reasoning that would lead us to conclude p in the absence of conflicts.

Thus in our modification of the $+\partial$ condition, a counterargument may be disregarded only if a literal in its body is known not to be supported ($-\int$), which is stronger than the previous condition that made it sufficient for a literal in the body not be defeasibly provable. First we modify the $+\partial$ condition (the $-\partial$ condition is modified accordingly following the Principle of Strong Negation).

$+\partial_{ap}$: If $P(i+1) = +\partial_{ap}q$ then either
(1) $+\Delta q \in P(1..i)$ or
(2) (2.1) $\exists r \in R[q]\ \forall a \in A(r) : +\partial_{ap}a \in P(1..i)$ and
(2.2) $-\Delta \sim q \in P(1..i)$ and
(2.3) $\forall s \in R[\sim q]$ either
(2.3.1) $\exists a \in A(s) : -\int a \in P(1..i)$ or
(2.3.2) $\exists t \in R[q]$ such that
$\forall a \in A(t) : +\partial_{ap}a \in P(1..i)$ and
$+\partial_{ap}(label(t) \succ label(s)) \in P(1..i)$

Next we define the inference conditions for support. If we ignore the superiority relation we could define it simply as a simple chain. However, in situations where two conflicting rules can be applied and one rule is inferior to another, the inferior rule should not be counted as supporting its conclusion. Thus we refine the inference rule as follows ($-\int$ is defined accordingly):

$+\int$: If $P(i+1) = +\int q$ then either
$+\Delta q \in P(1..i)$ or
$\exists r \in R[q]$ such that
$\forall a \in A(r) : +\int a \in P(1..i)$, and
$\forall s \in R[\sim q]$ either
$\exists a \in A(s) : -\partial_{am}a \in P(1..i)$ or
$+\partial(label(r) \succ label(s)) \in P(1..i)$

3.3 PROPERTIES

First we make the following observation: if $\succ$ does not occur in a defeasible theory, then DDL1 behaves exactly as the classical defeasible logic.

Theorem 3.1 (Coherence) *It is impossible to derive both $+tag\ q$ and $-tag\ q$ from a defeasible theory D, for $tag \in \{\Delta, \partial, \partial_{ap}, \int\}$.*

Theorem 3.2 (Consistency) *Let the set $\{label(r_1) \succ label(r_2) \mid D \vdash +\partial(label(r_1) \succ label(r_2))\}$ not contain a cycle. Then $+\partial p$ and $+\partial\neg p$, or $-\partial_{ap}p$ and $-\partial_{ap}p$, only if $+\Delta p$ and $-\Delta p$.*

In other words, an inconsistency in the defeasible part can only occur if the same inconsistency already occurs in the strict part. This property does not hold if we do not assume the extra condition, as the following example shows.

Example 3.2 Consider the theory

$r_1 : \Rightarrow r_3 \succ r_4$ $\quad r_3 : \Rightarrow p$
$r_2 : \Rightarrow r_4 \succ r_3$ $\quad r_4 : \Rightarrow \neg p$

We have $-\Delta p$ and $-\Delta\neg p$, but $+\partial p$ and $+\partial\neg p$. It is instructive to see the difference between $r_4 \succ r_3$ and $\neg r_3 \succ r_4$. If we modify rule r_2 to $\Rightarrow \neg r_3 \succ r_4$, then we can derive $-\partial p$ and $-\partial\neg p$.

The following theorem says that there exists a chain of increasing expressive power among several tags, similar to the situation in defeasible logic without dynamic priorities.

Theorem 3.3 $+\Delta \subset +\partial_{ap} \subset +\partial \subset +\int$.

For each inclusion there are defeasible theories in which the inclusion is strict.

4 DDL2: DYNAMIC DEFEASIBLE LOGIC WITH CYCLE CHECKING

4.1 PROOF THEORY

Define $Chains(label(r_1) \succ label(r_2))$ to be the set of all sequences of strict or defeasible rules, without multiple occurrences $C = (r^{(1)}, \ldots, r^{(n)})$ such that

- $C(r^{(i)}) = label(r'_i) \succ label(r'_{i+1})$, for all $i \in \{1,\ldots,n-1\}$.
- $r'_1 = r_1$, $r'_n = r_2$.

Intuitively, *Chains* collects all chains of priority statements, starting with $label(r_1)$ and ending at $label(r_2)$, that can be derived if all rules in the chain are applied. For a rule with head $label(r_2) \succ label(r_1)$ to be applied, we must first find in all chains for $label(r_1) \succ label(r_2)$ an inapplicable rule.

Formally, we expand the inference condition for $+\partial$ by a case (2.4), in case $P(i+1) = +\partial(label(r_1) \succ label(r_2))$.

(2.4) $\forall C \in Chains(label(r_2) \succ label(r_1))$
$\exists r \in C\ \exists a \in A(r) : -\partial a \in P(1..i)$

The inference conditions for $-\partial$, $+\partial_{ap}$ and $-\partial_{ap}$ are modified in the same way. The inference conditions for strict provability and support remain unaffected.

Example 4.1 Consider

$r_1 : \Rightarrow r_3 \succ r_4$ $\quad r_3 : \Rightarrow p$
$r_2 : \Rightarrow r_4 \succ r_3$ $\quad r_4 : \Rightarrow \neg p$

We cannot derive $+\partial(r_3 \succ r_4)$ because of the chain (r_2, r_1), in which neither rule can be proven to be inapplicable. Therefore $-\partial p$ and $-\partial\neg p$.

4.2 PROPERTIES

First we make the following observation: if $\succ$ does not occur in a defeasible theory, then DDL2 behaves exactly as the classical defeasible logic.

Theorem 4.1 (Coherence) *It is impossible to derive both $+tag\ q$ and $-tag\ q$ from a defeasible theory D, for $tag \in \{\Delta, \partial, \partial_{ap}, \int\}$.*

Theorem 4.2 (Consistency) *$+\partial p$ and $+\partial\neg p$, or $-\partial_{ap} p$ and $-\partial_{ap} p$, only if $+\Delta p$ and $-\Delta p$.*

In other words, an inconsistency in the defeasible part can only occur if the same inconsistency already occurs in the strict part.

As in DDL1, there exists a chain of increasing expressive power among several tags, similar to the situation in defeasible logic without dynamic priorities.

Theorem 4.3 $+\Delta \subset\subset +\partial_{ap} \subset +\partial \subset +\int$.

For each inclusion there are defeasible theories in which the inclusion is strict.

5 ARGUMENTATION SEMANTICS

For space limitations we will only discuss the ambiguity blocking variant of DDL2. The argumentation semantics will be an expansion of (Governatori and Maher 2000) which considered the classic defeasible logic with static priorities.

5.1 ARGUMENTS

Usually arguments are defined to be proof trees (or monotonic derivations). However, defeasible logic requires a more general notion of argument due to the notion of team defeat (see (Governatori and Maher 2000) for details).

An *proof tree* for a literal p based on a set of rules R is a (possibly infinite) tree with nodes labelled by literals such that the root is labelled by p and for every node with label h:

> If $b_1, \ldots, b_n$ label the children of h then there is a ground instance of a rule in R with body $b_1, \ldots, b_n$ and head h.

Any literal labelling a node of proof tree P is called a *conclusion* of P. However, when we refer to *the* conclusion of a proof tree, we refer to the literal labelling the root. If all rules in a proof tree are strict, then the tree is called *strict*, else *defeasible*.

An *argument* A for p is a set of proof trees for p. A is *finite*, or *strict*, if all proof trees in A are finite or strict, respectively. A (*proper*) *subargument* of A is a (proper) subtree of a tree in A. Given a defeasible theory D, the set of arguments that can be generated from D is denoted by $Args_D$. Finally we define $\succ_{Args}$ to be the set of all literals $label(r_1) \succ label(r_2)$ which are conclusions of a rule that appears in an argument in $Args$.

At this stage we can characterize the definite conclusions of defeasible logic in argumentation-theoretic terms.

Proposition 5.1 *Let D be a defeasible theory and p be a literal.*

1. *$D \vdash +\Delta p$ iff there is a finite strict argument for p in $Args_D$*
2. *$D \vdash -\Delta p$ iff there is no (finite or infinite) strict argument for p in $Args_D$*

5.2 INTERACTION BETWEEN ARGUMENTS

An argument A *attacks* as argument B if a conclusion in A is the complement of a conclusion in B; or if $label(r_1) \succ label(r_2)$ is a conclusion in A, and all rules in a member of $Chains(label(r_2) \succ label(r_1))$ occur in a proof tree in B.

An argument A *defeats* a defeasible argument B at literal q w.r.t a set of arguments $Args$ if (1) there exist r_A in A and r_B in B with conclusions $\sim q$ and q, respectively, such that $label(r_A) \not\succ_{Args} label(r_B)$; or (2) $q = label(r_1) \succ label(r_2)$, r_B in B has conclusion q, and all rules in a member of $Cycles(label(r_2) \succ label(r_1))$ occur in a proof tree of A.

A set of arguments S defeats a defeasible argument B w.r.t. $Args$ if there is an $A \in S$ that defeats B w.r.t. $Args$. Note that $Args$ is used as a context which provides the superiority relation.

Example 5.1 Consider

$r_1 : a \Rightarrow p \quad \rightarrow a$
$r_2 : p \Rightarrow q \quad \rightarrow b$
$r_3 : b \Rightarrow \neg p \quad r_5 : \Rightarrow r_2 \succ r_4$
$r_4 : \neg p \Rightarrow \neg q$

Consider the arguments $A : a \Rightarrow p \Rightarrow q$ and $B : b \Rightarrow \neg p \Rightarrow \neg q$, and the set $Args = \{\Rightarrow r_2 \succ r_4\}$. The only priority information is $r_2 \succ_{Args} r_4$. Therefore A defeats B at $\neg p$ and $\neg q$ w.r.t. $Args$, and B defeats A at p w.r.t. $Args$.

Similar to (Governatori and Maher 2000) we define team defeat and undercut as follows: An argument *team defeats* a defeasible argument B at q w.r.t. $Args$ if for every rule r_B with conclusion q there is a rule r_A in A with conclusion $\sim q$, such that $label(r_A) \succ_{Args} label(r_B)$.

Let $strong(S)$ be the set of arguments, all of whose subarguments are supported by S. Obviously $S \subseteq strong(S)$. Also, note that if $A_1, A_2 \in strong(S)$ are arguments for the literal q, then $A_1 \cup A_2$ is also an argument for q. Thus there exists a maximal argument for q in $strong(S)$, denoted by $max(q, S)$.

A defeasible argument A is *undercut* by a set of arguments S if there is a literal q such that $strong(S)$ defeats a proper subargument of A at q w.r.t. S, and A does not team defeat $max(\sim q, S)$ at $\sim q$ w.r.t. S.

5.3 THE STATUS OF ARGUMENTS

An argument A is *supported* by a set of arguments S if every conclusion in A is also the conclusion of a finite argument in S. An argument A for p is *acceptable w.r.t. a set of arguments* S if (1) A is strict; or (2):

(2a) every proper subargument of A is supported by S, and

(2b) every argument attacking A is either undercut by S or team defeated by A.

Let D be a defeasible theory. We define J_i^D as follows:

- $J_0^D = \emptyset$
- $J_{i+1}^D = \{a \in Args_D \mid a \text{ is acceptable w.r.t. } J_i^D\}$

The set of *justified arguments* in D is $JArgs^D = \cup_{i \geq 1} J_i^D$. A literal is *justified* if it is the conclusion of a finite argument in $JArgs^D$.

Theorem 5.1 *$D \vdash +\partial q$ iff p is justified.*

An argument A is *rejected by sets of arguments S and T* if (1) A is defeasible, and either

(2a) a proper subargument of A is in S, or

(2b) there exists an argument B attacking A, such that B is supported by T, and A does not team defeat B w.r.t. T.

We define R_i^D as follows:

- $R_0^D = \emptyset$
- $R_{i+1}^D = \{a \in Args_D \mid a \text{ is rejected by } R_i^D \text{ and } JArgs^D\}$

The set of *rejected arguments* in D is $RArgs^D = \cup_{i \geq 1} R_i^D$. A literal is *rejected* if there is no argument for p in $Args^D - RArgs_D$.

Theorem 5.2 *$D \vdash -\partial q$ iff p is rejected.*

6 CONCLUSION

In this paper we studied dynamic priorities in the defeasible logics. We studied two variants, depending on whether cyclic priority information is allowed to occur or not. For each variant we considered levels of proof: strict, defeasible with ambiguity propagation,

defeasible with ambiguity blocking, and support. We defined the proof theory, derived coherence and consistency, and established a chain of inclusion for the various levels of proof. Finally we defined a more abstract characterisation in the form of an argumentation semantics.

The next step in this research line is to implement the logics and use them in applications.

References

G. Antoniou, M.J. Maher and D. Billington (2000). Defeasible Logic versus Logic Programming without Negation as Failure, Journal of Logic Programming, 42: 47–57.

G. Antoniou, D. Billington, G. Governatori and M.J. Maher (2000). A Family of defeasible reasoning logics and its implementation. In *Proc. 14th European Conference on Artificial Intelligence (ECAI'2000)*, 459–463.

G. Antoniou, D. Billington, G. Governatori and M.J. Maher (2001). Representation results for defeasible logic. *ACM Transactions on Computational Logic* 2: 255–287

G. Brewka (1996). Well-Founded Semantics for Extended Logic Programs with Dynamic Priorities. *Journal of Artificial Intelligence Research* 4: 19-36.

G. Brewka and T. Eiter (1999). Preferred Answer Sets for Extended Logic Programs. *Artificial Intelligence* 109: 297-356.

J.P. Delgrande, T. Schaub and H. Tompits (2000). Logic Programs with Compiled Preferences. In *Proc. ECAI'2000*, 464–468.

G. Governatori and M.J. Maher (2000). An Argumentation-Theoretic Characterization of Defeasible Logic. In *Proc. ECAI'2000.*

G. Governatori, A. ter Hofstede and P. Oaks (2000). Defeasible Logic for Automated Negotiation. In *Proc. Fifth CollECTeR Conference on Electronic Commerce*, Brisbane.

B.N. Grosof (1997). Prioritized conflict handling for logic programs. In *Proc. International Logic Programming Symposium*, MIT Press, 197–211.

B.N. Grosof, Y. Labrou and H.Y. Chan (1999). A Declarative Approach to Business Rules in Contracts: Courteous Logic Programs in XML. In *Proc. 1st ACM Conference on Electronic Commerce (EC-99)*, ACM Press.

M.J. Maher, A. Rock, G. Antoniou, D. Billington and T. Miller (2000). Efficient Defeasible Reasoning Systems. In *Proc. 12th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2000)*, IEEE, 384–392.

M.J. Maher (2002). Propositional Defeasible Logic has Linear Complexity. *Theory and Practice of Logic Programming*, to appear.

D. Nute (1994). Defeasible Logic. In D.M. Gabbay, C.J. Hogger and J.A. Robinson (eds.): *Handbook of Logic in Artificial Intelligence and Logic Programming Vol. 3*, Oxford University Press, 353–395.

H. Prakken (1997). *Logical Tools for Modelling Legal Argument: A Study of Defeasible Reasoning in Law.* Kluwer Academic Publishers.

H. Prakken, G. Sartor. Argument-based logic programming with defeasible priorities. *Journal of Applied Non-Classical Logics* 7 (1997): 25-75.

D.M. Reeves, B.N. Grosof, M.P. Wellman, and H.Y. Chan (1999). Towards a Declarative Language for Negotiating Executable Contracts, *Proceedings of the AAAI-99 Workshop on Artificial Intelligence in Electronic Commerce (AIEC-99)*, AAAI Press / MIT Press.

D.D. Touretzky, J.F. Horty and R.H. Thomason (1987. A Clash of Intuitions: The Current State of Nonmonotonic Multiple Inheritance Systems. In *Proc. IJCAI-87*, Morgan Kaufmann, 476–482.

Intuitions and the Modelling of Defeasible Reasoning: some Case Studies

Henry Prakken
Institute of Information and Computing Sciences
Utrecht University
Utrecht, The Netherlands
henry@cs.uu.nl
http://www.cs.uu.nl/staff/henry.html

Abstract

The purpose of this paper is to address some criticisms recently raised by John Horty in two articles against the validity of two commonly accepted defeasible reasoning patterns, viz. reinstatement and floating conclusions. I shall argue that Horty's counterexamples, although they significantly raise our understanding of these reasoning patterns, do not show their invalidity. Some of them reflect patterns which, if made explicit in the formalisation, avoid the unwanted inference without having to give up the criticised inference principles. Other examples seem to involve hidden assumptions about the specific problem which, if made explicit, are nothing but extra information that defeat the defeasible inference. These considerations will be put in a wider perspective by reflecting on the nature of defeasible reasoning principles as principles of justified acceptance rather than 'real' logical inference.

1 Introduction

The purpose of this paper is to address some criticisms recently raised by John Horty in two articles (Horty; 2001, 2002) against the validity of some commonly accepted defeasible reasoning patterns. Horty criticises two such patterns, viz. reinstatement and floating conclusions. In addition, he criticises a group of inferences where the failure to interleave argument construction and argument evaluation would lead to problems. For each of these cases, Horty presents a series of examples in which his intuitions are that it is coherent for a reasoner to accept the premises but not the conclusion.

Horty's examples are intriguing and his discussion of them is very insightful. Yet I disagree with the conclusions he draws from them. I shall argue that his examples do not demonstrate the invalidity of the reasoning patterns. Some of them reflect patterns which, if made explicit in the formalisation, avoid the unwanted inference without having to give up the inference principles criticised by Horty. In the remaining examples Horty's intuitions seem to be based on hidden assumptions about the specific problem which, if made explicit, are nothing but extra information under which the inference cannot be drawn; in other words, these examples do not show that the reasoning patterns are invalid but just that they are defeasible. I shall then put these considerations in a wider perspective, reflecting on the nature of defeasible reasoning principles as principles of justified acceptance rather than 'real' logical inference.

The discussion in this paper will be informal and conceptual rather than technical, and will freely switch from one style of nonmonotonic logic to another whenever appropriate (although with some bias to argumentation approaches). I assume that the reader is familiar with the essentials of the main approaches to nonmonotonic logic, especially default logic, preferential entailment and argumentation systems (particularly in the style of Pollock (1995) and Dung (1995)). Formal treatments of most examples of this paper can be found in the two papers by Horty and in (Prakken and Vreeswijk; 2002).

First I will say more on the role of intuitions in the development of logic in general and of nonmonotonic logic in particular.

2 Intuitions in logic

In both his articles Horty mainly relies on intuitions in concrete examples. In fact, he explicitly says that this is "the only method that I know of in this area, where there is no recourse to anything like a formal semantics: (...)" (Horty; 2001, p. 9). In fact, in the field of nonmonotonic logic this approach has been very popular; see e.g. Lifschitz (1988)'s well-known list of benchmark examples. Setting aside the issue whether a formal semantics of defeasible reasoning

is indeed impossible, I will argue that in this area relying on intuitions is extremely prone to error, and that there are better, more principled approaches.

The use of intuitions in evaluating logical systems has been criticed before; see e.g. Veltman (1985). For one thing, the question arises whose intuitions should count. Those of logicians are hopelessly corrupted by overexposure to formalism. So should we ask the 'average language user', hoping that they are not infected by theoretical bias? Then the problem often arises that their answers reveal a lacking understanding of the reasoning patterns; however, teaching them about these reasoning patterns infects them with the theoretical bias we were hoping to avoid. Another obvious problem is that intuitions of different persons often conflict. For these and other reasons a sensible way to use intuitions seems, each time they present us with a puzzle, not to directly rely on them, but to see whether they reflect some underlying pattern. This is the strategy I will follow in the present paper.

3 Reinstatement

The first reasoning pattern discussed by Horty (2001) is reinstatement. This pattern, endorsed by most nonmonotonic logics, can be naturally explained in terms of argumentation systems.

Let me first fix some terminology on argumentation systems that I will use throughout the paper, sometimes applied to other nonmonotonic logis as well (such application is possible since, as shown by e.g. Dung (1995) and Bondarenko et al. (1997), many nonmonotonic logics can be reformulated as argumentation systems). *Defeat* is a binary relation between arguments: that argument A defeats argument B means that A and B are in conflict with each other and B is not preferred over A. I say that A *strictly defeats* B if A defeats B and not vice versa.

Justification is a skeptical notion of nonmonotonic consequence. Argumentation systems usually classify argments into three classes, justified, defensible and overruled arguments. Loosely speaking, the *justified* arguments are those with which a dispute can be 'won', the *overruled* arguments are those with which a dispute should be 'lost', and the *defensible* arguments should leave the dispute undecided. Conclusions of justified arguments are skeptical consequences and those of defensible arguments are credulous consequences of a theory. Horty only discusses skeptical consequence, i.e., justification.

Now *reinstatement* is the phenomenon that an argument that is (perhaps strictly) defeated by another argument is still justified since it its defeater is itself strictly defeated by another justified argument; cf. Figure 1.

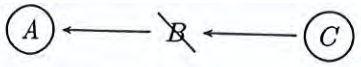


Figure 1: Argument C reinstates argument A.

Reinstatement is embodied in all Dung-style argumentation systems via a notion of *acceptability*: an argument A is acceptable with respect to a set S of arguments if all arguments defeating A are defeated by an argument in S. For instance, in Figure 1 the argument A is acceptable with respect to the set $S = \{A, C\}$.

Reinstatement has a *direct* and an *indirect* form, and Horty only rejects its direct form. Direct reinstatement is when all three arguments are in conflict on their final conclusions, as in the following example (where the more specific default is preferred over the less specific one).

Example 3.1

A: Tweety flies because it is a bird
B: Tweety does not fly because it is a penguin
C: Tweety flies because it is a magic penguin

Indirect reinstatement, on the other hand, is when the reinstating argument C defeats the 'middle' argument B on one of its intermediary conclusions, such as in the following example.

Example 3.2

A: Tweety flies because it is a bird
B: Tweety is a penguin because it was observed to be so, so Tweety does not fly
C: The observation that Tweety is a penguin is unreliable since it was done during a blizzard

Figure 2 depicts indirect reinstatement (as for notation, for any pair of arguments X and X^-, the latter is a proper subargument of the first.)

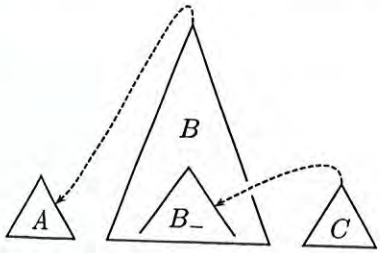


Figure 2: Indirect reinstatement

3.1 Is direct reinstatement invalid?

Horty (2001) argues that Example 3.1 shows that direct reinstatement is invalid, since the reason that Tweety flies is

that it is a magic penguin and not that it is a bird. (He does so in a critical discussion of (Prakken and Sartor; 1997), a system of prioritised extended logic programming that instantiates Dung (1995)'s grounded semantics.)

Before Horty reaches this conclusion, he first considers an alternative strategy with such examples, viz. to simply live with the anomaly that the first default is not defeated on the grounds that after all the correct conclusion is drawn that Tweety can fly. He rejects this strategy with an example in which the consequent of the least preferred default is logically stronger than that of the most preferred default.

Example 3.3

d_1: Microsoft employees tend to be millionaire
d_2: New Microsoft employees tend to have less than half a million
d_3: New Microsoft employees in department X have at least half a million.

Suppose again that the more specific default is preferred over the less specific one. According to Horty, in this case an argument for new Microsoft employees in department X based on the least specific default d_1 leads to the incorrect conclusion that they are millionaires. Horty therefore concludes that direct reinstatement must be rejected as a principle of defeasible reasoning.

How convincing is Horty's argument? I agree with him that in Examples 3.1 and 3.3 the least specific default should not be applied, so Horty is right when he says that reinstatement cannot be combined with the simple representation method of Example 3.3. In this sense I agree with his criticism on my earlier work in (Prakken and Sartor; 1997). However, I think that these observations do not imply invalidity of direct reinstatement, since there is an alternative approach which respects Horty's observations in the examples but validates reinstatement, and which can make relevant distinctions between kinds of examples that go unnoticed in the simple representation scheme of Example 3.3.

The alternative approach is to make the language expressive enough to let applicable defaults block the applicability of other defaults when needed. One way to implement this approach is well-known technique of exception or abnormality clauses (note that the logic of (Prakken and Sartor; 1997) supports this technique, although in most of our examples we did not use it). For instance, in preferential entailment with minimisation of the Ab predicate, all preferred models of the following theory plus the fact that Jeff is a new Microsoft employee of department X satisfy that Jeff owns at least half a million dollars, but not all such models satisfy that he is a millionaire.

r_1: *Microsoft employee* $\wedge \neg Ab_1 \rightarrow \geq 1M$
r_2: *New Microsoft Employee* $\rightarrow$ *Microsoft employee*
r_3: *New Microsoft employee* $\wedge \neg Ab_3 \rightarrow < 1/2M$
r_4: *New Microsoft employee* $\rightarrow Ab_1$
r_5: *New MicrosoftX Employee* $\rightarrow$ *New Microsoft employee*
r_6: *New MicrosoftX employee* $\wedge \neg Ab_6 \rightarrow \geq 1/2M$
r_7: *New MicrosoftX employee* $\rightarrow Ab_3$

In argumentation terms, the argument for Ab_1 is not affected by the argument for Ab_3: a new Microsoft employee is still an abnormal Microsoft employee, but he may also be an abnormal new Microsoft employee.

The reason that this approach seems better than simply invalidating reinstatement is that whether reinstatement should go through or not seems to depend on the nature of the domain, the kind of knowledge involved and the context in which this knowledge is used. Consider the following moral example on reasons for the severity of punishment.

Example 3.4

r_1: For theft imprisonment upto 6 years is acceptable
r_2: For theft out of poverty imprisonment of more than 3 years is not acceptable
r_3: For theft during riots only imprisonment of more than 4 years is acceptable

And suppose that r_3 is preferred to r_2 which is in turn preferred to r_1. As in the Microsoft example, the consequent of the most preferred default is logically weaker than the consequent of the least preferred default. Yet it seems that for theft during riots out of poverty any imprisonment must be between 4 and 6 years. A general reason for this outcome could be that reasons for the severity of punishment do not block but just outweigh each other.

Or consider the following legal evidence example.

Example 3.5 Consider the following three witness statements.

w_1: John says that the suspect stabbed the victim
w_2: Bob says that the suspect did not kill the victim
w_3: Al says that the suspect killed the victim

And suppose that for whatever reason Bob is regarded as more credible than John and Al as more credible than Bob (suppose, for instance, that Al is always right). Here it seems quite reasonable to accept the least credible source since the only defeating source is overruled.

The point of these examples is that in modelling default reasoning one needs the extra expressiveness in order to control the blocking of defaults. Of course, one should avoid tinkering with the formalisation in concrete examples just

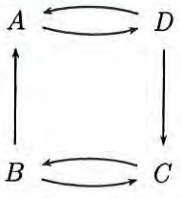
to get a desired outcome. Therefore, the challenge for research is to find general principles for choosing the right formalisation. For two examples of research in this spirit see Pollock's research on undercutting defeaters in epistemic contexts, e.g. Pollock (1995), and Hage and Verheij's work on defeasible reasoning with legal rules and principles, e.g. Verheij et al. (1998). This work indicates that such principles depend on the domain, the kind of knowledge involved and the context in which it is used. For instance, in statistical reasoning with empirical regularities blocking more general defaults by more specific defaults can be justified by the total evidence requirement of statistical reasoning. Now a crucial difference between the three examples of this section is that only Example 3.3 involves reasoning of the latter kind; Example 3.4 involves moral reasoning, while Example 3.5 involves reasoning about the credibility of information sources.

Concluding this section, I agree with Horty that direct reinstatement does not go together with the simple representation scheme of Example 3.3 (used by Horty (1994) in his inheritance systems and by me and Giovanni Sartor in most of (Prakken and Sartor; 1997)). However, I have argued that the problems are not due to direct reinstatement but to the weak expressiveness of the representation. This conclusion can be strengthened if further defeasible reasoning patterns can be found where more expressiveness of the above kind is needed. And below I will argue that such further reasoning patterns indeed exist.

3.2 Should argument construction and argument evaluation be interleaved?

Horty (2001) discusses a further set of examples which according to him reveals another flaw of Dung-style systems, viz. that they don't interleave the construction and evaluation of arguments. However, I will argue that these examples are better treated as attacks on reinstatement.

Example 3.6 Consider four arguments A, B, C and D such that B strictly defeats A, D strictly defeats C, A and D defeat each other and B and C defeat each other.



Horty discusses the following instantiation, again with subclass-superclass defeat.

- A: Larry is rich because he is a public defender, public defenders are lawyers, and lawyers are rich
- B: Larry is not rich because he is a public defender, and public defenders are not rich
- C: Larry is rich because he lives in Brentwood, and people who live in Brentwood are rich
- D: Larry is not rich because he rents in Brentwood, and people who rent in Brentwood are not rich

Since none of these arguments is undefeated, in none of the Dung-style systems it skeptically follows that Larry is poor. The same holds for prioritised default logics (e.g. Brewka (1994)). Yet it might be argued that since both arguments that Larry is rich are strictly defeated by an argument that Larry is not rich, it should skeptically follow that Larry is not rich. This is the outcome obtained by Horty (1994).

Horty's diagnosis is that the problems are caused by mistakenly separating the phases of argument construction and argument evaluation. According to him, Dung-style systems reflect a picture of defeasible reasoning in which first all possible arguments are constructed, all relevant defeat relations are identified, and then the justified arguments are determined on the basis of this overall pattern of defeat. Horty prefers an approach in which "arguments are constructed step-by-step and are evaluated in each step of their construction: those that are indefensible (...) are discarded at once, and so cannot influence the status of others." (Horty; 2001, p. 11). Applied to the example, in this approach the arguments A and C are discarded as soon as their defeat by B, respectively, D is noted, so that they cannot interfere with D, respectively, B.

To comment on this, note first that the interleaving approach is consistent with the validity of direct reinstatement. Suppose a reasoner, when confronted with the arguments A and B, aks whether A is now "clearly indefensible". The answer is "yes" only if the fact that B is in turn defeated by C is regarded irrelevant for the status of A, so only if reinstatement is rejected; but the interleaving approach does not force this answer. So, Horty's examples seem more convincing as examples against reinstatement than against 'non-interleaving'.

Now does the example reveal a flaw with direct reinstatement? The same alternative approach as above prevents a clear positive answer:

- r_1: $\mathit{Lawyer} \wedge \neg \mathit{Ab}_1 \rightarrow \mathit{Rich}$
- r_2: $\mathit{Public\ defender} \rightarrow \mathit{Lawyer}$
- r_3: $\mathit{Public\ defender} \wedge \neg \mathit{Ab}_3 \rightarrow \neg\, \mathit{Rich}$
- r_4: $\mathit{Public\ defender} \rightarrow \mathit{Ab}_1$
- r_5: $\mathit{Brentwood\ resident} \wedge \neg \mathit{Ab}_5 \rightarrow \mathit{Rich}$
- r_6: $\mathit{Brentwood\ tenant} \rightarrow \mathit{Brentwood\ resident}$
- r_7: $\mathit{Brentwood\ tenant} \wedge \neg \mathit{Ab}_7 \rightarrow \neg\, \mathit{Rich}$
- r_8: $\mathit{Brentwood\ tenant} \rightarrow \mathit{Ab}_5$

The point is that r_4 and r_8 give rise to two additional arguments E and F, undercutting, respectively, A and C, so that B and D are justified. Again the undercutters are based on the general principle that statistical information about subclasses overrides conflicting statistical information about superclasses.

3.3 Conclusion on reinstatement

What conclusions can be drawn fom the discussion so far? I have said that there are good reasons not to reject direct reinstatement but instead to adopt a logic of sufficient expressiveness in which reinstatement is valid but can be avoided. With the latter I mean that the logic should offer the means to formalise, when desirable, reinstatement-like examples in a way that does not formally reflect the reinstatement pattern. This approach should be combined with an investigation of general principles for choosing the proper formalisation.

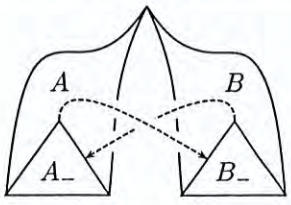
4 Floating conclusions

I now turn to Horty (2002)'s criticism of another defeasible reasoning pattern. Sometimes one is faced with two conflicting and equally strong arguments, but whatever way the conflict would be resolved, a certain conclusion could always be reached. Such conclusions were by Makinson and Schlechta (1991) called 'floating conclusions'. When modelling skeptical reasoning, the question is whether floating conclusions should come out as justified.

Floating conclusions can only be modelled as justified if the logic gives rise to alternative conclusion sets in case of an irresolvable conflict (such as default logic or Dung's preferred and stable argumentation semantics). In such logics the justified conclusions can be defined as those contained in all such sets. If a logic always induces a unique conclusion set (such as Dung (1995)'s grounded semantics and Horty (1994)'s inheritance system), then floating conclusions cannot be recognised as justified. This is generally regarded as a drawback of such systems.

Let us consider an example.

Example 4.1 (Floating conclusions.) Consider the arguments A^-, A, B^- and B such that A^- and B^- defeat each other and A and B have the same conclusion.



An intuitive reading (entirely based on reality) is

A^-: Brygt Rykkje is Dutch since he was born in Holland
B^-: Brygt Rykkje is Norwegian since he has a Norwegian name
A: Brygt Rykkje likes ice skating since he is Dutch
B: Brygt Rykkje likes ice skating since he is Norwegian

Whichever way the conflict between A^- and B^- is decided, we always end up with an argument for the conclusion that Brygt Rykkje likes ice skating, so it seems that this conclusion is justified, even though it is not supported by a justified argument. In other words, the status of this conclusion floats on the status of the arguments A^- and B^-.

It is easy to see that Horty's 'interleaving' approach to defeasible reasoning fails to draw this floating conclusion, since it cuts off both arguments after A^- and B^-. However, it is important to note that rejecting floating conclusions does not imply a rejection of direct reinstatement. There are natural systems (such as grounded semantics for Dung's argumentation systems) that validate reinstatement but that do not validate floating conclusions.

4.1 Horty examples (1): counterexamples or additional information?

In the above example, there seems nothing wrong with accepting the floating conclusion as justified. However, Horty (2002) presents an intriguing set of other instantiations, in all of which the situation is less clear. Horty therefore concludes that it is at least coherent for a skeptical reasoner not to accept floating conclusions, so that such conclusions cannot be accepted as a matter of logic.

Horty's main example is a rather dramatic one, in which a person (call him Bob) must quickly decide about a certain investment, and whose parents are both terminally ill. Bob knows that both his parents are rich, but he does not know whether they will let him inherit a large sum of money. His sister tells him that she has spoken to both her parents, and that father will not let him inherit anything but mother will let him inherit enough to make his investment. His brother tells him the reverse, viz. that he has also spoken to both his parents, and that not his mother but his father will let him inherit enough to make his investment. Horty argues that in this example it is reasonable to withhold any judgement on whether the investment is secure, since brother and sister's conflicting testimonies undermine each other's credibility; and why should it be more reasonable to believe one of them than to believe neither?

The examples highlight a very interesting pattern, viz. the pattern where the fact that sources of information con-

flict undermines the credibility of both sources. However, I think that this specific example has some weaknesses, which indicate the problematic nature of intuitions in concrete examples. The point is that it is very easy to read an additional default principle into the example, viz. that people tend to speak the truth about their intentions, and that it is also very easy to mistake this additional information for the intuition that the reasoning pattern is invalid. Note that this additional default principle is undercut as soon as it turns out that a person has told conflicting things about his or intentions to different persons. Now if both the additional principle and its undercutter are made explicit, there will also be conclusion sets where neither father nor mother lets Bob inherit anything, so that the issue of floating conclusions does not arise.

The same analysis applies to several other of Horty (2002)'s examples. For instance, Horty discusses an example with two defaults saying that persons tend to live where their spouse lives and that they tend to live where they work. He applies this to a case where a person works in X while his wife lives in Y, and he observes that the conclusion that the person lives in X or Y is less likely than the conclusion that he lives somewhere in between X and Y. Again I agree, but I think that Horty fails to make another default explicit, viz. that couples who both work but not in the same city, tend to live somewhere in between where they work. If this default is added, the unwarranted disjunctive conclusion is not justified any more.

The main lesson to be learned from these examples is that in modelling defeasible reasoning it is very dangerous to rely on intuitions in concrete examples. In fact, counter-intuitions are much more problematic in defeasible than in deductive reasoning. In the latter, any counterexample invalidates an inference, precisely because the inferences are meant to be deductive. But when extra information can invalidate an inference, the boundary between being a counterexample and being additional invalidating information becomes vague.

However, there are also examples to which the above criticism does not seem to apply with the same force.

4.2 Horty examples (2): mutually undermining floating conclusions

Horty also discusses examples where it is less easy to point at suppressed additional information. In one example, two military spies report to their commander on their enemy: one spy says that the enemy has retreated to the forest, the other that the enemy has retreated to the mountains. Should the commander believe that the enemy has retreated and act accordingly? In this example I agree with Horty that the fact that the two spy reports conflict undermines the credibility of both, so that the commander is not justified in believing that the enemy has retreated.

For a legal version of this example, consider the following two witness testimonies.

Example 4.2

w_1: John says that the suspect stabbed the victim
w_2: Bob says that the suspect shot the victim

Clearly, the two witnesses undermine each other's credibility, so in the absence of further information on their credibility the conclusion that the suspect killed the victim is not warranted.

However, again I disagree that these examples simply show the invalidity of a reasoning pattern. In fact, the same strategy works as when reinstatement seems undesirable: whenever floating conclusions appear unacceptable, formalise the example in such a way that it does not instantiate the pattern of floating conclusions.

Of course, this strategy makes sense only if the best formalisation can be chosen on the basis of general criteria. Now in all of these examples the key issue is reliability of a certain source of information or advice. So we should look for general criteria for how the credibility of witnesses or experts can be undercut. One natural principle is that the fact that two witnesses or experts contradict each other undercuts the credibility of both. In Pollock (1995)'s system, which formalises the interaction between defeasible reasons and their possible undercutters, this could be expressed as follows.

r_1: If witness w says p, this is a prima facie reason for believing p
r_2: if witness w' says q and p and q are incompatible, this is a prima facie reason for believing $\neg r_1$

r_2 is a (defeasible) undercutter of r_1. It is easy to see that in case of John and Bob r_2 applies to both of them so r_1 is blocked for of them.

However, in Section 3 I argued that a third witness statement might reinstate one of Bob or John's testimony. Consider again Example 3.5 and assume that for certain particular reasons Al's statement is judged more credible than Bob's:

w_1: John says that the suspect stabbed the victim
w_2: Bob says that the suspect did not kill the victim
w_3: Al says that the suspect killed the victim
r_3: w_3 and *special reasons* are a prima facie reason for believing $\neg r_2$ as applied to Bob.

r_3 is an undercutter of the instance of r_2 that applies to Bob. The result in Pollock's system is that not only the argument

based on Al's testimony but also the argument based on John's testimony is justified, so that we are warranted in believing not only that the suspect killed the victim but also that he shot him.

So far I have put a lot of effort in avoiding floating conclusions by formalising examples in a certain way. Of course, this effort is worthwhile only if there are also also examples where floating conclusions should be accepted, otherwise I might just as well adopt a logic that simply invalidates them. Example 4.1 is such an example. The defaults in that example are just statistical regularities; the fact that they conflict does not undermine the fairness of the samples on which they are based, so they do not undercut each other. Another example where conflicting defaults do not undermine each other is conflicting interpretations of a legal norm. If both interpretations lead, for instance, to the conclusion that someone has committed a tort, there seems no reason for the judge to choose.

4.3 Concluding

Concluding, I have shown that skeptical reasoning in multiple-extension logics with a sufficiently expressive language can distinguish between cases where floating conclusions appear to be justified and cases where they appear not to be justified. Of course, it is debatable whether the patterns identified by me are indeed relevant. Therefore, I will next discuss how Horty's approach fares if the pattern of floating conclusions is rejected.

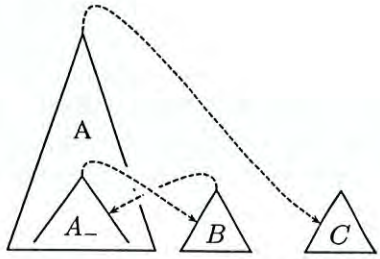
5 Zombie arguments

Let us for the sake of argument suppose that floating conclusions should not be accepted as justified. Is Horty's approach then unproblematic, or are there other reasoning patterns where it runs into problems?

Following Makinson and Schlechta (1991) I think there is such a pattern, viz. so-called 'zombie arguments'. To explain this, Horty's approach must be described in more detail. Recall from Section 3 that Horty regards defeasible reasoning as bottom-up construction of arguments interleaved with argument comparison: moreover, arguments are compared in a "deeply skeptical" way: as soon as a counterargument is found that is at least not weaker, an argument is cut off. In case two counterarguments are equally strong, this means that both arguments are cut off. This explains why Horty's approach invalidates floating conclusions, since in the relevant examples construction of an argument is cut off before it reaches the floating conclusion.

Now this "deeply skeptical" approach deals in a peculiar way with another type of example.

Example 5.1 (Zombie arguments.) Consider the arguments A^-, A, B and C such that A^- and B defeat each other and A defeats C.



A concrete example is

A^-: Dixon is a pacifist since he is a quaker
B: Dixon is no pacifist since he is a republican
A: A^- + Dixon has no gun since he is a pacifist
C: Dixon has a gun since he lives in Chicago

Suppose further that the first two defaults in the example are equally strong so that A^- and B defeat each other, while the third default is stronger than the fourth so that A strictly defeats C. In Horty's approach we must conclude that Dixon has a gun, since the only potential counterargument, A, is cut off after detecting that its subargument A^- is defeated by B.

Here Horty's deep skepticism seems self-defeating, since it makes an argument justified that is not justified in an approach that Horty regards as a weaker form of skepticism, viz. to consider all ways to resolve the conflict between A^- and B. Makinson and Schlechta (1991) call arguments like B 'zombie' arguments: B is not 'alive', (i.e., not justified) but it is not fully dead either; it has an intermediate status (defensible), in which it can still influence the status of other arguments.

Now is someone who does not want to accept floating conclusions forced to accept problems with zombie paths? This is not the case, since there are forms of skepticism that reject floating conclusions but recognise zombie arguments. For instance, in Dung's preferred or stable semantics one could define the justified conclusions to be those of arguments that are in all extensions. Or consider Dung's grounded semantics, which regards as justified all arguments in the least fixpoint of an operator that for any set of arguments returns all arguments that are acceptable to it. In the example this operator, when applied to the empty set, returns the empty set since none of the arguments is undefeated. Now, interestingly, these 'moderate' forms of skepticism validate direct reinstatement, so that it seems hard to reject direct reinstatement without incurring problems with Zombie arguments.

Finally, how does the alternative approach deal with Zom-

bie arguments? Recall that this approach consists of accepting both direct reinstatement and floating conclusions while providing the means to formalise examples such that they are not instances of these patterns. Now interestingly, in cases where I agree with Horty that floating conclusions should be avoided, zombie arguments might remain undetected. To see this, extend the military example with two defaults, viz. that in general the mountains are safe, but that if the enemy is in the mountains, the mountains are not safe. Then we have the following three arguments:

Example 5.2

A^-: The enemy has retreated to the mountains since spy Al says so
B: The enemy has retreated to the forest since spy Bob says so
A: A^- + The mountains are not safe since the enemy is in the mountains
C: By default, the mountains are safe.

If the two spy defaults are blocked as above, then the zombie argument A is not constructible, so C comes out as justified.

Is this a problem for the alternative approach? I don't think so, because of the same difference in underlying pattern as we have seen above: while Example 5.1 involves empirical regularities, Example 5.2 involves credibility of information sources. And in the latter case it seems reasonable to accept that two conflicting spy reports undermine each other if there is no reason to prefer one of them.

6 Conclusion

In this paper I have examined whether Horty's counterexamples against reinstatement and floating conclusions have provided convincing reasons to reject the validity of these reasoning patterns. In the course of my discussion, I have also examined the method by which Horty evaluates nonmonotonic logics, viz. assessing them in the light of intuitions concerning concrete examples. I now summarise the main conclusions drawn in this paper, and then put them in a wider perspective.

6.1 On reinstatement and floating conclusions

To start with, we can say that Horty's discussion of reinstatement and floating conclusions has significantly raised our understanding of these reasoning patterns: Horty has rightly pointed at problems with naively accepting them. However, I have argued that his examples do not yield convincing reaosns to reject these patterns. A better conclusion from Horty's examples is that nonmonotonic logics should provide the expressiveness to make relevant distinctions between classes of examples. More precisely, I have argued that nonmonotonic logics should validate at least reinstatement and perhaps also floating conclusions, but provide the means to formalise examples such that they do not instantiate these patterns. This approach should be complemented with investigating general criteria for choosing the proper formalisation.

An additional reason for this approach is that a complete rejection of reinstatement leads to a failure to recognise so-called zombie arguments, which failure seems to indicate that Horty's deeply skeptical approach is in fact self-defeating. On the other hand, we have seen that rejecting floating conclusions does not imply a rejection of reinstatement so that Horty's arguments against the former pattern do not reinforce his reasons for rejecting the latter pattern.

6.2 On Horty's method of evaluating nonmonotonic logics

As I said above, Horty claims that the only suitable way to evaluate nonmonotonic logics is applying them to concrete examples and checking whether they validate one's intuitions concerning these examples. In Section 2 I already hinted at some general drawbacks of this approach, and I have illustrated these drawbacks in this paper with several examples in which intuitions seemed to be confused with implicit information which, when made explicit, makes the counterintuitiveness disappear. This confusion can very easily arise in defeasible reasoning, since a hallmark of such reasoning is that inferences that are valid relative to certain information can be invalidated by further information. In my opinion it is therefore better to use intuitions not as critical tests but as generators for further investigation. One should always first ask whether perhaps the counterexample is based on suppressed additional information. If that is not the case, one should ask whether the counterexample reflects a general pattern that can be made explicit by enriching the language.

6.3 On the nature of principles of defeasible reasoning

I end my conclusions with some speculative remarks on the nature of defeasible reasoning principles, which are meant to put the foregoing in a wider perspective. These remarks are partly repeated from (Prakken and Vreeswijk; 2002), in some agreement with e.g. Loui (1998) and Schlechta (2000) and perhaps going back to Toulmin (1958). Interestingly, towards the end of his (2002) paper Horty comes very close to endorsing a similar position.

For the most part, Horty's two papers reflect a traditional picture of nonmonotonic logic as essentially just another

kind of logic, defining validity of inferences, so that the question whether a certain reasoning pattern should be endorsed solely depends on the meaning of the logical operators involved. This explains his attitude that one counterexample suffices to reject the validity of a defeasible reasoning pattern. However, a different view is possible, viz. to simply accept that defeasible reasoning is logically invalid reasoning: after all, nonmonotonic conclusions of a theory are not satisfied by all models of the theory. This leads to a different, more pragmatic picture of the status of defeasible reasoning patterns as principles of justified acceptance, i.e., criteria for when a statement can be accepted on the basis of a theory given that it does not logically follow from the theory. Put differently, since defeasible reasoning is "quick and dirty" reasoning (Schlechta; 2000), we cannot expect a neat and clean theory of such reasoning; the best we can hope for is heuristics, and heuristics are not invalidated by single counterexamples.

If this perspective is taken, room is made for other than just syntactic and semantic criteria: what is also important is, for instance, the nature of the domain and the knowledge employed (e.g. statistical, causal, moral, legal) and the context in which this knowledge is used (e.g. scientific reasoning, quick commonsense planning, a law suit, a medical treatment). This paper's approach with exception clauses provides the means to express at least some such considerations in the object level language, so that their 'nonlogical' nature is respected. This perspective can also do justice to the dynamic aspects of defeasible reasoning (cf. Schlechta (2000)); for instance, a judge faced with two conflicting witnesses as in Example 4.2, would not yet draw a conclusion but instead ask the witnesses "did you hear any sound?".

References

Bondarenko, A., Dung, P., Kowalski, R. and Toni, F. (1997). An abstract, argumentation-theoretic approach to default reasoning, *Artificial Intelligence* **93**: 63–101.

Brewka, G. (1994). Reasoning about priorities in default logic, *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pp. 247–260.

Dung, P. (1995). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming, and n–person games, *Artificial Intelligence* **77**: 321–357.

Horty, J. (1994). Some direct theories of nonmonotonic inheritance, *in* D. Gabbay, C. Hogger and J. Robinson (eds), *Handbook of Logic in Artificial Intelligence and Logic Programming*, Clarendon Press, Oxford, pp. 111–187.

Horty, J. (2001). Argument construction and reinstatement in logics for defeasible reasoning, *Artificial Intelligence and Law* **9**: 1–28.

Horty, J. (2002). Skepticism and floating conclusions, *Artificial Intelligence* **135**: 55–72.

Lifschitz, V. (1988). Benchmark problems for formal nonmonotonic reasoning, version 2.00, *Proceedings of the Second Workshop on Nonmonotonic Reasoning*, number 346 in *Springer Lecture Notes in AI*, Springer Verlag, Berlin.

Loui, R. (1998). Process and policy: resource-bounded non-demonstrative reasoning, *Computational Intelligence* **14**: 1–38.

Makinson, D. and Schlechta, K. (1991). Floating conclusions and zombie paths: two deep difficulties in the "directly sceptical" approach to defeasible inheritance nets, *Artificial Intelligence* **48**: 199–209.

Pollock, J. (1995). *Cognitive Carpentry. A Blueprint for How to Build a Person*, MIT Press, Cambridge, MA.

Prakken, H. and Sartor, G. (1997). Argument-based extended logic programming with defeasible priorities, *Journal of Applied Non-classical Logics* **7**: 25–75.

Prakken, H. and Vreeswijk, G. (2002). Logics for defeasible argumentation, *in* D. Gabbay and F. Günthner (eds), *Handbook of Philosophical Logic*, second edn, Vol. 4, Kluwer Academic Publishers, Dordrecht/Boston/London, pp. 219–318.

Schlechta, K. (2000). Remarks about nonmonotonic logics and dynamic reasoning. Unpublished notes, once available at www.cmi.univ-mrs.fr/~ks/remarks.html.

Toulmin, S. (1958). *The Uses of Argument*, Cambridge University Press, Cambridge.

Veltman, F. (1985). *Logics for Conditionals*, Doctoral dissertation, Department of Philosophy, University of Amsterdam.

Verheij, B., Hage, J. and van der Herik, H. (1998). An integrated view on rules and principles, *Artificial Intelligence and Law* **6**: 3–26.

Changing and Integrating Information: From Theory to Practice

Possibility Theory in Information Fusion [1]

Didier Dubois Henri Prade

IRIT- Institut de Recherche en Informatique de Toulouse
Université Paul Sabatier, 118 route de Narbonne
F-31062 Toulouse cedex 4, France

Abstract

Possibility theory and the body of aggregation operations from fuzzy set theory provide some tools to address the problem of merging information coming from several sources. Possibility theory is a representation framework that can model various kinds of information items: numbers, intervals, consonant random sets, special kind of probability families, as well as linguistic information, and uncertain formulae in logical settings. The possibilistic approach to fusion is general enough to encompass logical modes of combination (conjunctive and disjunctive) as well as fusion modes used in statistics. This general framework allows to import inconsistency handling methods, inherited from logic, into numerical fusion problems. The approach applies to sensor fusion, aggregation of expert opinions as well as the merging of databases especially in case of poor, qualitative information.

1 INTRODUCTION

Information fusion is a problem met in various areas:

• in robotics, data coming from various sensors must be merged in order to establish the positioning of autonomous robots (Abidi and Gonzales, 1992);

• in image processing, several remotely-sensed images of some area must be merged with maps of the same area; the same problem occurs with medical images that must be exploited conjointly with anatomic pictures. See (Bloch, 2001), Bezdek *et al.* (1999, Section 4.9);

• in risk analysis, information items pertaining to the reliability of physical components are supplied by several experts and must be fused (Cooke, 1991);

• the fusion of databases, or of replies to queries addressed to several databases is also of current interest (Baral *et al.* 1992).

The information fusion problem has several characteristics that distinguish it from other kinds of apparently similar problems like criteria aggregation, estimation, revision, and its operational setting is less restrictive than the one of statistical sciences. The fusion problem considered here can be summarized as follows: Given a set of sources of information, generally heterogeneous ones, supplying data about the value of some parameter or the description of a situation, find the most plausible values of the observed parameter or the most credible description of the situation, trying to discard erroneous pieces of information, while remaining faithful to the available information.

By a source, we mean a lot of different things: it can be a human (an expert), a sensor, or a database, hence a potential heterogeneity of the pieces of information. In the case of an expert, information can be linguistic, or can be expressed as a set of intervals or subsets weighted by levels of confidence. Information extracted from a database can be a histogram (for a statistical database) or a logical formula (for a deductive database). For a sensor, information is a number with an error interval, or a likelihood function induced by a set of chance densities. Our view of the information to be combined is thus rather generic. However, it is assumed that all pieces of information pertain to the same matter and is expressed in the same format. This assumption is not always verified in practice. When it is not, there is a problem of information correspondence which is an area of research of its own and is not addressed here.

A first issue to be clarified is the meaning of the items that are supposed to be merged. Major distinctions are to be made between:

i) preference aggregation *vs.* information fusion;

---

1. The full text of this paper is published in the book "Data Fusion and Perception" (G. Della Riccia, H.-J. Lenz, R. Kruse, Eds)

CISM Courses and Lectures N° 431, Springer Verlag, Berlin, 2001, pp. 53-76.

ii) the combination of information coming from parallel sources *vs.* the revision of already available information;

iii) estimation *vs.* fusion.

In this paper, we are interested in merging information, not preference. In the preference aggregation problem, it makes sense to find the opinion of the "average person" in an homogeneous group of individuals, and to look for trade-offs between criteria expressing different points of view. On the contrary, the information merging problem is a matter of finding the truth in a reliable way: what is required is to make the best of the available information by discarding the wrong data when possible, and keeping the right information. Logical combinations are then natural candidates as fusion operators. Conjunctive combinations apply when all the sources are reliable, while disjunctive combinations deal with the case of unreliable sources hidden in a group of other reliable ones. Obviously, prioritized logical combinations may be considered, in particular when the sources are not equally reliable. Averaging operations in information aggregation can be justified when the set of sources can be viewed as a single random source producing different inputs. In that case, indeed, the set of data to be fused can be interpreted as standard statistics. For instance, several successive measurements from a single sensor can be viewed as the result of a random experiment. Then the discrepancies between the sources can be explained in terms of random variability and fusion comes close to a filtering problem. However in the case of unique measurements issued from heterogeneous sensors, or in the case of expert knowledge or databases, it is not clear that averaging combination modes make sense.

Besides, we oppose the case of merging information from parallel sources to the problem of belief revision where sources do not play a symmetrical role. In the first situation, all the sources provide information simultaneously, while in the revision process there is a chronological ordering between the source which represents the prior state of belief, and the source which supplies the new information. In each case, the pooling obeys different requirements. For instance, belief revision is generally not commutative. Kalman filtering proceeds by repeated sequences of a prediction step followed by a revision step.

One must moreover distinguish the estimation problem from the fusion problem. In the case of estimation, the problem is to find a representative value for a numerical parameter, on the basis of point-value estimates. Several kinds of estimators exist in the statistical literature, like the mean value, most prominently, but also the median value, or the mode of the distribution induced by the data set. In the fusion problem, the data is generally imprecise. Intervals, fuzzy intervals, uncertainty distributions or even linguistic terms are present. These pieces of information may belong to different referential sets, and a common space must be built for all of them. In information fusion, the question is to provide a synthetic view of the available data, not to supply a single representative value.

Simple probabilistic methods in data fusion have some limitations, which motivate the introduction of other approaches: First, the identification of a single probability distribution requires more information than what is sometimes actually available. Human experts supply intervals rather than point-values because their knowledge is not only of limited reliability, but also tainted with imprecision. Probability theory is not concerned with the modeling of imprecision but rather captures the notion of random variability, or in the case of subjective probabilities, betting odds. Partial ignorance cannot be properly distinguished from randomness by means of a single probability function. Probability theory is also a restricted framework for information fusion, because the number of fusion modes it proposes in practice is very limited. Basically, only convex linear sums of probability functions are probability functions (e.g. Lehrer and Wagner, 1981). Bayesian probability does provide a general setting for information fusion, but makes sense only if a priori distributions are available. This assumption is not always verified when information comes from parallel sources.

This paper surveys a general approach to information fusion in the setting of possibility theory. (Dubois *et al*, 1999). Our basic claim is that there cannot be a unique mode of combination, which would be satisfactory in any situation, even when the framework for representing information is chosen, here possibility theory. We stress that both logical fusion modes and statistical fusion modes can be accommodated and generalized in the possibilistic framework. The Bayesian approach to fusion is not incompatible with our proposal.

However, this chapter does not discuss the application of fuzzy rule-based techniques to data fusion. Bouchon-Meunier (1997) offers a collection of papers showing the use of fuzzy set theory in various aggregation problems including data fusion. We also do not treat the problem of merging logical databases at the syntactic level. This problem can also be addressed in our framework, but has been already discussed at length elsewhere (Benferhat *et al*, 1997, 2000). The fuzzy and possibilistic approaches have been applied in several fields where merging information is needed (see the full paper). Bloch and Hunter (2001) propose a discussion of general concepts and methods of data and knowledge

fusion, and a series of papers applied to various domains of information technology.
This paper is organized as follows. Section 2 points out the variety of information items a possibility distribution is capable of accounting for. Section 3 presents the basic modes of information fusion, first in the case of set-valued imprecise information, then for merging general possibility distributions. An efficient algorithm is provided for computing maximal subsets of coherent sources in the case of intervals. Section 4 presents variants and extensions of the basic fusion modes.

2 REPRESENTING INCOMPLETE INFORMATION

Consider an ill known parameter x. Let U denote the referential where x takes its values. The available information on the actual value of x is supposed to be modelled by a possibility distribution π_x (Zadeh, 1978). It is a mapping from U to a totally ordered plausibility scale L. When U is a real interval or a Cartesian product thereof, L = [0, 1]. A possibility distribution can be viewed as a representation of the more or less plausible values of an unknown quantity x. These values are assumed to be mutually exclusive, since x takes on only one value (its true value). Since one of the elements of U is the true value of x, $\pi_x(u^*) = 1$ for at least one value $u^* \in U$. This is the normalization condition, that claims that at least one value is viewed as totally possible. If u and u' are such that $\pi_x(u) > \pi_x(u')$, u is considered to be a more plausible value than u'. When $\pi_x(u) = 0$, then x cannot take on value u. The possibility degree of an event A, understood as a subset of U is

$$\Pi(A) = \sup_{u \in U} \pi_x(u).$$

It is computed on the basis of the most plausible values of x in A, neglecting other realizations. A possibility distribution π_x is at least as informative (more specific) as another one π'_x if and only if $\pi_x \leq \pi'_x$. In particular, if $\forall u \in U$, $\pi_x(u) = 1$, π_x contains no information at all (since it expresses that any value in u is possible for x).

This representation is quite general, because it is capable of modeling several kinds of imprecise information within a unique setting.

2.1 Intervals

The simplest form of a possibility distribution on a numerical interval U is the characteristic function of a sub-interval I of U, i.e., $\pi_x(u) = 1$ if $x \in I$, 0 otherwise. This type of possibility distribution is naturally obtained from experts stating that "x lies between a and b". This way of expressing knowledge is more natural than giving a point-value u* for x right away, because it allows for some imprecision. However this binary representation is not entirely satisfactory. If the interval is too narrow, the piece of information is not so reliable. When $\pi_x(u) = 0$ for some u, it means that x = u is impossible. This is too strong, and one is then tempted to use wide uninformative intervals. Sometimes, even the widest, safest interval does not rule out some residual possibility that the value of x lies outside it.

2.2 Confidence Intervals

It is more satisfactory to describe imprecise information by means of several intervals with various levels of confidence. A possibility distribution π_x can then represent a finite family of nested confidence subsets $\{A_1, A_2, \ldots, A_m\}$ where $A_i \subset A_{i+1}$, i = 1, ...m − 1. Each confidence subset A_i is attached a positive confidence level λ_i. The set of possibility values $\{\pi(u) \mid u \in U\}$ is then finite. The links between the confidence levels λ_i's and the degrees of possibility are defined by postulating λ_i is the degree of necessity (i.e. certainty) of A_i which is defined as $N(A_i) = 1 - \Pi(A_i^c)$ where $\Pi(A_i^c)$ is the degree of possibility of the complement A_i^c of A_i (Dubois and Prade, 1988b). This entails that $\lambda_1 \leq \ldots \leq \lambda_m$ due to the monotonicity of the necessity function N. The possibility distribution equivalent to the family $\{(A_1, \lambda_1), (A_2, \lambda_2), \ldots, (A_m, \lambda_m)\}$ is defined as the least informative possibility distribution π that obeys the constraints $\lambda_i = N(A_i)$, i = 1,m. It comes down to maximizing the degrees of possibility $\pi(u)$ for all u in U, subject to these constraints. The solution is unique and is $\pi_x(u) = \min_i \max(1 - \lambda_i, A_i(u))$, where $A_i(\cdot)$ is the characteristic function of A_i. This solution is the least committed one with respect to the available data, since by allowing the greatest possibility degrees in agreement with the constraints, it defines the least restrictive possibility distribution. Conversely, the family $\{(A_1, \lambda_1), (A_2, \lambda_2), \ldots, (A_m, \lambda_m)\}$ of confidence intervals can be reconstructed from the possibility distribution π_x. Suppose that the set of possibility values $\pi_x(u)$ is $\{\alpha_1 = 1, \alpha_2 \geq \alpha_3 \ldots \geq \alpha_m\}$ and let $\alpha_{m+1} = 0$. Then, $\forall$ i = 1, ...,m,

$$A_i = \{u \mid \pi_x(u) \geq \alpha_i\}; \lambda_i = 1 - \alpha_{i+1},.$$

In particular $\lambda_m = 1$ and A_m is the subset which for sure contains x; moreover, $A_m = U$ if no strict subset of U surely includes x. This analysis extends to an infinite nested set of confidence intervals $\{(A_\alpha, 1 - \alpha), \alpha > 0\}$, where $N(A_\alpha) = 1 - \alpha$, and $\pi_x(u) = \inf_\alpha \max(\alpha, A_\alpha(u))$.

2.3 Random Sets

Letting $p_i = \alpha_i - \alpha_{i+1}$, $\forall$ i = 1,...,m. Note that

$$\forall u, \pi_X(u) = \Sigma_{i:u\in A_i} p_i. \quad (1)$$

The sum of weights $p_1, \ldots, p_m$ is 1. Hence the possibility distribution can be cast in the setting of random sets (Dubois and Prade, 1988a), and more precisely Shafer (1976) theory of evidence. From a mathematical point of view, the information modelled by π_X can be viewed as a nested random set $\{(A_i,...p_i), i = 1,...m\}$, which allows for imprecision (reflected by the size of the A_i's) and uncertainty (the p_i's). And p_i is the probability that the source supplies exactly A_i as a a faithful representation of the available knowledge of x (it is *not* the probability that x belongs to A_i). The random set view of possibility theory is developed in details in (Gebhardt and Kruse, 1998). Namely given a bunch of imprecise (not necessarily nested) observations, equation (1) supplies an approximate representation of the data. See Fig.1.

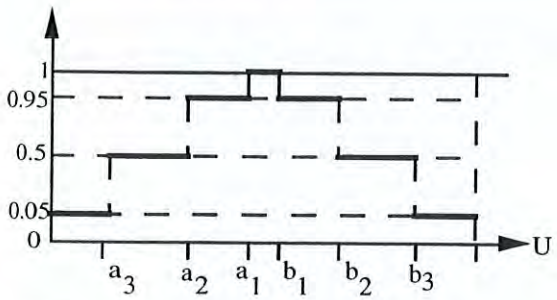


Fig. 1 Expert-originated possibility distribution

2.4 Imprecise Probability

The level of confidence λ_i can also be conveniently interpreted as a lower bound on the probability that the true value of x hits A_i. Then the possibility distribution is viewed as encoding the family of probability measures $\mathbb{P} = \{P, P(A_i) \geq \lambda_i, i = 1,m\}$ because the possibility measure $\Pi(B)$ coincides with the upper probability $P^*(B) = \sup\{P(B), P\in \mathbb{P}\}$ while the necessity measure $N(B) = \inf\{P(B), P\in \mathbb{P}\}$ is the lower probability; see (Dubois and Prade, 1992b, De Cooman and Aeyels, 1999) for details.

2.5 Probabilistic Confidence Intervals

Possibility distributions may also come from the transformation of a probability distribution (Dubois *et al.* 1993). Namely, given a unimodal probability density p on the real line, a (nested) set of confidence intervals of p can be encoded as a possibility distribution. For any interval [a, b], let $\pi(a) = \pi(b) = 1 - P([a, b])$. Choosing the confidence intervals centered on the mode of P and obtained by level-cutting the density, π is the most specific possibility distribution compatible with P (the inequality $\Pi \geq P$ holds). Starting from a uniform probability distribution on [a, b], one can justify the symmetric triangular fuzzy number of support [a, b] as one best possibilistic approximation of the uniform density in the above sense. Moreover, if Π is the possibility measure obtained from the symmetric triangular fuzzy number of support [a, b], then the inequality $\Pi \geq P$ holds for any symmetric normal density of support [a, b]. This suggests that triangular fuzzy numbers are genuine counterparts of uniform probability densities. This setting is adapted to sensor measurements.

2.6 Likelihood Functions

Yet another interpretation of the possibility distribution π consists in viewing it as a likelihood function, that is, identifying $\pi(u)$ to the probability $P(u_m \mid u)$ that the source indicates the measured value u_m, when the actual value of x is u. Indeed suppose only the likelihood function $P(u_m \mid u)$ is known, $\forall u \in U$. The probability $P(u_m \mid u)$ is understood as the likelihood of $x = u$, in the sense that the greater $P(u_m \mid u)$ the more $x = u$ is plausible. In particular, note that $\forall A \subseteq U$

$$\min_{u\in A} P(u_m \mid u) \leq P(u_m \mid A) \leq \max_{u\in A} P(u_m \mid u).$$

Identifying $\pi(u)$ to $P(u_m \mid u)$, it is clear that the upper bound of the probability $P(u_m \mid A)$ is a possibility measure (see Dubois *et al.*, 1997). Hence there is a strong similarity between maximum likelihood reasoning and possibilistic reasoning in the finite case. The degree of possibility that $x = u$ in the face the measurement u_m can be defined as $\pi_m(u) = \pi_X(u \mid u_m) = P(u_m \mid u)$. In the continuous case $\pi_X(u \mid u_m)$ must be extrapolated from the density, whose values may be greater than 1. This setting is adapted to sensor measurements as well. Note that the probabilistic likelihood view is somewhat restrictive because in general, the available information may be weaker than the probability $P(u_m \mid u)$.

We may only have limited knowledge of this probability under the form of some bounds. For instance all that may be known is that if $x = u$ then $u_m \in E$ for some interval E. Then define $\pi(u_m \mid u) = 1$ if $u_m \in E$ and 0 otherwise. More generally the conditional possibility distribution may take values between 0 and 1. In general $\sup_u \pi(u_m \mid u) \leq 1$ only, since in this approach, the norma-lisation with respect to u is not warranted. Yet, it is natural to assume that $\max_u \pi(u_m \mid u) = 1$. It corresponds to existing practice in statistics whereby the likelihood function is renormalized via a proportional rescaling (Edwards, 1972). It is also a postulate of observation relevance (Dubois Prade, and Yager 1999). It means there is at least one value $x = u$ that makes the observation $x_m = u_m$

completely possible; in other words u_m is a completely relevant observation for the set of assumptions U. Indeed, if for instance $\max_u \pi(u_m \mid u) = 0$, it would mean that it is impossible to observe $x_m = u_m$ for any value $x = u \in U$.

2.7 Linguistic Information

Human-originated information, even when it pertains to orders of magnitude, is often linguistic. In some contexts, only witnesses can supply information, and they use words. Zadeh (1965) proposed the theory of fuzzy sets as a tool for mathematical modelling of linguistic information, especially on numerical universes. A possibility distribution can be identified to the membership function of a fuzzy set modelling linguistic terms (in a given context) like "tall" (for heights) "hot" (for temperatures) "far" (for distances), describing possible values of a quantity x. So linguistic information taking the form of statements like "this man is tall", "the room temperature is hot", "the robot is far from the table" can be modelled by means of possibility distributions as well. However, the semantics of the possibility degrees is then described in terms of distance to fuzzy set prototypes, whose membership values is maximal. If the piece of information is of the form "x is F", where F is a fuzzy set, the closer a value u to a prototype u* of F, the more plausible it is, which is expressed by letting $\pi_x(u) = F(u)$, where $F(\cdot)$ is the membership function of F, aand satisfies $F(u^*) = 1$.

3 Fusion Modes

Consider a question whose answer is unknown and that, for simplicity, comes down to finding the real value of a parameter x (generally a numerical value, but also possibly a Boolean one if the question is of the yes/no type; possibly several such parameters). A set S of sources is supposed to provide some estimations of the value of x. These pieces of information can be very poor, not necessarily point-values. They can take the form of tolerance intervals, or can be synthetized by plausibility distributions of some sort. In the limit, they can be non-informative at all, as when some source expresses its ignorance as to the value of x. The fusion process aims at finding the most plausible values of the parameter x on the basis of the available pieces of information. The pooling of possibility distributions can be based on a set-theoretic, or in other words, logical, view of the combination of uncertain information as advocated a long time ago in (Dubois and Prade, 1988a). There are basically two extreme logical modes of symmetric combination: the conjunctive mode when all sources agree and are considered as reliable; and the disjunctive mode when sources disagree and at least one of them is wrong. These modes are implemented respectively in terms of fuzzy set intersections and fuzzy set unions. However, the statistical approach of synthetizing observed data can also be extended. The general postulates of the possibilistic approach to the fusion of information issued from several distinct sources are, first, that there is no unique combination mode, and, second, that the choice of the combination mode depends on an assumption about the reliability of sources, as formulated by some analyst. No a priori knowledge about the inquired parameter is needed, and the sources, viewed in parallel, are to be combined in a symmetric way if equally reliable.

3.1 Basic fusion modes for sets

In order to clarify the situation of the fusion problem, it is useful to consider the case of sets representing imprecise pieces of information. We do not make any prior assumption on the type of sources. Especially, the case of heterogeneous parallel sources is included in the approach, so that possible conflicts between the pieces of information are not the result of random variations only. Consider the case of two sources of information giving evidence about the value of some variable x under the form of a set. The elementary fusion problem is simply the following one:

Source 1 $\quad x \in E_1 \subseteq U$

Source 2 $\quad x \in E_2 \subseteq U$

What should be said about x?

We claim that the choice of a type of pooling operator that solves this problem is a matter of context. There is no theory liable of prescribing a universal pooling method for these two pieces of information, that would apply to all situations. The chosen pooling scheme depends not only on formal properties the method should fulfil, but also on how much the sources agree, and what is known about their reliability. It means that a panoply of combination schemes together with underlying assumptions must be laid bare and used in an adaptive way. There are two basic settings for fusion: logical and statistical.

Logical fusion modes

Set-theoretic operations such as unions and intersections (expressing disjunction and conjunction) are the basic natural consensus rules in set theory and logic. In logic, putting two formulas ϕ and ψ in a knowledge base exactly means that only the models of the conjunction $\phi \wedge \psi$ will be

retained. If $E_1 \cap E_2$ is not empty, the default assumption that both sources are reliable makes sense, and the combination rule can be the intersection, i.e., $x \in E_1 \cap E_2$. Otherwise, if $E_1 \cap E_2$ is empty, at least one of the sources is necessarily wrong, and a more natural combination rule is the union, i.e., $x \in E_1 \cup E_2$, which assumes that the other source is not wrong. This operator does not require the identification of the right source.

Let us consider now the case of more than two sources yielding subsets $E_1, E_2,\ldots E_n$. The two above elementary fusion modes cannot be directly used in a convincing way. Indeed the more sources, the more likely it is that the intersection $\cap_{i=1,\ldots,n} E_i$ is empty. However the more sources, the more likely it is that the union $\cup_{i=1,\ldots,n} E_i$ covers the whole universe. In logic, when a knowledge base is inconsistent, it is usual to consider maximal consistent sub-bases, and derive plausible conclusions from them (Rescher and Manor, 1970). So a natural suggestion for pooling set-valued information coming from several sources is to consider maximal consistent subsets of sources. Let $K \subseteq S$ be a maximal subset of sources such that $\cap_{i \in K} E_i \neq \emptyset$. There are generally more than one such subset and let us denote them by K_1, $K_2,\ldots, K_p$. The inconsistency tolerant logical fusion mode leads to consider the set of values

$$E = \cup_{j=1,\ldots,p} \cap_{i \in K_j} E_i, \qquad (2)$$

as the result of a rational fusion mode. On the real line, the E_i are often represented by intervals. E is then a union of disjoint intervals, and it makes sense to approximate the result by means of its convex hull.

Another approach to the fusing of n sets is to assume that there are k reliable sources among the n ones; this assumption leads to a symmetric quantified aggregation that lies between the conjunctive (k = n) and the disjunctive (k = 1) fusion modes:

$$E = \cup_{K:|K|=k} \cap_{i \in K} E_i. \qquad (3)$$

This aggregation does not require any knowledge about which sources are reliable and which ones are not. However it makes sense only if $\cap_{i \in K} E_i \neq \emptyset$ for at least one subset K of k sources. Hence the most optimistic assumption to be made in the quantified aggregation is $k = \max\{|K|, \cap_{i \in K} E_i \neq \emptyset, K \subseteq S\}$. It leads to the cardinality-based counterpart of the maximal consistent subset aggregation (2), where $K_1, K_2,\ldots, K_p$ are selected as the maximal consistent sets of sources in the sense of cardinality.

Yet another assumption consists in assuming that sources may lie, i.e. may claim that a value of x is possible while it is not. This type of assumption is studied by Mundici (1992) in the framework of Lukasiewicz logic and is dealt with in the next section, as it comes down to merging possibility distributions.

Statistical fusion mode

A more usual mode of fusing numerical information is to add pieces of information and form a histogram. This mode makes sense, in practice, when sources are sufficiently numerous, considered independent and identical (like a set of identical independently performed random experiments). However, it is easily extended to sources supplying set-valued pieces of information E_i. Under the proposed assumption, the family $\{E_i, i = 1,\ldots,n\}$ is viewed as a uniform random set, each set E being the right information with probability m(E) equal to the proportion of sources delivering the information "x is E". The counterpart of the histogram is then the possibility distribution π such that (see for instance Gebhardt and Kruse, 1998)

$$\pi(u) = \Sigma_{u \in E}\, m(E) = |\{i : u \in E_i\}| / n.$$

Note that in the case of usual histograms, the family of sets with positive mass forms a partition of the referential, each set being weighted by the proportion of precise information items it contains. Clearly, in the general case, the possibility distribution π does not contain the whole information described by the mass function m.

The latter can be recovered only if the family $\{E_i, i = 1,\ldots,n\}$ is nested, which will not hold generally. Moreover, this fusion mode goes beyond the pure set theoretic framework. Indeed, this mode of fusion is intermediary between the conjunctive and disjunctive modes (less optimistic than the former and less pessimistic than the latter), and yields the membership function π of a fuzzy set. This remark leads us to extend all the fusion modes described in this section to the case where sources deliver possibility distributions directly.

3.2 Basic combination modes in the many-valued case

We now assume that the information supplied by each source can be modelled by a possibility distribution. As seen in section 2, this model covers many situations ranging from linguistic to probabilistic information (such as likelihood functions). The logical fusion modes naturally exploit fuzzy set-theoretic operations. We assume that degrees of possibility are numerical, as the range of fusion operations is richer.

There exist several possible choices among pointwise fuzzy set-theoretic operations for defining conjunctive and disjunctive combinations of possibility distributions. Let π_i be the possibility

distribution supplied by source i, i = 1,n. Define $\forall$ u $\in$ U,

$\pi_{\wedge}(u) = *_{i=1,n}\, \pi_i(u)$ (fuzzy set intersection)

$\pi_{\vee}(u) = \perp_{i=1,n}\, \pi_i(u)$ (fuzzy set union)

where $*$ and $\perp$ are [0,1]-valued operations defined on [0,1] $\times$ [0,1] which enjoy the duality relation $\alpha \perp \beta = 1 - (1 - \alpha) * (1 - \beta)$ in order to express De Morgan's laws. Candidates for conjunctive and disjunctive fusion operations are triangular norms and co-norms, respectively. The conjunctive t-norm operation $*$ is associative, commutative, increasing (in the wide sense) in both places and such that $\alpha * 1 = \alpha$ and $0 * 0 = 0$ (e.g., Fodor and Yager, 2000). The main continuous solutions are

$\alpha * \beta = \min(\alpha, \beta)$,
$\alpha * \beta = \alpha \cdot \beta$ (product)
and $\alpha * \beta = \max(0, \alpha + \beta - 1)$ (linear product),

which, by duality correspond to disjunctive operations

$\alpha \perp \beta = \max(\alpha, \beta)$,
$\alpha \perp \beta = \alpha + \beta - \alpha \cdot \beta$ (probabilistic sum),
and $\alpha \perp \beta = \min(1, \alpha + \beta)$.

Interpreting Basic Fusion Rules

As in the set-valued case, conjunctive aggregations make sense if all the sources are considered as equally and fully reliable, while disjunctive aggregations only presuppose that there is at least one reliable source for sure in the group, but it is not known which one.

In the conjunctive case, the minimum operation corresponds to a purely logical view of the combination process: the source which assigns the least possibility degree to a given value is considered as the best-informed with respect to this value. Note that with minimum, when all sources perfectly agree ($\forall$ i, $\forall$ j, $\pi_i = \pi_j$), there is no reinforcement effect. The idempotence of minimum deals with the possible redundancy of sources due to common background, and it copes with duplicated information.

On the contrary, if $*$ = product, and all the sources agree that a value x = u is not fully possible, then this value has a possibility degree strictly smaller than $\min_{i=1,\ldots n}\, \pi_i(u)$, i.e., the lack of complete possibility is reinforced; a necessary condition for choosing such an operation is the independence of the sources. This assumption may be more adapted to sensor fusion problems than to expert opinion pooling. Especially, the possibilistic model then gets very close to the Bayesian approach to fusion.

Assume, that n independent sensors measure a single quantity x, and let u^i_m be the measurement delivered by sensor i. Let $P(u^1_m, u^2_m, \ldots, u^n_m \mid x)$ be the conditional probability that the measured values supplied by the sensors are $(u^1_m, u^2_m, \ldots, u^n_m)$, depending on x. In order to simplify the data collection, the conditional independence assumption is often made. Interpreting $\pi_i(x)$ as $P(u^i_m \mid x)$, the possibilistic product rule is recovered, namely,

$$P(u^1_m, u^2_m, \ldots, u^n_m \mid x) = P(u^1_m \mid x) \cdot P(u^2_m \mid x) \cdot \ldots \cdot P(u^n_m \mid x)$$

It is also possible to use the linear product $\alpha * \beta = \max(\alpha + \beta - 1, 0)$ instead of product or min; this is a very drastic combination rule that discards elements that both sources consider to be little possible. It has been proposed by Boldrin and Sossai (1995) in a possibilistic logic setting (up to normalization), and by Mundici (1992) with a different terminology. The drastic reinforcement effect is explained in the latter reference by assuming that in the set of sources, a given number of them lie, but they are not identified. For instance, assume n independent sources, k of which lie. Assume source i says $x \in E_i \subseteq U$. Any value that is rejected by more than $k < n$ sources is truthfully rejected. Values rejected by k sources or less cannot be discarded. The information supplied by each source is then modelled by possibility distributions defined by

$\pi_i(u) = 1$ if $x \in E_i$, and $k/(k+1)$ otherwise.

Besides, the more sources agree on a value, the more this value is reliable. It can be checked that $\pi_{\wedge}(u) = \max(0, \sum_{i=1,n}\, \pi_i(u) - n + 1) = 0$ if and only if x = u is rejected by more than k sources. More generally, $\pi_{\wedge}(u) \in (0, 1]$ reflects the plausibility of x = u namely the number of sources that consider x = u as possible, when this number is at least n – k.

Similar comments can be made for disjunctive fusion rules. Tolerance to redundancy is achieved by the maximum operation. In the case of independent sources the probabilistic sum loses more information than the maximum, and the bounded sum even more so.

More generally, if the aim of the fusion is just to visualize the proportion of sources than claim that some value is possible, one may choose an extension of the statistical approach to fusion by preferring the arithmetic mean of possibility values

$$\pi_{AM}(u) = (\sum_{i=1,n}\, \pi_i(u)) / n.$$

This operator reduces to the statistical fusion mode when the imprecise information is set-valued, $x \in E_i \subseteq U$ for each source i. Here, a random fuzzy

set is built by considering that each possibility distribution π_i has probability 1/n. Then π_{AM} supplies the expected membership function.

The minimum and the product operations can also be justified in the setting of upper and lower probabilities as best possibilistic approximations of intersections of probability families (Dubois and Prade, 1992b; Dubois *et al.*, 1997). In the random set setting, the three basic triangular norms can be recovered from the intersection of random sets, of which the π_i's are the one-point coverage functions. The conjunctive fusion rule using the product operator coincides (up to the normalization factor) with Dempster's rule applied to consonant belief functions (which are mathematically the same as necessity measures), for the computation of the plausibility of singletons. See Dubois and Prade (1989), Gebhardt and Kruse (1998) for more justifications of fuzzy set connectives, in the random set setting. Other fusion modes such as disjunctive ones and averaging ones can be justified likewise. A disjunctive fusion corresponds to a union of probability families, or of random sets. A trade-off operation like the arithmetic mean is very similar to the consensus method of convex mixing of probabilities or the convex mixing of random sets.

Normalized Conjunctive Fusion

An important issue with the conjunctive combination is the fact that the result may be subnormalized, i.e., it may happen that $\forall u$, $\pi_\wedge(u) < 1$. In that case it expresses a conflict between the sources. Clearly the conjunctive mode makes sense if all the π_i's significantly overlap, i.e., $\exists u$, $\forall i$, $\pi_i(u) = 1$, expressing that there is at least one value of x that all sources consider as completely possible. If $\forall u$, $\pi_\wedge(u)$ is significantly smaller than 1, this mode of combination is debatable since, in that case, at least one of the sources or experts is likely to be wrong, and a disjunctive combination might be more advisable. Let the degree of consistency of π_1 and π_2 be defined as

$$h(\pi_1,\pi_2) = \sup_{u\in U} \pi_1(u)*\pi_2(u).$$

In case of subnormal results ($\sup\pi_\wedge < 1$), one may think of renormalizing $\pi_\wedge$, thus leading to the new operator (here written for n = 2 for simplicity) :

$$\pi(u) = \frac{\pi_\wedge(u)}{h(\pi_1, \pi_2)}.$$

The scalar value 1- $h(\pi_1,\pi_2)$ is a natural measure of conflict between two possibility distributions. Of course, $h(\pi_1,\pi_2)$ should be strictly positive, that is, the two sources must not be in absolute conflict.

The normalized conjunction enforces the assumption of full reliability of the sources since all values that are considered impossible by one source, even if considered possible by the others, are rejected. It presupposes that sources are reliable even if very conflicting, so that the values that remain after pooling contain the true one, however little possible they may be. This normalized operator is very analogous to Dempster's rule of combination of Shafer (1976)'s belief functions, which, from a random set point of view is nothing but a normalized random set intersection under a stochastic independence assumption. It has been also pointed out (Dubois and Prade, 1988a) that on a binary frame, i.e., a 2-element set U = {u,u'}, the normalized conjunction with * = product, coincides with the rule of combination used in the famous MYCIN expert system (Buchanan and Shortliffe, 1984), interpreting the MYCIN certainty factor CF(u) as $\pi(u) - \pi(u')$. Moreover for * = product, the normalized rule of combination is still associative, otherwise it may not be so. Yet another normalisation procedure consists in letting $\pi(u) = \pi_\wedge(u) + 1 - h(\pi_1,\pi_2)$ (Yager, 1985); it comes down to turning the level of conflict into a level of uncertainty on the set U.

When Fusion is not Appropriate

Assume that two pieces of information describing a situation are to be merged, one that stems from general considerations on similar situations, and the other is a piece of direct evidence on the situation. Then insofar as these pieces of evidence conflict, combination must be avoided and priority should be given to the most specific information. This idea has been around since the mid-eighties in the artificial intelligence literature (e.g., Poole, 1985) in reaction to problems encountered in uncertainty-handling rule-based systems such as MYCIN (Buchanan and Shortliffe, 1984) where fusion rules were sometimes blindly applied to combine conflicting inferences. For instance, if "Tweety is a bird" and "most birds fly", then "probably Tweety flies"; but if we also hear that "Tweety is a penguin", then, because "penguins do not fly", we have another argument claiming that "Tweety does not fly". Viewing this case as one of fusion from two independent sources is strange, because it will result in uncertainty about whether Tweety flies or not. However, it is clear that Tweety does not fly because penguins are an exceptional small class of birds and the generic information about birds is not relevant when information about the more specific class of penguins is available. If such a specificity-driven reasoning is carried out about sources, then a blind combination can be avoided. This is the case when

the pieces of information to be fused come as conclusions obtained from a knowledge base.

3.3 A linear complexity algorithm for finding maximal consistent subsets of intervals

The adaptive fusion rules based on the search for maximal consistent subsets require an efficient procedure for finding those families of consistent sources. Generally it is very difficult to identify these sets of maximally consistent sources. In logic, this problem is of exponential complexity. However, if the referential is totally ordered, for instance the real line, and the E_i are intervals, then finding maximal consistent subsets of intervals can be done in polynomial time in a very efficient way (Dubois *et al.*, 2000). Namely, maximal consistent subsets $K_1, \ldots, K_p$ of intervals $\{E_1, \ldots, E_n\}$ can be obtained by means of a sorting procedure, taking advantage of the ordered structure of U. Let $E_i = [a_i, b_i]$.

First, any non-empty intersection of intervals $\{E_1, \ldots, E_n\}$ is of the form $[a_i, b_j]$ (with maybe i = j). The intersections of maximal consistent subsets of $\{E_1, \ldots, E_n\}$ have a minimal width in the sense of inclusion. Thus, let $[a_i, b_j]$ be such a minimal interval. If we can find b_k such that $a_i < b_k < b_j$, or a_k such that $a_i < a_k < b_j$, this means that E_k is included in any maximal consistent subset including E_i and E_j, and that the interval $[a_i, b_j]$ is not minimal, which leads to a contradiction. The maximal consistent subset K of intervals with intersection $[a_i, b_j]$ is defined by the maximal number of elements of $\{E_1, \ldots, E_n\}$ which contain $[a_i, b_j]$. If $E_k \in K$ then $a_k \leq a_i$ (which is equivalent to $a_k \leq b_j$ since $[a_i, b_j]$ is minimal) and $b_k \geq b_j$. Thus all the $E_k = [a_k, b_k]$ such that $a_k \leq b_j$ are in K except those for which $b_k < b_j$. The minimal intervals $[a_i, b_j]$ including no a_k or b_k are easily found as follows: First, rank in increasing order the 2n numbers $\{a_i, i = 1,\ldots n\} \cup \{b_i, i = 1,\ldots n\}$; let $c_1 \leq \ldots \leq c_{2n}$ denote the result. Moreover, let us denote

type(i) = a if c_i is of the form a_k and

= b if c_i is of the form b_k.

Let s(1),..., s(p) be the p positions in the sequence of c_i such that type(s(j)) = b and type(s(j) − 1) = a. It is obvious that any interval $[c_{j-1}, c_j]$ such that type(j) = b and type(j − 1) = a, is the intersection of intervals in a maximal consistent subset. Such intervals are $\{[c_{s(j)-1}, c_{s(j)}], j = 1, p\}$ and any maximal consistent subset is characterized by such a pair in the sequence. The above result is the basis of an efficient approach : The p maximal consistent subsets of $\{E_1, \ldots, E_n\}$ can be computed as $K_j = \mathbb{A}_j - \mathbb{B}_j$ where $\mathbb{A}_j = \{E_k, a_k \leq c_{s(j)}\}$ and $\mathbb{B}_j = \{E_k, b_k < c_{s(j)}\}$. This leads to the following algorithm, linear in the number of intervals, exploiting the sequence $c_1 \leq \ldots \leq c_{2n}$:

Algorithm:
List = Ø; K = Ø;
for i = 1, 2n − 1
If type(i) = a, add to K all E_k such that $c_i = a_k$

if also type(i + 1) = b, add K to List
If not, delete from K all E_k such that $c_i = b_k$

end for

List will contain all the maximal consistent subsets of $\{E_1, \ldots, E_n\}$.

Remark : if $a_i = b_i$, the ordering should be done as if $a_i < b_i$. If $a_i = b_j$ the ordering should be done as if $a_i < b_j$ for taking into account the intersection between E_i and E_j which then reduces to a point.

Example

$E_1 = [1, 8]$; $E_2 = [2, 4]$; $E_3 = [3, 9]$, $E_4 = [5, 10]$, $E_5 = [5, 6]$; $E_6 = [7, 11]$. The algorithm first collects $K = \{E_1, E_2, E_3\}$. Then finding b_2, K is kept in List, since the mininal interval $[a_3, b_2]$ is found. E_2 is deleted from K, and E_4 and E_5 are added. Then the algorithm finds b_5 and minimal interval $[a_5, b_5]$, and so $\{E_1, E_4, E_3, E_5\}$ is added to List. The algorithm then deletes E_5 from K and then adds E_6, then finds b_1 and minimal interval $[a_6, b_1]$. Then $\{E_1, E_3, E_4, E_6\}$ is added to List. We might stop there, but the algorithm continues until K is empty. The result of the fusion is thus $[3,4] \cup [5, 6] \cup [7,8]$ whose convex hull is [3, 8].

4 Refined Fusion Modes

The basic symmetric fusion modes are generally not sufficient in practice, for several reasons. A first reason may be that it is known that some sources are more reliable than others. In order to take this information into account, weighted or prioritized fusion schemes must be envisaged. Even if the reliability of sources is unknown, one may suspect that not all of them are reliable and one may have some idea about the proportion of good sources. It leads to quantified fusion. Lastly, even if nothing is known about the sources, one may get an idea of how many of them are good by studying the conflicts among them. It leads to adaptive fusion modes.

4.1 Discounting

One may take into account the possibility that the information given by a source is meaningless, for instance, when a sensor is out of order. When the sensor works, it delivers some value u_m with some imprecision resulting in a possibility distribution π. When it is broken, the reading means nothing ($\pi(u) = 1, \forall u$). A reliability weight

attached to a source can be viewed as the level of certainty that the sensor is working. Then a possibility distribution accounting for the fact that the source is unreliable can be obtained via a form of discounting. If the degree of certainty that the source is reliable is known, say α, then it is possible to account for this information by changing the possibility distribution π, provided by the source, into (Dubois and Prade, 1988b):

$$\pi' = \max(\pi, 1 - \alpha).$$

When $\alpha = 1$ (fully reliable source), then $\pi' = \pi$, and when $\alpha = 0$ (absolutely unreliable source), then $\forall u, \pi'(u) = 1$ (total ignorance). Note that $\alpha = 0$ does not mean that the source lies, but that it is impossible to know whether its advice is good or not. This view makes sense even if the plausibility scale is not numerical.

There exist other proposals for numerical certainty qualification that behave similarly in limiting cases. Yager (1984) suggested an expression of the form

$$\pi' = \alpha * \pi + 1 - \alpha.$$

by analogy with the discounting of focal elements in Shafer (1976)'s approach to belief functions. Focal elements are ordinary subsets corresponding to probable imprecise localizations of the value of x in U. Here the focal elements are π and U itself, weighted by α and $1 - \alpha$ respectively, where π is possibly a fuzzy membership function. Operation $*$ is a conjunction like those in section 3.2. For $\gamma * \delta = \max(0, \gamma + \delta - 1)$, we recover $\pi' = \max(\pi, 1 - \alpha)$.

When the information provided by the source is given under the form of an ordinary subset E of U, all above expressions coincide and yield a simple support belief function (Shafer, 1976) focusing on the subset E with probability α.

4.2 Unequally reliable sources

When sources have unequal reliabilities, the symmetric fusion modes can be extended by allocating discounting coefficients to them. When these values are not available, a priority ranking of sources can be dealt with via a consistency analysis, whereby the information coming from a low priority source is allowed to modify the information coming from a higher priority source only if these pieces of information are consistent.

Discounted Fusion

The above discounting techniques lead to generalized conjunctive and disjunctive fusion rules which, for conjunctive and disjunctive idempotent operations, are of the form:

$$\pi_{\wedge\alpha} = \min_{j=1,n} \alpha_j * \pi_j + 1 - \alpha_j$$

$$\pi_{\vee\alpha} = \max_{j=1,n} \alpha_j - \alpha_j * (1 - \pi_j)$$

In these expressions, operation $*$ stands for minimum, product or linear product $\max(0, \gamma + \delta - 1)$. They remain dual to each other. These forms of "weighting" still apply when the minimum and maximum are changed into product and probabilistic sums respectively. The prioritized max and min combinations are the counterpart in possibility theory of the linear convex combination in probability theory; when $\alpha * \beta = \max(0, \alpha + \beta - 1)$ they take the more familiar form (Dubois and Prade, 1988b) :

$$\pi_{\wedge\alpha} = \min_{j=1,n} \max(\pi_j, 1 - \alpha_j),$$

$$\pi_{\vee\alpha} = \max_{j=1,n} \min(\alpha_j, \pi_j).$$

These prioritized minimum and maximum can be interpreted in the possibilistic framework as a kind of integral, just as the convex combination can be interpreted in terms of probabilistic expectation (see Dubois and Prade, 1986).

Consistency-Driven Prioritized Fusion

Sometimes, the priorities assigned to the sources are just an indication of which source is reliable and which source is not, as in the case of expert opinion pooling. Then, one approach to prioritized fusion is to accept the conjunctive merging of information from a reliable source s_1 and from a less reliable one s_2 as soon as the data coming from the latter is consistent with the former. In case of inconsistency, the information given by the less reliable source is simply discarded. The information from source s_2 is used to refine the information from s_2 insofar as the two sources do not conflict. If π_1 is obtained from s_1 and π_2 from s_2, the following combination rule has been proposed (Dubois and Prade, 1988c; Yager, 1991):

$$\pi_{\widehat{1-2}} = \min(\pi_1, \max(\pi_2, 1 - h(\pi_1, \pi_2))).$$

Note that when the consistency degree $h(\pi_1, \pi_2) = 0$, s_2 contradicts s_1 and only the opinion of s_1 is retained ($\pi_{\widehat{1-2}} = \pi_1$), while if $h(\pi_1, \pi_2) = 1$ then $\pi_{\widehat{1-2}} = \min(\pi_1, \pi_2)$. Here π_1 is viewed as a specific piece of information while π_2 is viewed as a fuzzy default value which is less reliable. The prioritized conjunctive fusion rule is easily interpreted as the conjunctive combination of the information supplied by source s_1 and the information supplied by source s_2, the latter being discounted by a certainty coefficient $h(\pi_1, \pi_2)$. The degree of possibility that source s_2 is wrong is taken as $1 - h(\pi_1, \pi_2)$. $\pi_{\widehat{1-2}}(u)$ evaluates, in the sense of fuzzy logic, the information "x = u is possible for source

s_1 and, if source s_2 is consistent with source s_1, x = u is possible for source s_2". $\pi_{\widehat{1-2}}$ is subnormalized when it differs from $\min(\pi_1,\pi_2)$. Hence a renormalization can be used.

The disjunctive counterpart of this prioritized operator is

$$\pi_{1-2} = \max(\pi_1, \min(\pi_2, h(\pi_1,\pi_2)).$$

The effect of this operator is to truncate the information supplied by the less prioritary source, disjunctively combined with source s_1. Again if the two sources disagree ($h(\pi_1,\pi_2) = 0$) then $\pi_{1-2} = \pi_1$; if $h(\pi_1,\pi_2) = 1$ then $\pi_{1-2} = \max(\pi_1, \pi_2)$. Here, when source s_2 is somewhat consistent with source s_1, values considered possible by source s_2 are considered possible in the whole, even if impossible for s_1.

Other types of weighted combination schemes for fuzzy sets exist. For instance, Kuncheva and Krisnapuram (1996) have embedded standard aggregation operations into a more elaborated fusion scheme where the degree of consistency between sources intervenes and is compared to a threshold above which consensual opinions reinforce, and under which they attenuate each other.

4.3 Quantified Adaptive Fusion

The renormalization applied to conjunctive operators erases the possible conflict between the sources and creates numerical instability problems in case of strong conflicts. It is sensible that the result of the combination should focus on the values on which all the sources partially agree (in the sense that none of them gave to these values a possibility equal to 0). But it would be better if the result could also keep track of the conflict in some way. A natural idea for keeping track of a partial conflict is to discount the renormalized result by a factor corresponding to the lack of normalization, i.e., $h(\pi_1,\pi_2)$. Namely $1 - h(\pi_1,\pi_2)$ is viewed as the degree of possibility that the result of the combination is wrong, while, when $h(\pi_1,\pi_2) = 1$, it is useful and reasonable to suppose that the two sources are right. The amount of conflict $1 - h(\pi_1,\pi_2)$ induces a uniform level of possibility for all values outside the ones emerging in the renormalized intersection of π_1 and π_2. It expresses that the result of a conflicting combination is not fully certain. If π results from the normalized conjunctive fusion of π_1 and π_2, then this result is discounted into $\pi' = \max(\pi, 1 - h(\pi_1,\pi_2))$. Clearly, in the limit, when the conflict is total, this procedure yields the totally ignorant possibility distribution.

When $h(\pi_1,\pi_2)$ is low, one could be more optimistic and assume that the discrepancy between the sources is due to one of them being wrong, not both. Then, instead of transferring the amount of conflict as a uniform degree of possibility on the whole referential U, one may restrict to the union of π_1 and π_2, considering that there is no reason to put a non-zero possibility degree on values that both sources agree to consider as impossible. This amounts to replacing $1 - h(\pi_1,\pi_2)$ by $\min(\max(\pi_1(u), \pi_2(u)), 1 - h(\pi_1,\pi_2))$ in the above expression. For details see Dubois and Prade (1992a), Dubois *et al.* (1999). It gives, for the minimum and maximum fusion rule, the adaptive fusion rule at the bottom of this page.

An obvious extension of this scheme to $m > 2$ sources would mean changing $\min(\pi_1,\pi_2)$ and $\max(\pi_1,\pi_2)$ into $\min(\pi_1,\ldots, \pi_k)$, and $\max(\pi_1,\ldots, \pi_m)$ respectively. But this extension, if natural, will not be efficient because it only considers a trade-off between two assumptions "all sources are right" and "one source is right". Clearly the more sources, the more likely they will supply scattered information so that most of the time, the agreement index $h(\pi_1,\pi_2,\ldots, \pi_m) = \sup_u \min_i \pi_i(u) = 0$, while $\max_i \pi_i$ will yield a very uninformative result, i.e., the adaptive operator will always behave disjunctively. Other intermediary assumptions about the sources can be envisaged as considered now.

$$\pi_{AD}(u) = \max\left(\frac{\min(\pi_1(u), \pi_2(u))}{h(\pi_1, \pi_2)}, \min(\max(\pi_1(u), \pi_2(u)), 1 - h(\pi_1, \pi_2))\right)$$

A first idea to cope with this situation is to assume the number of reliable sources in S is fixed and to use quantified fusion. An intermediary mode of pooling the π_i's ranging between the conjunctive and the disjunctive modes consists in assuming that j sources out of n = |S| are reliable. The pooling method then consists in selecting a subset $J \subseteq S$ of sources such that |J| = j, assume that they are reliable and combine their opinions conjunctively. Then, considering that it is not known which one of these subsets J contain reliable sources, the intermediary results must be combined disjunctively. The following extension of (3) is obtained (Dubois, Prade and Testemale, 1988).

$$\pi^{(j)}(u) = \max_{J \subseteq S,\ |J|=j} \min_{i \in J} \pi_i(u).$$

Clearly, $\pi^{(m)} = \pi_\wedge$ (for * = min) and $\pi^{(1)} = \pi_\vee$ (for $\perp$ = max) i.e., this mode of aggregation subsumes the conjunctive and disjunctive ones. The above combination rule is equivalent to some quantified aggregation functions also proposed by Yager (1985), and can be easily calculated for each value of u as an order-statistic. This scheme can be extended to fuzzy quantifiers, in order to model assumptions such as "most sources are reliable", "approximately j sources are reliable", etc. (Dubois *et al.*, 1988; Yager, 1985).

The choice of the number j of supposedly reliable sources can be guided by the expert assessment step for expert opinion pooling, or by prior knowledge about sensors. However guessing the value of j is not easy from scratch. A more interesting idea is to extend the adaptive operator on the basis of a consistency analysis and to derive two values j^- and j^+ representing a pessimistic and an optimistic evaluation of the number of reliable sources (Dubois and Prade, 1994). Let $T \subseteq S$ be a subset of sources and $h(T) = \sup_u \min_{i \in T} \pi_i(u)$ be the consistency index among sources in T. Define

$$j^- = \sup\{|T|, h(T) = 1\}\ ;\ j^+ = \sup\{|T|, h(T) > 0\}.$$

Then it is plausible to assume that at least j^- sources in S are reliable (since there is a fully agreeing subset of j^- sources) and at most j^+ are reliable (since there is a total conflict within all groups of more than j^+ sources). It can be shown that generally $j^- \leq j^+$. The optimistic (resp. pessimistic) fusion is achieved by the quantified operator $\pi^{(j-)}$ (resp. $\pi^{(j+)}$). So, a natural extension of the adaptive operator to the case of m > 2 sources is

$$\pi_{AD}(u) = \max\left(\frac{\pi^{(j^+)}(u)}{h(j^+)},\ \min\left(\pi^{(j^-)}(u),\ 1 - h(j^+)\right)\right)$$

where $h(j^+) = \max\{h(T), |T| = j^+\}$. It can be checked that when |S| = 2, if $h(\pi_1,\pi_2) = 1$, then $j^+ = j^- = 2$ and $\pi_{AD} = \min(\pi_1,\pi_2)$. If $h(\pi_1,\pi_2) \in (0,1)$, $j^- = 1$, $j^+ = 2$, $\pi^{(1)} = \max(\pi_1,\pi_2)$, $\pi^{(2)} = \min(\pi_1,\pi_2)$. Moreover, in the case of set-valued information, it is easy to check that $j^+ = j^-$ is the maximal number of sources that do not conflict with one another, so $h(j^+) = 1$. The quantified adapted rule then yields the logical fusion rule (2) based on cardinality-maximal consistent subset of sources.

The resulting distribution may have a complex shape; however it highlights which values are most plausible for the parameter. Moreover, one may simplify the shape of the resulting distribution by using fuzzy convex hulls. This adaptive fusion rule remains partially ad hoc, and is in need of formal justification.

Variants can also be thought of where minimum is changed into another triangular norm, and the discounting method is different, keeping the same approach. Some authors have noticed that the values of j^+ and j^- are not robust to small changes in the distributions. For instance when going from triangular to trapezoidal distributions, j^- may increase, and the resulting distribution may drastically change (See Pannerec *et al.*, 1998). This defect can be mended by changing j^+ and j^- into fuzzy integers and using fuzzy quantified fusion to compute substitutes of $\pi^{(j+)}$ and $\pi^{(j-)}$.

4.4 Fusion of possibility distributions using maximal consistent subsets of sources

One may alternatively extend the logical conflict analysis in terms of maximal consistent subsets to level-cuts of the possibility distributions. Let $E^\alpha_1, E^\alpha_2, \ldots E^\alpha_n$ be the strong α-cuts of π_1, π_2, ..., π_n. Namely, $E^\alpha_i = \{u, \pi_i(u) > \alpha\}$. At each level α, the fusion rule (2) can be applied, leading to $E^\alpha = \bigcup_{j = 1, \ldots p(\alpha)} \bigcap_{i \in K^\alpha_j} E^\alpha_i$ where K^α_1, $K^\alpha_2, \ldots, K^\alpha_{p(\alpha)}$ are the maximal consistent subsets of sources at possibility level α. Unfortunately, a possibility distribution whose level-cuts are $\{E^\alpha,\ \alpha < 1\}$ generally does not exist, even if one may compute a possibility distribution π of the form

$$\pi(u) = \sup\{\alpha, u \in E^\alpha\}. \qquad (4)$$

The reason is that the family $\{E^\alpha,\ \alpha < 1\}$ is generally not nested. This family cannot then be recovered from the resulting π. Assuming the possibility distributions are continuous unimodal membership functions on the real line, there will exist a finite number of k thresholds $0 = \beta_0 < \beta_1 < \ldots < \beta_k < \beta_{k+1} = 1$, such that E^α results from the same family of maximal consistent sources. Let $\alpha \in (\beta_i, \beta_{i+1}]$ for some i, and $\{K^i_1, K^i_2, \ldots, K^i_{p(i)}\}$ be the family of maximal consistent sources for α-cuts, $\alpha = (\beta_i, \beta_{i+1}]$. In this range, the family $\{E^\alpha, \alpha \in (\beta_i, \beta_{i+1}]\}$ remains nested.

To find the thresholds, one may use the following procedure:

From i = 0 to k do:

1) $\alpha = 0 : i = 1$.
2) Find the family of maximal consistents subsets of sources $K^i_1, K^i_2, \ldots, K^i_{p(i)}$ for the strong α-cuts of the possibility distributions. Let $h(K^i_j)$ be the degree of consistency of the conjunction of possibility distributions supplied by sources in K^α_j.
3) Let $\beta_i = \min (h(K^i_1), \ldots, h(K^i_{p(i)}))$.
3) If $\beta_i < 1$, $\alpha = \beta_i$; i = i+ 1. Go to 2.

Equation (4) can be applied to $\alpha \in (\beta_i, \beta_{i+1}]$ and yields a genuine fuzzy set, not normalized since its height is β_{i+1}. Let F^i be this fuzzy set once renormalized (dividing its membership function by its height). It then makes sense to represent the result of the fusion by a random fuzzy set defined by the mass function $m(F^i) = \beta_{i+1} - \beta_i$ for i = 0,... k. What is obtained is a belief function with fuzzy focal elements, which has been already considered in the literature see Yen (1992), Denoeux (2000) for instance.

4.5 Fusion under prior information.

A counterpart of Bayes theorem holds for possibilistic conditioning, namely

$$\pi(u_m \mid u) * \pi_x(u) = \pi_x(u \mid u_m) * \pi(u_m)$$

where $\pi_x(u)$ is the a priori knowledge about x, $\pi(u_m \mid u)$ the possibility distribution describing the range of possibly observed values when x = u. By definition, notice that the quantity $\pi(u_m) = \max_{u'} \pi(u_m \mid u') * \pi_x(u')$. Operation * denotes min or product according to whether the setting is ordinal or numerical. Assume no a priori knowledge about x, that is, $\pi_x(u) = 1$, $\forall u$. Then the possibilistic Bayes theorem becomes

$$\pi(u_m \mid u) = \pi_x(u \mid u_m) * \max_{u'} \pi(u_m \mid u').$$

If * = min, one may let $\pi_x(u \mid u_m) = \pi(u_m \mid u)$ in accordance with the use of likelihood functions for abductive reasoning provided that $\max_u \pi(u_n \mid u) = 1$ (observational relevance). A qualitative normalization is possible which consists in moving to 1 the greatest possibility degrees and leaving the other degrees unchanged.

If * = product, one gets

$$\pi_x(u \mid u_m) = \frac{\pi(u_m \mid u)}{\max_{u'} \pi(u_m \mid u')}.$$

When there is some prior information $\pi_x(u)$, the possibilistic counterpart of Bayes updating procedure is

$$\pi_x(u \mid u_m) = \frac{\pi_x(u) * \pi(u_m \mid u))}{\sup_{u' \in U} \pi_x(u') * \pi(u_m \mid u'))}.$$

When there are several sources yielding n possibility distributions π_1 , π_2 ..., π_m and some prior information $\pi_x(u)$, the above updating scheme can be applied, letting $\pi(u_m \mid u) = f(\pi_1(u), \pi_2(u),\ldots, \pi_m(u))$ for some fusion operation f. It comes down to a first fusion step merging the information coming from the sources, followed by another fusion step using the normalized product rule to merge the result of the first step and the prior information. This is very similar to Bayesian fusion, but for the type of renormalization. Moreover, this scheme is more flexible because more freedom is granted for combining the likelihood terms, and because it also includes the case of no prior information in a natural way.

5 Conclusion

The possibilistic framework for data fusion makes sense when it seems difficult to represent the information supplied by the sources in a regular probabilistic setting, due to linguistic imprecision and/or lack of statistical evidence. It is more versatile than the probabilistic setting for the expression of various fusion modes, as it encompasses both logical and statistical fusion methods. Regarding the conjunctive and disjunctive pooling modes that possibility theory provides, they are particularly suitable when sources are heterogeneous, (i.e., cannot be viewed as instances of a single random source) and supply information under several formats, such as numerical and linguistic. Most fusion operations on numerical scales can fit in the fuzzy/possibilistic representation framework.

Possibilistic data fusion can also be used to cope with inconsistency in multiple source databases using notions of maximal consistent subbases expressed in syntactic representations (see Benferhat *et al.* 1997, 2000).

References

Abidi M.A. and Gonzalez R.C., eds. (1992) *Data Fusion in Robotics and Machine Intelligence* . Academic Press.

Baral C., Kraus S., Minker J. and Subrahmanian (1992). Combining knowledge bases consisting in first order theories, *Comput. Intelligence*, **8**(1), 45-71.

Benferhat S., Dubois D. and Prade H. (1997). From semantic to syntactic approaches to information combination in possibilistic logic, *Aggregation and Fusion of Imperfect Information* (Bouchon-Meunier B., Ed.), Physica-Verlag, Heidelberg, Germany, 141-161.

Benferhat S., Dubois D. and Prade H. (2000) An overview of possibilistic logic and its application to nonmonotonic reasoning and data fusion. In Della Riccia, G, et al. Eds, *Computational Intelligence and Data Mining*, Springer Verlag, Berlin, 69-93.

Bezdek J., Keller J., Krishnapuram R. and Pal N. R. (1999). *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*, Kluwer Acad. Publ., Dordrecht.

Bloch I. (2001) Fusion of image information under imprecision and uncertainty: numerical methods. In "*Data Fusion and Perception*" (G. Della Riccia, et al., Eds) SpringerVerlag, Berlin, 135-170

Bloch I., Hunter A. (2001) Data and Knowledge Fusion, Special Issue of the Int. J. Intelligent Systems, 16, N° 10 and 11, J. Wiley, New York.

Boldrin L. and Sossai C. (1995). An algebraic semantics for possibilistic logic, 11th Conf. on Uncertainty in AI, Morgan Kaufmann, San Francisco, CA, 27-35.

Bouchon-Meunier B., Ed. (1997). *Aggregation and Fusion of Imperfect Information*, Physica-Verlag.

Buchanan B. G. and Shortliffe E. H. (1984). *Rule-Based Expert Systems*, Addison-Wesley.

Cooke R. M. (1991). *Experts in Uncertainty*, Oxford University Press

De Cooman G., Aeyels D. (1999). Supremum-preserving upper probabilities. *Information Sciences*,118 ,173 –212.

Denoeux T. (2000) Modeling vague beliefs using fuzzy-valued belief structures, *Fuzzy Sets and Systems*, 116, 167-200.

Dubois D., Moral S. and Prade H. (1997). A semantics for possibility theory based on likelihoods, *J. of Mathematical Analysis and Applications*, **205**, 359-380.

Dubois D. and Prade H. (1986). Weighted minimum and maximum operations in fuzzy set theory, *Information Sciences*, **39**, 205-210.

Dubois D. and Prade H. (1988a). Representation and combination of uncertainty with belief functions and possibility measures, *Computational Intelligence.*, **4**, 244-264.

Dubois D. and Prade H. (1988b). *Possibility Theory* Plenum Press, New York.

Dubois D. and Prade H. (1988c). Default reasoning and possibility theory, *Artificial Intell.*, **35**, 243-257.

Dubois D. and Prade H. (1989). Fuzzy sets, probability and measurement, *Europ. J. of Operational Research*, **40**, 135-154.

Dubois D. and Prade H. (1992a). Combination of fuzzy information in the framework of possibility theory, *Data Fusion in Robotics and Machine Intelligence* (Abidi M.A. and Gonzalez R.C., eds.), 481-505. Academic Press.

Dubois D. and Prade H. (1992b). When upper probabilities are possibility measures, *Fuzzy Sets and Systems*, **49**, 65-74.

Dubois D. and Prade H. (1994). Possibility theory and data fusion in poorly informed environments, *Control Engineering Practice*, **2**(5), 811-823.

Dubois D., Prade H. and Sandri S. (1993). On possibility/probability transformations, In Lowen R. and Lowen M., eds., *Fuzzy Logic: State of the Art*, Kluwer Academic Publ., 103-112.

Dubois D., Prade H. and Testemale C. (1988). Weighted fuzzy pattern matching, *Fuzzy Sets and Systems*, **28**, 313-331.

Dubois D., Prade H. and Yager R. R. (1999). Merging fuzzy information, In Bezdek J.C., Dubois D., Prade H., Eds., *Fuzzy Sets in Approximate Reasoning and Information Systems*, Kluwer, Dordrecht, 335-401.

Edwards W. F. (1972). *Likelihood*, Cambridge University Press, Cambridge, U.K.

Fodor J. and Yager R. (2000). Fuzzy set-theoretic operators and quantifiers, In *Fundamentals of Fuzzy Sets*, Dubois D. and Prade H., eds., Kluwer;Dordrecht, 125-193

Gebhardt J. and Kruse R. (1998). Parallel combination of information sources, In Dubois D. and Prade H., Eds, *Belief Change*, Kluwer Acad. Publ., Dordrecht, 393-440.

Kuncheva L. and Krishnapuram R. (1996). A fuzzy consensus operator, *Fuzzy Sets and Systems*, 79, 347-356.

Lehrer K, Wagner C.(1981) *Rational Consensus in Science and Society*, Reidel, Dordrecht.

Mundici D. (1992). The logic of Ulam games with lies, Knowledge, In Bicchieri C., Dalla Chiara M., eds., *Belief and Strategic Interaction*, Cambridge Univ. Press, 275-284.

Pannerec T., Oussalah M., Maaref H. and Barret. C. (1998). Absolute localization of a miniature mobile robot using heterogeneous sensors: comparison between Kalman filter and possibility theory method, *Proc. IEEE Symposium on Robotics and Cybernetics*, CESA'98, Tunis.

Poole D. L. (1985). On the comparison of theories: preferring the most specific explanation, *Proc. 9th Int. Joint Conf. on Artif. Intelligence*, Los Angeles, 465-474.

Rescher N. , Manor R. (1970) On inference from inconsistent premises. *Theory and Decision*, 1, 179-219.

Shafer G. (1976). *A Mathematical Theory of Evidence*, Princeton Univ. Press, Princeton, NJ.

Yager R. R. (1984). Approximate reasoning as a basis for rule-based expert systems, *IEEE Trans. on Systems, Man and Cybernetics*, 14, 636-643.

Yager R. R. (1985). Aggregating evidence using quantified statements, *Information Sciences*, **36**, 179-206.

Yager R. R. (1991). Non-monotonic set-theoretic operators, *Fuzzy Sets and Systems*, **42**, 173-190.

Yen J. (1992). Computing generalized belief functions for continuous fuzzy sets, *International Journal of Approximate Reasoning*, **6**, 1-31.

Zadeh L.A. (1965). Fuzzy sets, *Information Control*, **8**, 338-353.

Zadeh L. A. (1978). Fuzzy sets as a basis for a theory of possibility, *Fuzzy Sets and Systems*, **1**, 3-28.

Iterated revision and the axiom of recovery: a unified treatment via epistemic states

Samir Chopra
schopra@cse.unsw.edu.au
Knowledge Systems Group
University of New South Wales
Sydney, NSW 2052, Australia

Aditya Ghose
aditya@uow.edu.au
Department of Information Systems
University of Wollongong
Wollongong, NSW 2522, Australia

Thomas Meyer
tmeyer@cs.up.ac.za
Computer Science Department
University of Pretoria
Pretoria 0002, South Africa

Abstract

The axiom of recovery, while capturing a central intuition regarding belief change, has been the source of much controversy. We argue briefly against putative counterexamples to the axiom—while agreeing that some of their insight deserves to be preserved—and present additional recovery-like axioms in a framework that uses epistemic states, which encode preferences, as the object of revisions. This provides a framework in which iterated revision becomes possible and makes explicit the connection between iterated belief change and the axiom of recovery. We provide a representation theorem that connects the semantic conditions that we impose on iterated revision and the additional syntactical properties mentioned. We also show some interesting similarities between our framework and that of Darwiche-Pearl [4]. In particular, we show that the intuitions underlying the controversial (C2) postulate are captured by the recovery axiom and our recovery-like postulates (the latter can be seen as weakenings of (C2).

1 Introduction

A particularly simple sequence of belief change in reasoning agents is that of giving up and then adopting the same belief ("I believed I had money for the movies, but then realized I had left my wallet at home. However, a few minutes later, I discovered a twenty in my pocket and regained my belief that I had enough money for the movies"). The axiom of recovery in the AGM framework [1] attempts to place a rationality constraint on the form of such a change. It states that expansion by a belief recovers any beliefs lost by the previous contraction by that belief. The status of the axiom of recovery has been a source of much controversy in belief revision [7, 8, 9, 12]. There are well-known counterexamples to recovery, with the most convincing ones amongst these being Hansson's Cleopatra and George-the-criminal examples [8, 10]. The following is a slightly amended version of the former:

> I believe that 'Cleopatra had a son' (ϕ) and that 'Cleopatra had a daughter' (ψ), and thus also that 'Cleopatra had a child' ($\phi \vee \psi$). Then I receive information that Cleopatra had no children, which makes me give up my belief in $\phi \vee \psi$. But then I am told that Cleopatra did have children, and so I add $\phi \vee \psi$. But I should not regain my belief in either ψ or ϕ as a result.

One response to this situation is to isolate a class of belief change operators that do not satisfy recovery i.e., the so-called withdrawal operators [13]. We do not adopt this approach for a couple of reasons. Firstly, withdrawal operators violate the principle of minimal change [10]. As an example, consider the operator $\dot{-}$ defined as follows (K is a belief set closed under logical consequence, α an arbitrary epistemic input): if $\alpha \notin K$, then $K \dot{-} \alpha = K$, otherwise, $K \dot{-} \alpha = Cn(\emptyset)$. A fundamental intuition behind minimal contraction, the principle of core-retainment[1], is only satisfied by withdrawal operators if they satisfy the recovery axiom as well. This should reinstate our faith in the recovery axiom since it is hard to find a satisfactory alternative formalization of the intuition that beliefs that do not contribute to K implying α should be retained in $K \dot{-} \alpha$. So, while the counterexamples do tickle our intuitions, it is equally the case that there is an important intuition about rational belief change that the recovery postulate captures. Indeed, the recovery postulate is best thought of as a version of the

[1] The principle of core-retainment states that if $\beta \in K$ and $\beta \notin K \dot{-} \alpha$ then there is a set K' such that $K' \subseteq K$ and that $\alpha \notin Cn(K')$ but $\alpha \in Cn(K' \cup \{\beta\})$; it requires of an excluded sentence β that it in some way contribute to the implication of α from K.

principle of minimal change: so much of the original belief state is retained on contraction that the original belief state can simply be restored on adopting the same belief. Our approach to this situation is that even if the original postulate is rejected as being too permissive, *some* recovery like postulates must constrain belief revision if the principle of minimal change is to be taken seriously. Furthermore, recovery follows from other highly plausible postulates such as closure, inclusion, vacuity, success, extensionality and core-retainment [10]. Significantly, there is a clear and intimate connection between iterated revision and the recovery axiom: we can view the axiom as specifying the form of the iterated revision that should take place when contracting and revising by the same belief. In what follows, we will make this connection clearer.

But what about the counterexamples? Surely, they point to counterintuitive scenarios arising from the adoption of the recovery axiom? We argue that, underlying these examples is an assumption that information leading to the specified sequence of contraction and expansion is not received from the same source. That is, our claim is that recovery should always hold when restricted to the case where information is obtained from the same source, but that it need not hold when information is obtained from different sources. Consider the Cleopatra counterexample. The agent believes both ϕ and ψ originally, and as a result is committed to the belief that $\phi \vee \psi$. Now the agent receives information to the effect that $\neg(\phi \vee \psi)$. Crucially, what is left out of this example is details about the sources of the epistemic inputs. If source S_1 provides the *reasons* for believing $\neg(\phi \vee \psi)$ and source S_2 provides the reason for believing $\phi \vee \psi$ then it makes sense to think that the agent does not recover its original beliefs in ϕ or ψ. However, if it is the same source that provides information on both $\neg(\phi \vee \psi)$ and $\phi \vee \psi$, then why should the agent not regain its belief in ϕ and ψ? After all, source S_1 provided the reason for the agent dropping its belief in ϕ and ψ in the first place. If it then supplies information to the contrary, the agent's reasons for dropping those beliefs have been negated, and it should regain its original beliefs. To do otherwise would be counterintuitive. If however it is another source that provides the new information, then the agent's original reasons for contracting by ϕ and ψ remain unaffected and there is no reason for it to start believing ϕ or ψ again. (For a similar though crucially different response see [16]).

The issue of what happens when information is obtained from different sources is interesting in its own right, and deserves to be treated separately. In general our attitude is that the intuitions behind the recovery axiom are worth capturing: it attempts to place rational constraints on what happens when we revise and contract by the same formula. This sort of belief change is commonplace and must be handled by any adequate formal framework.

1.1 Our Proposal

We will consider versions of postulates in the same spirit as recovery. We argue that to do so, a shift to belief change on epistemic states, in the Darwiche-Pearl spirit is necessary, since we need a framework in which to talk about iterated revision. Cantwell [3] also provides recovery-like properties in the context of iterated revision, but these however restate recovery itself in terms of revision (where contracting with α is replaced by a revision with $\neg\alpha$). This is done to show that the counterexamples to recovery are not only a criticism of AGM contraction (as has been argued in the past), but also a criticism of AGM revision. Cantwell goes on to show that examples similar to the Cleopatra and George-the-criminal examples can be constructed for iterated revision as well.

While adopting the representational framework of epistemic states, we do not accept all the Darwiche-Pearl postulates. There is sufficient debate in the belief revision literature on the appropriateness of these postulates. In principle, though, we are of the opinion that the 3rd and 4th Darwiche-Pearl postulates are valid. Like others we feel that the 2nd postulate is too strong. The results in this paper provide a weaker, and, we think, acceptable alternative to the 2nd postulate. We are also of the opinion that the 1st Darwich-Pearl postulate is too strong ([14] provides examples to back up this claim). We adopt the basic setting in which belief change is performed on epistemic states, from which a total preorder on valuations and a knowledge base can be extracted. We provide a set of reformulated AGM postulates for belief change on epistemic states and *insist* on these.

We present some recovery-like postulates, as well as restrictions on the way in which the orderings extracted from epistemic states may be modified when revision and contraction take place, and provide a representation theorem that connects the recovery-like postulates and the postulates on orderings. It turns out that the recovery-like postulates, when combined, can be thought of as a weakened version of the (C2) postulate of Darwiche-Pearl. This is brought out clearly when the postulates on orderings are considered. The link between recovery and the (C2) postulate is interesting and surprising. This makes it possible to think of (C2) as having overstated the case and of the recovery postulate and our weakenings as having addressed its problems.

1.2 Notation and basic definitions

We assume a finitely generated propositional language L closed under the usual propositional connectives and equipped with a classical model-theoretic semantics; the constants $\top, \bot$ are in L. V is the set of valuations of L

and $M(\alpha)$ is the set of models of $\alpha \in L$. Classical entailment is denoted by $\models$. Roman letters, $p, q, r, \ldots$ denote propositional atoms; Greek letters $\alpha, \beta, \ldots$ stand for arbitrary formulas. We reserve the letter Φ to denote epistemic states. $M_{\preceq_\Phi}(\alpha)$ denotes the minimal models of α in the total preorder on valuations associated with the epistemic state Φ.

Definition 1 *Associated with an* epistemic state Φ *is a total preorder on valuations* $\preceq_\Phi$*, and a knowledge base* $K(\Phi)$*.* $M_{\preceq_\Phi}(\alpha)$ *denotes the minimal models of* α *in the total preorder on valuations. The knowledge base associated with the epistemic state is obtained by considering the minimal models in* $\preceq_\Phi$ *i.e.,* $M(K(\Phi)) = M_{\preceq_\Phi}(\top)$.

ϕ represents the set of all wffs entailed by ϕ (the theory obtained from the set of minimal models in $\preceq_\Phi$). Observe that the knowledge bases extracted from Φ are all logically equivalent. We will often abuse notation by using $K(\Phi)$ to refer to the *the* knowledge base extracted from Φ. The intention is that $K(\Phi)$ is some canonical representative of *all* the knowledge bases extracted from Φ.

1.3 The reformulated AGM postulates

In the reformulated postulates below, $*$ and $\dot{-}$ are belief change operations on epistemic states, not knowledge bases. So $*$ takes an epistemic state and a sentence and produces an epistemic state. For $\dot{-}$ and $*$ to satisfy the AGM postulates means that they satisfy the reformulated AGM postulates which apply to epistemic states, not knowledge bases. The reformulated AGM postulates guarantee a unique extracted knowledge base when revision or contraction is performed i.e., the lowest level of valuations in the resulting epistemic state is fixed. What is not fixed is how to order the remaining valuations. Note that the object of revision is the epistemic state, but in stating the postulates we specify the form of the knowledge base extracted from the epistemic state. Here are the reformulated AGM postulates. First contraction:

- $(\Phi{-}1)$: $K(\Phi - \alpha) = Cn(K(\Phi - \alpha))$
- $(\Phi{-}2)$: $K(\Phi - \alpha) \subseteq K(\Phi)$
- $(\Phi{-}3)$: If $\alpha \notin K(\Phi)$ then $K(\Phi - \alpha) = K(\Phi)$
- $(\Phi{-}4)$: If $\not\models \alpha$ then $\alpha \notin K(\Phi - \alpha)$
- $(\Phi{-}5)$: If $\alpha \equiv \beta$ then $\Phi - \alpha = \Phi - \beta$
- $(\Phi{-}6)$: If $\alpha \in K(\Phi)$ then $K(\Phi) \subseteq (K(\Phi - \alpha)) + \alpha$
- $(\Phi{-}7)$: $K(\Phi - \alpha) \cap K(\Phi - \beta) \subseteq K(\Phi - (\alpha \wedge \beta))$
- $(\Phi{-}8)$: If $\beta \notin K(\Phi - (\alpha \wedge \beta))$ then $K(\Phi - (\alpha \wedge \beta)) \subseteq K(\Phi) - \beta$

In what follows, we will be particularly interested in the relationship between $K(\Phi * \alpha - \alpha)$ and $K(\Phi)$. We will show that equality between the two sides conflicts with the reformulated AGM postulates but does hold under some conditions.

The intuitions corresponding to the postulates are roughly the same as those underlying the original AGM postulates. For example, $(\Phi{-}1)$ states that the knowledge base associated with the revised epistemic state is closed under logical consequence. $(\Phi{-}5)$ states that contracting by logically equivalent formulas results in the same epistemic state. This particular postulate highlights a difference between the original AGM postulates and our reformulations. The original AGM postulate requires the belief set after revision to be the same after revisions by logically equivalent formulas, whereas we require that if two epistemic states are the same, then revisions by logically equivalent formulas should result in the same epistemic state. This is crucially different from merely requiring that the knowledge base associated with the epistemic state be the same (such a reformulation of the original AGM axioms by Darwiche-Pearl is responsible for making (C2) compatible with them). Note that we include the recovery axiom above.

The following are the reformulatedreformulated AGM postulates for revision:

- $(\Phi{*}1)$ $K(\Phi * \alpha) = Cn(K(\Phi * \alpha))$
- $(\Phi{*}2)$ $\alpha \in K(\Phi * \alpha)$
- $(\Phi{*}3)$ $K(\Phi * \alpha) \subseteq K(\Phi) + \alpha$
- $(\Phi{*}4)$ If $\neg\alpha \notin K(\Phi)$ then $K(\Phi) + \alpha \subseteq K(\Phi * \alpha)$
- $(\Phi{*}5)$ If $\alpha \equiv \beta$ then $\Phi * \alpha = \Phi * \beta$
- $(\Phi{*}6)$ $\bot \in K(\Phi * \alpha)$ iff $\models \neg\alpha$
- $(\Phi{*}7)$ $K(\Phi * (\alpha \wedge \beta)) \subseteq K(\Phi * \alpha) + \beta$
- $(\Phi{*}8)$ If $\neg\beta \notin K(\Phi * \alpha)$ then $K(\Phi * \alpha) + \beta \subseteq K(\Phi * (\alpha \wedge \beta))$

As with the contraction postulates, the intuitions corresponding to the postulates are roughly the same as those underlying the original AGM postulates. For example, $(\Phi{*}1)$ states that the knowledge base associated with the revised epistemic state is closed. $(\Phi{*}6)$ states that an inconsistent knowledge base only results when revising by contradictions (note the modified $(\Phi * 5)$ postulate as well).

For the sake of completeness, we include the Darwiche-Pearl postulates for iterated revision [4] reformulated for our framework. In the four postulates below $\circ$ is the update operator, α, μ, represent new epistemic inputs and Φ represents an epistemic state.

(C1) If $\alpha \models \mu$, then $K(\Phi \circ \mu \circ \alpha) = K(\Phi \circ \alpha)$.

(C2) If $\alpha \models \neg\mu$, then $K(\Phi \circ \mu \circ \alpha) = K(\Phi \circ \alpha)$.

(C3) If $K(\Phi \circ \alpha) \models \mu$, then $K(\Phi \circ \mu \circ \alpha) \models \mu$.

(C4) If $K(\Phi \circ \alpha) \not\models \neg\mu$, then $K(\Phi \circ \mu \circ \alpha) \models \mu$.

The postulate (C1) is a more powerful version of the $(\Phi * 7)$ and $(\Phi * 8)$ postulates (it implies them); it states that when two pieces of information (one more specific than the other) arrive, the first is made redundant by the second. (C2) says that when two contradictory epistemic inputs arrive, the second one prevails; the second evidence alone yields the same belief state. Here the *prima facie* connection with recovery should be obvious; for the basic form of the recovery axioms deals with 'contract by α and then expand by α' while (C2) deals with 'revise by α and then revise by (effectively) $\neg\alpha$'. The latter is clearly stronger. (C3) says that a piece of evidence μ should be retained after accommodating more recent evidence α that entails μ given the current belief state. (C4) simply says that no epistemic input can act as its own defeater. Arlo-Costa and Parikh [2] and Lehmann [11] have critically commented on (C2) as have Freund and Lehmann [6] who have shown that it is inconsistent with the original AGM axioms for belief sets (as is the weaker axiom, $C2'$ proposed in Nayak *et. al.* [15]). This last objection, as noted above, is no longer a problem when the postulates are reformulated for epistemic states.

2 The new recovery postulates

In this section we provide additional recovery-like postulates and then provide a semantic condition that provides the means with which to carry out iterated revision. These additional properties can be viewed as desirable properties for iterated revision and cover a variety of situations, ranging from sequences of revisions and contractions by the same formula to sequences of revisions and contractions by a formula and its negation. In particular these properties describe the conditions under which we can expect stability or minimal loss of beliefs in the original epistemic state. Note that in all of these properties the sequence of belief changes reverses that in the original formulation of the recovery axiom where contraction is followed by expansion. Stating the postulates in this form enables the connection with iterated revision to become clear since it is in the case of revision followed by contraction that a notion of iterated revision is necessary (in the original formulation of the recovery axiom, expansion is equivalent to revision thus obviating the need for a framework that requires iteration). In the postulates we make the implicit assumption that information is received from the same source.

- (R1) $K(\Phi * \alpha - \alpha) \subseteq K(\Phi - \alpha)$
- (R2) $\alpha, \neg\alpha \notin K(\Phi)$ implies $K(\Phi) \subseteq K(\Phi * \alpha - \alpha)$
- (R3) $\alpha \notin K(\Phi)$ implies $K(\Phi) \subseteq K(\Phi * \alpha * \neg\alpha)$
- (R4) $\alpha \in K(\Phi)$ implies $K(\Phi - \alpha) \subseteq K(\Phi * \alpha - \alpha)$

(R1) says that the result of revising an epistemic state and then contracting by the same formula is always contained in the knowledge base obtained after simply contracting by the same formula. (If I add the belief that Cleopatra has children and then contract by this belief, the resultant knowledge base should be contained in the knowledge base obtained by my simply contracting by the belief that Cleopatra has children). (R2) says that if neither a formula nor its negation are in the knowledge base associated with an epistemic state then the original base will be contained in that obtained after revision and contraction by the same formula. (R3) says that if a piece of information is not contained in the knowledge base associated with an epistemic state, then a revision by that formula followed by its negation will always include the original knowledge base. (R4) says that if a formula is contained in the original knowledge base then contracting by the same formula will produce a knowledge base that is contained in one obtained by revising and contracting by the same formula.

The following additional properties further place conditions on recovery like situations:

- (R5) $K(\Phi * \alpha - \alpha) \subseteq K(\Phi)$
- (R6) $\alpha \notin K(\Phi)$ implies $K(\Phi) \subseteq K(\Phi * \alpha - \alpha * \neg\alpha)$
- (R7) $\neg\alpha \in K(\Phi)$ implies $K(\Phi) \subseteq K(\Phi * \alpha - \alpha)$
- (R8) $\alpha \in K(\Phi)$ implies $K(\Phi) \subseteq K(\Phi * \alpha - \alpha * \alpha)$
- (R9) $\alpha, \neg\alpha \notin K(\Phi)$ implies $K(\Phi - \alpha) \subseteq K(\Phi * \alpha - \alpha)$

(R5) says that the knowledge base obtained by revising by an input and then contracting by it is contained in the knowledge base associated with the original epistemic state. (R6) says that if a belief is not contained in the original knowledge base, then the knowledge base is contained in the result of revising by a formula, contracting it and then revising by its negation. (R7) says that if a belief is not contained in the original knowledge base, then the original knowledge base is contained in that obtained after revising and contracting by its negation. (R8) says that if a belief is contained in the original knowledge base, then that belief will be preserved under a sequence of revisions which begin with revision followed by contraction and then revision. (R9) says that if the original knowledge base is agnostic about a particular belief then contracting by that belief will result in a knowledge base that is contained in one obtained by revising and then contracting by that belief.

Observation 1

1. *(R3) holds because $K(\Phi)$ is consistent.*
2. *(R9) follows from (R2).*
3. *(R5) follows from (R1) and the reformulated AGM postulates.*
4. *(R6) is the same as (R3), given the reformulated AGM postulates.*
5. *(R7) contradicts $(\Phi - 2)$ and $(\Phi * 2)$.*
6. *(R1) follows from (R5) if $\alpha \notin K(\Phi)$.*
7. *(R8) follows from the reformulated AGM postulates.*

The reformulated AGM postulates for epistemic states and our additional recovery postulates, in our opinion, provide a comprehensive framework for iterated revision which does justice to the intuitions expressed in the original recovery axiom. One of our stated aims is to link up $K(\Phi * \alpha - \alpha)$ and $K(\Phi)$. We do this via (R2), (R4), (R5), (R7) and (R8). And it is (R7) which contradicts the reformulated AGM postulates, as we have seen. Also, (R8) follows from AGM anyway. Another way to put it: if $\alpha, \neg\alpha \notin K(\Phi)$ then $K(\Phi) = K(\Phi * \alpha - \alpha)$. If $\neg\alpha \in K(\Phi)$ then AGM prevents $K(\Phi) = K(\Phi * \alpha - \alpha)$. If $\alpha \in K(\Phi)$ then, since $\alpha \notin K(\Phi * \alpha - \alpha)$ by AGM, it is AGM that prevents $K(\Phi) = K(\Phi * \alpha - \alpha)$.

2.1 Semantic properties

We now provide conditions in semantic terms on revisions of epistemic states. The following lay conditions on the positions of valuations by revision.

- (S1) $M_{\preceq_\Phi}(\neg\alpha) \subseteq M_{\preceq_{\Phi * \alpha}}(\neg\alpha)$
- (S2) $M_{\preceq_{\Phi * \alpha}}(\neg\alpha) \subseteq M_{\preceq_\Phi}(\neg\alpha)$

The semantic properties taken together state an equality between the minimal models of $\neg\alpha$ in the epistemic state prior to revision and after revision. (S1) and (S2) taken together state that these minimal models of $\neg\alpha$ retain their position after revision by α. For ease of statement of Theorem 1 below, we state these properties as two separate containments rather than the implied equality. The property stated here is straightforward. Consider the minimal models of $\neg\alpha$ in the total preorder associated with the epistemic state; these might or might not be included in the minimal models of the total preorder itself. After revision by α, the minimal models of the ordering cannot contain any $\neg\alpha$ models. So the minimal models of $\neg\alpha$ are either demoted in the ordering or stay where they are. Whatever be the case, no models of $\neg\alpha$ can be promoted in the ordering to join the old minimal models of $\neg\alpha$ and furthermore, none of the minimal models of $\neg\alpha$ are demoted. Revision by α can increase the plausibility of α and decrease that of $\neg\alpha$; it certainly cannot increase the plausibility of $\neg\alpha$. Remarkably, this simple condition provides all the semantic linkage we need with the numerous syntactic properties (R1-6, R8-9) stated above. It should be clear that the semantic properties stated above are a weaker version of the (C2) postulate since in the Darwiche-Pearl framework, which relies on a form of Spohnian conditioning [17], the position of *all* $\neg\alpha$ models is determined in the new epistemic state (via pointwise decrease in their plausibility by one rank after revision by α, thus preserving their relative ordering in the new epistemic state) whereas in our condition, we simply specify the minimal models of $\neg\alpha$ in the new epistemic state. Strengthening these postulates is possible, but possibly counterproductive and in any case, it is not our present concern.

Theorem 1 *Let $*$ and $\dot{-}$ be belief change operations on epistemic states satisfying the reformulated AGM postulates.*

1. *$*$ and $\dot{-}$ satisfy (R1) iff $*$ satisfies (S1).*
2. *$*$ and $\dot{-}$ satisfy (R2)-(R4) iff $*$ satisfies (S2).*

Proof:

1. (S1) follows immediately from (R1). Suppose (S1) and pick a $u \in M(K(\Phi - \alpha))$. If $u \in M(\alpha)$ then $u \in M(K(\Phi * \alpha - \alpha))$ by AGM. If $u \in M(\neg\alpha)$ then $u \in M_{\preceq_\Phi}(\neg\alpha)$. By (S1), $u \in M_{\preceq_{\Phi * \alpha}}(\neg\alpha)$. Therefore $u \in M(K(\Phi * \alpha - \alpha))$.
2. Suppose (S2). Now suppose $\alpha, \neg\alpha \notin K(\Phi)$. Pick a $u \in M(K(\Phi * \alpha - \alpha))$. If $u \in M(\alpha)$ then $u \in M(K(\Phi))$ by AGM. Otherwise $u \in M(K(\Phi))$ by (S2). So (R2) holds. Now suppose $\alpha \notin K(\Phi)$. Pick a $u \in M(K(\Phi * \alpha * \neg\alpha))$. Since $u \in M(\neg\alpha)$ it follows that $u \in M(K(\Phi))$ by (S2). So (R3) holds. Now suppose $\alpha \in K(\Phi)$. Pick a $u \in M(K(\Phi * \alpha - \alpha))$. If $u \in M(\alpha)$ then $u \in M(K(\Phi))$ by AGM. Otherwise $u \in M(K(\Phi))$ by (S2). So (R4) holds. Conversely, suppose (R2)-(R4). If $\alpha, \neg\alpha \notin K(\Phi)$ then (S2) follows from (R2). If $\neg\alpha \in K(\Phi)$ then (S2) follows from (R3). If $\alpha \in K(\Phi)$ then (S2) follows from (R4).

■

The following shows that the case we were interested in, the relationship between $K(\Phi * \alpha - \alpha) = K(\Phi - \alpha)$ is

one of equality in the case when $\neg\alpha$ is not contained in the original knowledge base.

Corollary 1 *From (R1)-(R4) it follows that, if* $\neg\alpha \notin K(\Phi)$ *then* $K(\Phi * \alpha - \alpha) = K(\Phi - \alpha)$.

Proof: Follows from (S1) and (S2), which state together that $M_{\preceq_{\Phi * \alpha}}(\neg\alpha) = M_{\preceq_\Phi}(\neg\alpha)$. ■

Furthermore, note that since and $\dot{-}$ are operations that satisfy the reformulated AGM postulates, it follows that they satisfy (R5), (R6), (R8) and (R9) as well.

2.2 C2 and the new recovery postulates

The connections between (S1), (S2) and (C2) are interesting. Delgrande and Schaub [5] have raised objections to (C2) which rely on the observation that revising a belief state ψ with a sentence of the form $p_1 \wedge p_2 \wedge \ldots \wedge p_n \wedge q$ followed by a revision with $\neg q$ reduces to revision with $\neg q$. Thus the (potentially useful) belief in the conjunct $p_1 \wedge p_2 \wedge \ldots \wedge p_n$ is discarded (unless it was believed in the first place) even though it does not in itself contradict $\neg q$. It can be argued that these criticisms of the (C2) postulate are somewhat unfair, since this unintuitive outcome does not follow if revision by $p_1 \wedge p_2 \wedge \ldots \wedge p_n \wedge q$ is replaced by a sequence of revisions by each of the conjuncts. One would revise with the full conjunction only if these beliefs were somehow implicitly related. One scenario where this behaviour required by the (C2) postulate appears to be fully justified is when a source provides $p_1 \wedge p_2 \wedge \ldots \wedge p_n \wedge q$ as an input, and subsequently changes its mind (thus revising by $\neg q$). In a similar vein, if two consecutive sensor readings contradict each other, it makes more sense to believe the more recent reading, even if the previous reading provided additional information. The (C2) postulate has also been criticized from other perspectives. Cantwell [3] uses a version of the George-the-criminal example to criticize the (C2) postulate. We note that it is possible to argue against Cantwell's criticism along similar lines to our arguments against the Cleopatra example (if the inputs come from the same source, then the outcomes are intuitive, while inputs from different sources would appear as distinct sentences, making the example redundant).

The following example, a variation of the George-the-criminal setting, makes clear that (C2) is too strong, and that (S1) and (S2) are useful alternative weakenings. Assume that we start by believing George is an armed robber. Our friend the police detective tells us that this is incorrect, since no criminal records can be found for George. Subsequently, she corrects her original statement—she did find a criminal dossier on George at police headquarters (it had been misplaced) and given its location, it could have only come off the stack of files for people convicted of illegal gun possession or the stack of convicted shoplifters' dossiers. We must now revise our beliefs with the information that George is not an armed robber, but either a shoplifter or a person convicted of illegal gun possession. We construct below a scenario where the (C2) postulate forces us to believe that George was convicted of illegal gun possession (clearly too strong given the available evidence). We let r denote 'George is an armed robber', g denote 'George has been convicted of illegal gun possession' and s denote 'George is a convicted shoplifter' and use c as an abbreviation for 'George is a criminal' i.e., $r \vee g \vee s$. Given the propositional language $\{r, g, s\}$, we will represent models as sequences of 0's and 1's, representing the valuations of r, g and s respectively (thus 100 represents a model in which r is true and g and s are false). We assume for the sake of explanatory convenience that epistemic states map valuations to natural numbers with the minimal models being identified as those assigned the lowest rank (not necessarily 0)—thus inducing a total preorder on valuations. Let the initial epistemic state Φ_1 be defined as follows:

$$\Phi_1(100) = \Phi_1(101) = \Phi_1(110) = \Phi_1(111) = 0$$
$$\Phi_1(010) = \Phi_1(011) = 1$$
$$\Phi_1(000) = \Phi_1(001) = 2$$

Observe that, next to the models of r, we believe the models of g to be most plausible, reflecting the intuition that if George is not an armed robber, then the next most likely scenario is where George is in illegal possession of firearms. To satisfy (C2) the epistemic state $\Phi_2 = \Phi_1 * \neg c$ must appear as follows:

$$\Phi_2(000) = 0$$
$$\Phi_2(100) = \Phi_2(101) = \Phi_2(110) = \Phi_2(111) = 1$$
$$\Phi_2(010) = \Phi_2(011) = 2$$
$$\Phi_2(001) = 3$$

This leads to the epistemic state $\Phi_3 = \Phi_2 * \neg r \wedge (g \vee s)$ where:

$$\Phi_3(010) = \Phi_3(011) = 0$$
$$\Phi_3(000) = 1$$
$$\Phi_3(100) = \Phi_3(101) = \Phi_3(110) = \Phi_3(111) = 2$$
$$\Phi_2(001) = 3$$

Observe that $g \in K(\Phi_3)$, i.e., we are forced to believe George has been convicted of illegal gun possession. If we relax (C2) with (S1) and (S2), a permissible outcome of revising Φ_1 by $\neg c$ is the epistemic state Φ_2' where:

$$\Phi_2'(000) = 0$$
$$\Phi_2'(100) = \Phi_2'(101) = \Phi_2'(110) = \Phi_2'(111) = 1$$

$\Phi'_2(010) = \Phi'_2(011) = \Phi'_2(001) = 2$

Further revising with $\neg r \wedge (g \vee s)$ gives us the epistemic state Φ'_3 where:

$\Phi'_3(010) = \Phi'_3(011) = \Phi'_3(001) = 0$
$\Phi'_3(000) = 1$
$\Phi'_3(100) = \Phi'_3(101) = \Phi'_3(110) = \Phi'_3(111) = 2$

Notice that $g \notin K(\Phi_3)$.

3 Conclusion

In this paper we have shown how the intuitions underlying the axiom of recovery can be rescued by paying attention to the assumptions underlying putative counterexamples. We argued that the axiom of recovery places an important rationality constraint on iterated revision, a framework that requires that we think of revision as taking place on epistemic states which encode preferences rather than just flat belief sets. We believe the connection between the axiom of recovery and the (C2) postulate of Darwiche-Pearl to be an interesting one. For future work it might be interesting to try and obtain a weakened version of the (C1) postulate in a way that is similar to what we have done in this paper.

References

[1] Carlos Alchourron, Peter Gärdernfors, and David Makinson, 'On the logic of theory change: Partial meet functions for contraction and revision', *J. of Symbolic Logic*, **50**, 510–530, (1985).

[2] Horacio Arlo-Costa and Rohit Parikh, 'Two place probabilities, beliefs and belief revision: on the foundations of iterative belief kinematics', in *Proceedings of the The Twelfth Amsterdam Colloquium*, (1999).

[3] John Cantwell, 'Some logics of iterated belief change', *Studia Logica*, **63**(1), 49–84, (1999).

[4] Adnan Darwiche and Judea Pearl, 'On the logic of iterated belief revision', *Artificial Intelligence*, **89**, 1–29, (1997).

[5] J. P. Delgrande and T. Schaub, 'A consistency-based model for belief change', in *Proceedings of the 2000 Non-Monotonic Reasoning Workshop (NMR'2000)*, Breckenridge, Colorado, (April 2000).

[6] Michael Freund and Daniel Lehmann, 'Belief revision and rational inference', Technical Report TR-ARP-94-16, The Leibniz Center for Research in Computer Science, Institute of Computer Science, Hebrew University, July 1994, (1994).

[7] André Fuhrmann, 'On the modal logic of theory change', in *The Logic of Theory Change: Workshop, Konstanz, FRG, October 1989, Proceedings*, eds., A. Fuhrmann and M. Morreau, volume 465 of *Lecture Notes in Artificial Intelligence*, pp. 259–281, Berlin, (1991). Springer-Verlag.

[8] Sven Ove Hansson, 'Belief contraction without recovery', *Studia Logica*, **50**, 251–260, (1991).

[9] Sven-Ove Hansson, 'Changes of disjunctively closed bases', *Journal of Logic, Language and Information*, **2**(4), 255–284, (1993).

[10] Sven Ove Hansson, *A Textbook of Belief Dynamics, Theory Change and Database Updating (to appear)*, Kluwer Academic Publishers, Dordrecht, 1999.

[11] Daniel Lehmann, 'Belief revision, revised', in *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pp. 1534–1540, (1995).

[12] Isaac Levi, *The Fixation of Belief and Its Undoing*, Cambridge University Press, Cambridge, 1991.

[13] David Makinson, 'On the status of the postulate of recovery in the logic of theory change', *Journal of Philosophical Logic*, **16**, 383–394, (1987).

[14] Thomas Meyer, 'Basic Infobase Change', in *Advanced Topics in Artificial Intelligence*, ed., Norman Foo, volume 1747 of *Lecture Notes In Artificial Intelligence*, pp. 156–167, Berlin, (1999). Springer-Verlag.

[15] A. C. Nayak, N.Y. Foo, M. Pagnucco, and A. Sattar, 'Changing conditional beliefs unconditionally', in *Proceedings of the Sixth Conference on Theoretical Aspects of Rationality and Knowledge*, pp. 119–135, De Zeeuwse Stromen, The Netherlands, (1996). Morgan Kaufmann.

[16] Abhaya Nayak, *Studies in Belief Change*, Ph.D. dissertation, University of Rochester, 1993.

[17] Wolfgang Spohn, 'Ordinal conditional functions: A dynamic theory of epistemic states', in *Causation in Decision: Belief, Change and Statistics: Proceedings of the Irvine Conference on Probability and Causation: Volume II*, eds., William L. Harper and Brian Skyrms, volume 42 of *The University of Western Ontario Series in Philosophy of Science*, pp. 105–134, Dordrecht, (1988). Kluwer Academic Publishers.

Explanatory Diagnosing and Meaningful Perception

Alberto Finzi and Fiora Pirri
Dipartimento di Informatica e Sistemistica
Università degli Studi di Roma "La Sapienza"
Via Salaria 113, 00198 Roma, Italy
finzi,pirri@dis.uniroma1.it

Abstract

In this paper we focus on the problem of sensing data interpretations in the context of a dynamic system accounting for both control and sensing actions failures. The execution of a sequence of actions mentioning sensing actions can lead to a misalignment between the observed status and the expected one. We introduce a model for failures and explanatory diagnosis to produce a plausible revision of the actions' history, one in which all actions leading to a misalignments are avoided. The revised history serves also as an explanation of the plausible/actual state of the agent. In connection with this revision we get an interpretation of what has been perceived.

1 INTRODUCTION

Consider the following simple scenario: an agent has to pass through a glass door using a remote control that changes the state of the door (from closed to open and vice versa), and if there are no obstacles it has to reach a place behind the door. To use the remote control the agent needs to step in front of the door. Let us consider only the high level actions, either sensing or control actions, i.e. a possible sequence (defined in an extremely simplified language in the style of the Situation Calculus) as follows, in which the numeric value is the outcome of the sensing action:

$$[sense(isDoor, 1); sense(isOpen, 0); open; sense(isFree, 0)]$$

how many failures can take place? The first sensing can report an error, because the agent senses that there is a door where there is not, the second sensing action could be wrong because the agent senses that the door is not open just because what it senses is not a door, or because the sensors are malfunctioning. The third action is a control action that could fail because the remote control changes the state of the door, and if the door was open it will become closed, or because the remote control is malfunctioning. Finally the last action could fail either because an object previously not in the map just appeared there or because the sonars are malfunctioning, and so on and so forth.

When failures are admitted, then the qualification problem becomes even worst than in any classical representation problem, and a very simple sequence of actions can explode into a tremendous set of nondeterministic actions. Moreover, after a failure, an action, previously considered executable might become non executable, that is, the preconditions to execute such an action might no longer be verified. Another problem is the following, when one or more sensing actions fail, the agent might become unable to state the real effects of its actions and, therefore, it might become unable to distinguish between what it entails (from its prior knowledge and the acquired one) and what the real world is like. In general, human beings rely on their senses, despite misperception is admitted. Obviously human beings are able to distinguish between a misperception and a malfunctioning of a sense. A misperception, in particular, is an random error while a malfunctioning can be procrastinated until it is repaired. In this paper we focus on the problem of sensing data interpretation in the context of a dynamic system admitting failures of both control and sensing actions. Given a Basic Theory of Actions (BTA), defined in the Situation Calculus (Reiter, 2001), we combine a model for perception, presented in (Pirri & Finzi, 1999), with a formal account for action failures and diagnosis, provided in (Finzi & Pirri, 2001b, 2001a), together with a model of failures for sensing actions. In order to preliminary provide a model for failures in which nothing is certain, nor the effects of actions nor

perception, we state some formal constraints that can help the agent to make an explanation and find a diagnosis that can be used to realign the agent behaviour with its goals. The success or failure of primitive actions is represented by means of stochastic actions: events are sequences of stochastic actions. The execution of a sequence of actions can induce the execution of several events, each of which having a given probability. In this way we provide an execution model for both control and sensing actions. The execution model, captures, somewhat, the agent's empirical beliefs on the effects of actions. A diagnosis/explanation is a history, i.e. a sequence of actions, that explains the observation (e.g. see the possible causes approach (Li & Pereira, 1996; Liberatore, 1999; Boutilier, 1995; McIlraith, 1999)). We call this process *explanatory diagnosis* (analogously to (McIlraith, 1999; Shananan, 1993), despite the approach is rather different and a comparison is discussed in Section 5). Consider the execution of a run mentioning both control and sensing actions. After its execution the outcome might be misaligned with respect to the entailed properties. If it is not possible to rely on perception, because also failures of sensing actions can take place, then there are the following possibilities:

1. The outcome of sensing is consistent with the entailed truth, then:
   (a) Despite the apparent consistency, some action (either sensing or control) failed.
   (b) Everything is correct.
2. The database entails some mistake or some misalignment between the inner beliefs (gathered from perception), and the a priori knowledge.

Now the only way to intervene is when a mistake is detected. In this case we will build a diagram that: 1. takes into account the plausibility of any sentence representing a possible state of the world *close enough* to the believed and mistaken state; 2. takes into account the plausibility of all the possible histories represented in the model of failure; 3. chooses an explanation which satisfies some minimality and optimality criterion and, at the same time, it explains a plausible state. A discussion of our approach in relation with other works will be provided in the discussion section of this paper.

2 PRELIMINARIES

We consider the Situation Calculus (SC) as the core of our logic and language, and we suitably extend it to include new sorts, new symbols for its alphabet and new axioms. See (Pirri & Reiter, 1999) for a full presentation of the core logic and language. To introduce a more expressive language able to deal with perception and failures we have extended the core logic and language in (Pirri & Finzi, 1999; Finzi & Pirri, 2001a). The logical structure of SC is a many sorted first order logic with equality defined by a signature $\mathcal{S}_{SC} = \langle Sort, Fun, Pred, Var\rangle$, a language $\mathcal{L}_{SC}$ (the language of Situation Calculus) and an interpretation $\mathcal{I} = (D, _^{Fun}, _^{Pred})$. *Sort* is a set, each element of which is a sort. The core *Sort* consists of three sorts: *action, situation* and *object* = *worldEntities* ∪ *genericObjects*. Where the sort *genericObject* is a catch-all that can include any possible additional sort needed in an application, like the reals or sets. In this paper we shall make use also of the following sorts, already introduced in (Pirri & Finzi, 1999; Finzi & Pirri, 2001a): *perceptible, outcome, stochasticAction, event.* We recall that the sort event consists of two sub-sorts, *event* = *condEvent* ∪ *seqEvent*. *Fun* is a family of sets including functions of sort genericObject like $\{+, -, /, length, ..\}$, and the function **P**, denoting the probability on events, taking as argument an event and returning a real, functions of sort action e.g. $\{grasp, goTo, fall, slide, ..\}$, functions of sort perceptibles e.g. $\{isOn, isGreen, ..\}$, and so forth: each element in a set of *Fun* is a function symbol whose sort is in *Sort* and having as arguments terms which, in turn, belong to *Sort*. *Pred* is a family of sets including, for example, relations taking as arguments terms of sort object and terms of sort situation, which are called relational fluents (i.e. dynamic relations, because their truth values depend on their situation argument) like $\{Clear, On, OnTable, ..\}$; the relation $\{Percept\}$, which takes as argument perceptibles and terms of sort situation; non-dynamic relations like $\{Table, Block, Set, Number, ..\}$. Each element in a set of *Pred* is a predicate symbol having as arguments terms, which belong to *Sort*. *Var* is a family of sets. Finally D is a sorted domain. Observe that D could be infinite or uncountable even if some of its subset is finite. For example we would be interested to impose a finite domain for world entities, while we would require probabilities to range over the reals. We shall use the following alphabet, with subscripts and superscripts, to denote elements of *Var*: a, for variables of sort action; s, for variables of sort situation; sta, for variables of sort stochastic action; e, for variables of sort event; w for variables of the sub-sort *seqEvent* and u for variables of the sub-sort *condEvent*; p, for variables of sort perceptible; pr, for variables of sort outcome and $x, y, z...$, for variables of sort object. A *basic sentence* is one formed

only by fluents, e.g. $\exists x \forall y(P(x, S_0) \rightarrow Q(x, y, S_0))$, in which fluents take as argument only elements of sort *worldEntities* (*we*). A *sentence uniform in* σ is one in which no variable, free or quantified, of sort situation is mentioned.

Often, in the SC, we introduce new functions and new relations, also new fluents, by definition, without extending the language, therefore the new predicates and functions are treated as macros. Sometimes we extend the notion of macros to special new terms which are neither functions nor predicates, an example is the term *Prob* which takes as argument a sentence and returns its probability. The term *Prob* is defined in (Finzi & Pirri, 2001a). For a discussion on the use of macros see (Reiter, 2001).

In the following subsections we shall give an overview of a *Basic Theory of Actions* and its expansions with probabilities, non deterministic actions and perception. All these expansions have been thoroughly presented in (Finzi & Pirri, 2001a; Pirri & Finzi, 1999), we refer the reader to these papers for a more detailed presentation.

2.1 THE BASIC THEORY OF ACTIONS

A *Basic Theory of Actions* (BTA) $\mathcal{D}$ is composed of the following sets of axioms in $\mathcal{L}_{SC}$:

$$\Sigma \cup \mathcal{D}_{una} \cup \mathcal{D}_{S_0} \cup \mathcal{D}_{ap} \cup \mathcal{D}_{ss} \tag{1}$$

Here Σ is the set of domain independent axioms specifying the properties of the domain of sort situation and $\mathcal{D}_{una}$ is the set of unique name for actions. $\mathcal{D}_{S_0}$ represents the initial database: the set of formulae that mention only the situation term S_0 and no other term of sort situation. $\mathcal{D}_{ap}$ is the set of action precondition axioms, one for each term of sort action:

$$Poss(A(\vec{x}), s) \equiv \Pi_A(\vec{x}, s) \tag{2}$$

For example:

$$\begin{aligned} &Poss(grasp(x), s) \equiv \\ &\exists x_0\ y_0\ x_1\ y_1 . Position(x, x_0, y_0, s) \wedge \\ &\quad Arm(x_1, y_1, s) \wedge dist(x_0 - x_1, y_0 - y_1) < 10. \end{aligned}$$

$\mathcal{D}_{ss}$ is the set of successor state axioms, one for each fluent:

$$F(\vec{y}, do(A(\vec{x}), s)) \equiv \Phi_F(\vec{y}, A(\vec{x}), s) \tag{3}$$

For example:

$$\begin{aligned} Position(o, x, y, do(a, s)) \equiv \\ a = goTo(x, y) \wedge Holding(o, s) \vee \\ (\neg Holding(o, s) \vee a \neq goTo(x, y)) \end{aligned}$$

Here $\Pi_A(\vec{x}, s)$ and $\Phi_A(\vec{y}, A(\vec{x}), s)$ are formulae whose free variables are only those among $\vec{x}$ and s, and they do not mention other situations, nor *do*, *Poss* nor $\sqsubset$.

The preeminent proof method for a BTA is regression (Pirri & Reiter, 1999). Given a sentence $\psi(\sigma)$ there exists a sentence $\mathcal{R}(\psi(\sigma))$ uniform in S_0, s.t.:

$$\mathcal{D} \models \forall(\psi(\sigma) \equiv \mathcal{R}(\psi(\sigma)))$$

In other words, given a sentence uniform in a situation σ regression amounts to the transformation of such a sentence into an equivalent sentence mentioning only S_0.

2.2 EXTENDING THE BTA WITH LOGICAL PROBABILITIES

If the initial database $\mathcal{D}_{S_0}$ is incomplete (see (Reiter, 2001; Finzi, Pirri, & Reiter, 2000)), we can assign a probability to those statements ψ, s.t. $\mathcal{D}_{S_0} \not\models \psi$. So, for example, if a goal G is given, we might be interested in finding a sequence σ of actions, such that $Prob(G(\sigma)) \geq 0.7$ is entailed by the theory. The notion of probability that we are concerned with, comes from the tradition of the *logical interpretations* of probability (Roeper & Leblanc, 1999). The basic idea consists in treating probability on statements in terms of logical proximity or degree of deducibility. These ideas have been presented in (Finzi & Pirri, 2001b, 2001a). As we mentioned above, the sort *object* is partitioned into two sets *worldEntities* and *genericObjects*. As the sort *worldEntities* (*we*) represents concrete entities of the domain (e.g. blocks in a blocks world), we impose a fixed and finite cardinality; on the other hand the sort *genericObjects* might have an infinite cardinality. This assumption is formalized via a set of axioms fixing the cardinality of the sort *we* in each model of the theory. These axioms are also used to describe a possible state, so for example if we want the domain to have only two elements of sort *we* and the language mentions two fluents $P(x, s)$ and $Q(x, s)$ then there would be $2^{m^n} = 16$, with n the cardinality of the domain, state descriptors. One of which would be as follows:

$$\begin{aligned} &\forall x \exists y_1 y_2 (y_1 \neq y_2 \wedge x = y_1 \vee x = y_2) \wedge \\ &P(y_1, S_0) \wedge \neg P(y_2, S_0) \wedge Q(y_1, S_0) \wedge Q(y_2, S_0) \end{aligned}$$

A state descriptor is a possible interpretation, on the domain *we* of the relations of the language (including equality), in the style of Carnap (Carnap, 1950). In (Finzi & Pirri, 2001a) we have shown how to quotient the structures of $\mathcal{D}_{S_0}$ into equivalence classes, and that each state descriptor identifies exactly a unique class of structures. State descriptors are thus used to count

these equivalence classes, on the basis of which we assign a probability to sentences, in particular *basic sentences* which are regressable sentences (i.e. uniform in a situation σ) mentioning only terms of sort *we*. For other sentences, in general, we are only able to approximate a probability value by an interval. In this paper we consider only probability on basic sentences.

We briefly summarize our approach to logical probabilities as follows. Let $P_i^\star$ be a state descriptor mentioning the (finite) set of all fluents, functions and relations of $\mathcal{L}_{SC}$ and asserting that the domain of sort *we* has cardinality n. Let $\mathcal{D}_{S_0}$ be equivalent, for some ordering on the $P^\star$s, to $\bigvee_{1\le i\le j} P_i^\star \wedge \bigwedge_{j<k\le n}^{i\neq k} \neg P_k^\star$, let $r \in \mathbb{R}$, we define:

$$Prob(P_i^\star) = \frac{1}{r} \text{ iff } r = \sharp\mathcal{M} \wedge P_1^\star \vee \ldots \vee P_j^\star \wedge \bigwedge_{j<k\le n}^{i\neq k} \neg P_k^\star$$

$$Prob(P_i^\star) = 0 \text{ iff } \neg P_i^\star$$

where $\sharp\mathcal{M}$ is a parameter defined so that it stands for the number of state descriptors compatible with $\mathcal{D}_{S_0}$ (i.e. $\sharp\mathcal{M} = \sharp\{P_i^\star | P_i^\star \models \mathcal{D}_{S_0}\}$).

The definitions of *Prob* over the state descriptors are indeed probabilities that can be assigned to basic sentences of $\mathcal{L}_{SC}$, uniform in S_0, as follows:

$$\begin{array}{c} Prob(\varphi) = \sum_{1\le i\le j} Prob(P_i^\star) \\ \text{iff} \\ \varphi \equiv \bigvee_{1\le i\le j} P_i^\star \wedge \bigwedge_{j<k\le n} \neg P_k^\star \end{array} \quad (4)$$

2.3 EXECUTION MODEL AND EMPIRICAL BELIEFS

Despite probabilities allow one to model an incomplete domain we still need to account for the nondeterminism of actions execution, in other words we have to face two sources of uncertainty: the incompleteness of the initial information, that we managed with logical probability, and the uncertainty due to the possible failure of actions execution. To represent nondeterministic actions in (Finzi & Pirri, 2001a) we have extended the BTA to a Basic Theory of Actions and Events (BTAE). Given a primitive action a (i.e. a primitive deterministic action as it is intended in a BTA), we consider a corresponding *stochastic action* $sta = \langle v(a), a\rangle$, where $v(a)$ is a random variable standing for either success ($v(a) = 1$) or failure ($v(a) = 0$), e.g. $\langle 0, moveOn(x,y)\rangle$ means that the execution of $moveOn(x,y)$ failed (see figure 1). An action failure could induce a *finite* chain of exogenous effects. Hence we model an *action failure* as: *missed execution + possible exogenous actions*. Since any exogenous action can take place as a consequence of a failure we make the hypothesis that all the possible exogenous actions are known, i.e. we make a *closure failure assumption*. In the case of a failure ($\langle 0, a\rangle$), a possible chain of stochastic exogenous actions can take place (e.g. $\langle 1, slideTo(posB)\rangle \circ \langle 0, moveTo(posA)\rangle \circ E_0$ means that the failure of moving to position $posA$ induced the exogenous event of sliding to position $posB$, with success). In (Finzi & Pirri, 2001a) we have formalized an event e as either a sequence event w of sort *seqEvent* or as a conditional event u of sort *condEvent*.

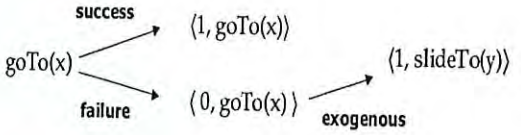


Figure 1: Stochastic actions

A *sequence event* ($\langle v(a), a\rangle \circ w$) corresponds to the event obtained as a sequence of stochastic actions (e.g. $\langle 0, goTo(a)\rangle \circ \langle 1, grasp(a)\rangle \circ E_0$).

A *conditional event* ($\langle v(a), a\rangle \mid w$) is the event obtained as a result of executing a stochastic action $\langle v(a), a\rangle$ conditioned on the sequence event w.

Nondeterminism of action is obtained by expanding a sequence of primitive actions into an event tree so, for example, $do(grasp(a), do(goTo(a), S_0))$ is expanded into the event tree depicted in Figure 2.

Events e are given a probability $\mathbf{P}(e)$. We have been introducing in (Finzi & Pirri, 2001a) a functional fluent $\mathbf{P}$: *event* $\mapsto$ *object*, in the event space, representing the probability for an event to happen while executing a sequence of actions σ, e.g. $\mathbf{P}(\langle 0, grasp(block1)\rangle | E_0) = 0.9$. Observe that events and situations are different terms but strictly related: a sequence event represents a *possible* execution of a situation. On the other hand, given an event, there exists a situation that can be obtained from it by considering the sequence of actions that succeeded. This conversion is provided by the the following function *trans*:

$$\begin{array}{l} trans(E_0) = S_0 \\ trans(\langle 1, a\rangle \circ e) = do(a, trans(e)) \\ trans(\langle 0, a\rangle \circ e) = trans(e) \end{array}$$

2.4 A BASIC THEORY OF ACTIONS AND PERCEPTION

In (Pirri & Finzi, 1999) we have been formalizing a new notion of perceptive actions in the context of the

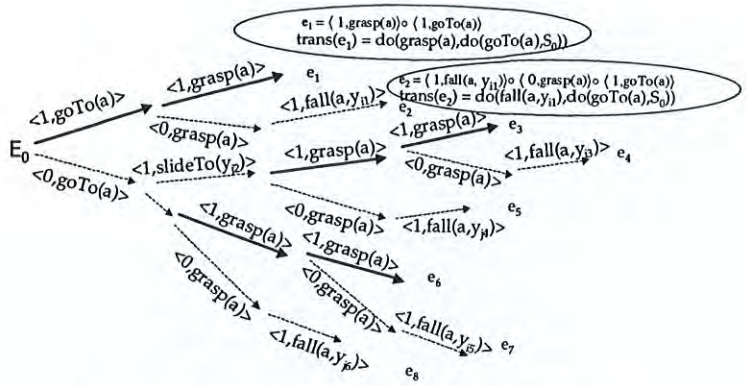


Figure 2: The tree $tree(do(grasp(a), do(goTo(a), S_0)))$

Situation Calculus. We have extended the language $\mathcal{L}_{SC}$ by introducing:

- *Perceptibles*: a perceptible is a term of the form $isF(\vec{x})$ representing a property, namely the observable fluent $F(\vec{x}, s)$, that can be inspected by sensors (e.g. $isColor(o, x)$ is the perceptible corresponding to the observable fluent $Color(o, x, s)$).

- *Sensing actions*: each perceptible can be a parameter of a sensing action: $sense(isF, pr)$ means that isF is a perceptible and $pr \in \{0, 1\}$ is the outcome (e.g. $sense(isColor(obj, red), 1)$).

- *Perception:* $Percept$ is the fluent representing the sensing actions' effects taking place only on the inner state of the agent. $Percept(isF, pr, s)$ means that the agent perceived isF with outcome pr in the situation s (e.g. $Percept(isColor(red), 1, do(sense(isColor(obj, red), 1), S_0))$).

Other special predicated have also been introduced, like: $Scene$ exploited for complex perceptibles description, and $Occluded$ used to reason about observability and so on, see also (Pirri & Romano, 2002) for a more deep analysis on the inner structure of a scene and its observability. In our approach, we distinguish between modeling sensing actions' direct effects (i.e. *direct perception*), and modeling the process of sensing data interpretation and assimilation (*selective perception*, and *meaningful perception*).

Direct Perception. The earlier perception is formalized by extending a BTA with the set of axioms $\mathcal{D}_{da}$ for describing the structure of the objects in the scene, and the set $\mathcal{D}_{gd}$ describing the class of primitive components (see for example Biedermann's *geons* (Biederman, 1987)). The set of axioms $\mathcal{D}_{una}$ is thus extended to include, together with the set of unique name for actions, unique name axioms for perceptibles. The set $\mathcal{D}_{ss}$ includes, together with the successor state axioms for each fluent in the language, also a successor state axiom for each perceptible, as an argument of the fluent $Percept$. According to the problem solving approach to perception (Rock, 1996) we define causal laws for perception, and by following the same ideas of (Reiter, 1991, 1993) we formulate them in terms of successor state axioms: for each perceptibles p

$$Percept(p, pr, do(A(\vec{x}), s)) \equiv \Phi_p(p, pr, A(\vec{x}), s) \quad (5)$$

here p is a perceptible, e.g. $isF(\vec{y})$, and pr is the outcome, $pr \in \{0, 1\}$. Analogously, we introduce successor state axioms for the fluents $Occluded$ and $Scene$, for each perceptible in the language. Finally $\mathcal{D}_{ap}$ includes action precondition axioms for sensing actions. For all perceptibles p:

$$Poss(sense(p, pr), s) \equiv \Pi_{sense(p,pr)}(x, a, s) \quad (6)$$

For example:

$$Poss(sense(isCube(x), pr), s) \equiv$$
$$Scene(isCube(x), s) \wedge \neg Occluded(isCube(x), s).$$

2.5 HIGH LEVEL PERCEPTION

The direct perception of a (dynamic) property of the world, i.e. a fluent $F(\vec{x}, s)$, is accounted by a sensing action $sense(isF(\vec{x}), 1)$ and by the fluent $Percept(isF(\vec{x}), 1, s)$. Sensing actions, indeed, do not affect the truth of *observable* fluents (which denote real world dynamic properties) they affect the inner

state of an agent through the fluent *Percept*. It might happen, therefore, that after an erroneous sensing action the agent comes to perceive a property, $Percept(isF(t), 1, \sigma)$ which does not hold in the real world at the specific situation σ. Here we have to distinguish two cases: $\mathcal{D} \not\models F(t, \sigma)$ and $\mathcal{D} \models \neg F(t, \sigma)$. In the first case, sensing does not contradict the information the agent has about the world, in the second it does, although $\mathcal{D} \models \neg F(t, \sigma) \wedge Percept(isF(t), 1, \sigma)$ does not imply a *logical contradiction*, as we have shown in (Pirri & Finzi, 1999). Now, if an agent wants to use perception to make hypotheses about the real world then we need to take care and trace the possible misalignment between perception (i.e. the inner state of the agent) and the real world information (i.e. what is described in terms of observable fluents and causal laws of transformation through the successor state axioms). For example suppose

$$\mathcal{D} \models Percept(isGreen(a), 1, do(sense(isGreen(a), 1, S_0)))$$

and $\mathcal{D} \not\models Green(a, do(sense(isGreen(a), 1, S_0)))$, and we want to use the perception to infer that a is green, then what we do is to add an hypothesis h to the initial database so that

$$\mathcal{D} \cup \{h\} \models Green(a, do(sense(isGreen(a), 1, S_0)))$$

The hypothesis is built using the *sensing history* $\sigma(\mathbb{H})$, that is the history of all the sensing actions that lead to the derivation of a given $Percept(p, pr, \sigma)$ or a given sentence mentioning the fluent *Percept*.

In fact, by the consistency theorem of the Situation Calculus (see (Pirri & Reiter, 1999)) a basic theory of actions is consistent given that the initial database $\mathcal{D}_{S_0}$ is consistent, therefore we want to add an hypothesis to the initial database. Since we want to gather the hypothesis from the *perceptual process* we proceed as follows. Suppose $\mathcal{D} \models \varphi(\sigma)$ and $\varphi(\sigma)$ is a sentence mentioning *Percept* and $\mathcal{D} \not\models \psi(\sigma)$, with $\psi(\sigma)$ a sentence in which $Percept(isF(\vec{x}), 1, \sigma)$ is transformed into $F(\vec{x}, \sigma)$ and $Percept(isF(\vec{x}), 0, \sigma)$ is transformed into $\neg F(\vec{x}, \sigma)$ (for details on the transformation see (Pirri & Finzi, 1999)), then we want to add an hypothesis h such that $\mathcal{D} \cup \{h\} \models \psi(\sigma)$. We construct such an hypothesis by regressing $\varphi(\sigma)$, in so extracting the sensing history. Then we transform the sensing history into observable fluents and we build a sentence uniform in S_0 using them. This explanation based (or abductive) process is called *meaningful perception*, because it allows us to use the perception process to infer real properties of the world. In fact, human being do this all the times, if one sees that the door is open, one usually uses the sensing to infer that indeed the door is open, obviously the outcome of sensing is not taken as *universal truth* in fact if the door was a glass door and one ends up bumping on it then one wants to give up the hypothesis.

Therefore we consider meaningful perception as a non monotonic inference process. Of course meaningful perception is not always possible. In (Pirri & Finzi, 1999) we have have shown that meaningful perception is always possible when there is no *misalignment* between perception and real world information. To account for the misalignment between the inner state of the agent which is transformed by perception (sensing actions and the fluent *Percept*) and the real world which is transformed by control actions we have introduced for each perceptible isF and property F the following definition[1]:

$$\begin{aligned} Mistaken(isF(\vec{x}), s) \stackrel{\text{def}}{\equiv} \\ (\neg F(\vec{x}, s) \wedge Percept(isF(\vec{x}), 1, s)) \vee \\ (F(\vec{x}, s) \wedge Percept(isF(\vec{x}), 0, s)). \end{aligned} \tag{7}$$

A misalignment admits two interpretations: (a) sensing is correct and the misalignment between perception and observable fluents is not a mistake; (b) the agent has a wrong perception of the real world. A brave approach would always consider sensing correct, admitting the outcome of sensing actions to changing the truth value of observable fluents (interpretation a). A cautious one would consider the truth value of the observable fluents a measure of the correctness of perception (interpretation b). The definition of *meaningful perception* given in (Pirri & Finzi, 1999) is based on this second approach.

3 WEAKENING MEANINGFUL PERCEPTION

Meaningful perception is a reasoning process requiring that there is no misalignment between what has been perceived and what can be derived in terms of observable properties of the real world. If a misalignment occurs then the following happens: $\mathcal{D} \models \varphi(\sigma)$ with $\varphi(\sigma)$ a sentence mentioning *Percept* and the transformation of $\varphi(\sigma)$ into an observable sentence $\psi(\sigma)$ is such that $\mathcal{D} \cup \{\psi(\sigma)\} \models \bot$. Now even if some misalignment takes place, we still might want to use the perceptual reasoning process consistently. In this section we discuss how to weaken meaningful perception in order to use the perceptual reasoning process as a form of hypothetical reasoning even in the presence of a misalignment.

[1] Observe that we assume that in the language for each observable fluent F there is a perceptible isF.

Definition 1 (Misperception) *Let $\varphi(\sigma)$ be a sentence, mentioning some perceptibles p_i, s.t. $\mathcal{D} \models \varphi(\sigma)$ and $\mathcal{D} \models \varphi(\sigma) \rightarrow \bigvee_i Mistaken(p_i, \sigma)$, then we call $\varphi(\sigma)$ a* misperception.

Suppose also that there exists a sentence $\varphi'(\sigma)$ such that

$$\mathcal{D} \models \varphi(\sigma) \rightarrow \varphi'(\sigma)$$

and for all perceptibles p_i mentioned in $\varphi'(\sigma)$,

$$\mathcal{D} \not\models \varphi'(\sigma) \rightarrow \bigvee_i Mistaken(p_i, \sigma)$$

then even if $\varphi'(\sigma)$ is a weaker sentence than $\varphi(\sigma)$ it does not induce misalignments, and therefore it can still be used for meaningful perception. For example let

$$\mathcal{D} \models Percept(isOnTable(a), 1, \sigma) \wedge Percept(isGreen(a), 1, \sigma)$$

and $\mathcal{D} \models \neg Green(a, \sigma)$, then there is a misalignment, still we would like to add the hypothesis that a is on the table. To this end we need to consider a subsequence σ' of σ excluding $sense(isGreen(a), 1)$ and keeping only the sensing actions that are essential to entail $OnTable(a, \sigma')$. In the following whenever we say that a sentence mentions perceptibles we intend that the perceptibles are argument of a fluent, e.g. *Percept*, not argument of a sensing action mentioned in σ.

Definition 2 (Weak Meaningful Perception) *Let $\varphi(\sigma)$ be a sentence mentioning perceptibles s.t $\mathcal{D} \models \varphi(\sigma)$ and for some perceptibles p_i, $\mathcal{D} \models \varphi(\sigma) \rightarrow \bigvee_i Mistaken(p_i, \sigma)$. The* weak meaningful perception *of $\varphi(\sigma)$ consists of an hypothesis h and a sentence $\varphi'(\sigma')$ such that $\mathcal{D} \models \varphi(\sigma) \rightarrow \varphi'(\sigma')$, for all perceptible p_i mentioned in $\varphi'(\sigma')$, $\mathcal{D} \not\models \varphi'(\sigma') \rightarrow \bigvee_i Mistaken(p_i, \sigma')$, and $\mathcal{D} \cup \{h\} \models \psi(\sigma')$, with $\psi(\sigma')$ the transformation of $\varphi'(\sigma')$ into a sentence not mentioning perceptibles.*

To account for weak meaningful perception we have to build both the hypothesis h and the sentence $\varphi'(\sigma)$ which is the weakened sentence for which we activate meaningful perception. To begin with, observe that we require that $\mathcal{D} \models \varphi(\sigma) \rightarrow \varphi'(\sigma')$ and for all perceptible p_i, $\mathcal{D} \not\models \varphi'(\sigma') \rightarrow \bigvee_i Mistaken(p_i, \sigma)$ therefore we have to give up some perceptibles which were misaligned or mistaken.

Proposition 1 *Let $\varphi(\sigma)$ be a sentence mentioning perceptibles s.t.*

$$\mathcal{D} \models \varphi(\sigma) \rightarrow \bigvee_i Mistaken(p_i, \sigma)$$

Let $\sigma(\mathbb{H})$ be the sensing history of $\varphi(\sigma)$ and let $\bigwedge tr(\sigma(\mathbb{H}))$ be the conjunction obtained through the transformation of each perceptible mentioned in the history into the corresponding fluent. Then

1. *$\mathcal{R}(\bigwedge tr(\sigma(\mathbb{H})))$ is inconsistent.*
2. *There exists a sequence $\sigma'(\mathbb{H})$, $\sigma'(\mathbb{H}) \sqsubseteq \sigma(\mathbb{H})$ s.t $\mathcal{R}(\bigwedge tr(\sigma'(\mathbb{H})))$ is consistent.*
3. *$\mathcal{D} \cup \{\mathcal{R}(\bigwedge tr(\sigma'(\mathbb{H})))\} \models \psi(\sigma')$, with $\psi(\sigma')$ a sentence not mentioning perceptibles and obtained by transformation of a sentence $\varphi'(\sigma')$, s.t.*
   (a) *For all perceptibles p_i mentioned in $\varphi'(\sigma')$, $\mathcal{D} \not\models \varphi'(\sigma') \rightarrow \bigvee_i Mistaken(p_i, \sigma')$.*
   (b) *$\varphi(\sigma) \rightarrow \varphi'(\sigma')$.*

Observe that the $\varphi'(\sigma')$ which are not misaligned and such that $\mathcal{D} \models \varphi(\sigma) \rightarrow \varphi'(\sigma')$ are implicates of $\varphi(\sigma)$ and there might be several of them. Let $\mathcal{I}(\varphi(\sigma))$, be the set of implicates of $\varphi(\sigma)$, such that

$$\mathcal{D} \not\models \gamma \rightarrow \bigvee_i Mistaken(p_i, \sigma)$$

for each $\gamma \in \mathcal{I}(\varphi(\sigma))$. We want to gather only those implicates of $\varphi(\sigma)$ which are minimal w.r.t. $\models$, in other words, we define the set of *plausible weakened meaningful perceptions* to be:

$$\mathcal{I}(min, \varphi(\sigma)) = \{\gamma' \in \mathcal{I}(\varphi(\sigma)) \mid \gamma \in \mathcal{I}(\varphi(\sigma)) \text{ and } \models \gamma \not\equiv \gamma', \text{ then } \not\models \gamma \rightarrow \gamma'\}$$

Observe that the set $\mathcal{I}(min, \varphi(\sigma))$ is the set of minimal implicates of $\varphi(\sigma)$.

Theorem 1 *Let $\gamma \in \mathcal{I}(\varphi(\sigma))$, with $\varphi(\sigma)$ a* misperception. *Let $\psi(\sigma)$ and θ be, respectively, the transformations of $\varphi(\sigma)$ and γ into sentences not mentioning perceptibles. Then:*

1. *$Prob(\psi(\sigma)) = 0$.*
2. *$Prob(\theta) > 0$*

According to the above theorem, we are interested in the probabilities of those sentences obtained by transforming the elements in $\mathcal{I}(\varphi(\sigma))$. Let

$$\mathcal{I}(tr, \varphi(\sigma)) = \{\theta \mid \gamma \in \mathcal{I}(\varphi(\sigma)) \text{ and } \theta = tr(\gamma)\}$$

Here $tr(\gamma)$ is the transformation of γ into a sentence not mentioning perceptibles, for details see (Pirri & Finzi, 1999). We shall choose among the set of not misaligned implicates $\mathcal{I}$ of $\varphi(\sigma)$ those more *reliable* according to the following preference criterion.

Definition 3 (Likelihood) *Let φ_1 and φ_2 be two sentences in $\mathcal{I}(tr, \varphi(\sigma))$, and let $\prec$ be a relation defined as follows:*

$$\varphi_1 \prec \varphi_2 \text{ iff } Prob(\varphi_1) > Prob(\varphi_2) \quad (8)$$

We say that φ_1 is more likely than φ_2 w.r.t. $\varphi(\sigma)$.

Observe that $\prec$ is a total ordering on the transformed implicates of $\varphi(\sigma)$ therefore we shall have a set of preferred sentences $\varphi(\sigma')$ satisfying weak meaningful perception. We call this set of preferred sentences a *reliable set of transformed implicates* of $\varphi(\sigma)$, and we denote them as $Rel(\mathcal{I}, \varphi(\sigma))$.

Now, given a $\psi \in Rel(\mathcal{I}, \varphi(\sigma))$, we want to find the hypothesis h s.t. $\mathcal{D} \cup \{h\} \models \psi$. The hypothesis h can be extracted from the sensing history $\sigma'(\mathbb{H})$ of ψ in the same way as it is done for meaningful perception. Observe that, in general, we can define a preference relation on hypotheses since we obviously prefer simpler than redundant hypotheses. A preference relation on hypotheses is a partial order, finally on the set of preferred hypotheses we can apply the same likelihood criterion, using probabilities that we applied for implicates.

4 SENSING DATA INTERPRETATION AS EXPLANATORY DIAGNOSIS

So far we accounted for *misperception*, i.e.

$$\mathcal{D} \models \varphi(p, \sigma) \rightarrow Mistaken(p, \sigma)$$

for some p, with $\varphi(\sigma)$ a sentence mentioning perceptibles, however we had no model for misperception, i.e. a model for sensing failures. In this section we introduce such a model, although we shall not distinguish between misperception as a pure logical mechanism and misperception due to failure, e.g. a broken sensor.

- Given a sensing action $a = sense(isP_i, p_i)$, we introduce the $\langle v_i, sense(isP_i, p_i)\rangle$ stochastic action. We call *control event* the event that does not contain stochastic sensing actions. Analogously, we call *sensing event* the event that contains just stochastic sensing actions.
- Given a stochastic sensing actions, we define a probability of success: $\mathbf{P}(\langle 1, a_s\rangle | e)$. In the hypothesis of accurate perception we assume $\mathbf{P}(\langle 0, a_s\rangle | e) = 0$: the probability of failure of a sensing action, given that it is executable, is 0. $\mathbf{P}(\langle 0, a_s\rangle | e) < 1$ captures noisy sensing.

Observe that in this framework, an action failure means a missed execution, hence $\langle 0, sense(isP_i, p_i)\rangle$ stands for missed sensing. Given an event mentioning the failure of a sensing action, *misperception* and *sensing action failure* coincides. Observe that, in the event space, we treat in a uniform way sensing action failures due to physical error (e.g. broken sensor) and interpretation failures (e.g. glass windows closed). Analogously to control actions, an exogenous action could take place as a consequence or cause of the failure of a sensing action. In general, in the context of sensing actions we prefer to interpret an exogenous action as an explanation of the sensing failure, e.g. in the case of a broken sensor, the exogenous action $break_sensor(x)$ can be an explanation of the relative misperception. In the case of an interpretation failure the relative exogenous action can be exploited in order to provide alternative interpretations. Consider the following example. The agent is in front of a room and it exploits its sensors in order to understand weather the door is open or closed. As there are glass doors in the domain, visual sensing of the perceptible $isOpenDoor$ could be misinterpreted. We can associate to the failure of the sensing action $\langle 0, sense(isOpenDoor, 1)\rangle$ the success of the exogenous sensing action $\langle 1, sense(isGlassDoor, 1)\rangle$. In this case an exogenous sensing action represents a correction of the interpretation and should be viewed as a plausible assumption.

Let $\varphi(\sigma)$ be a sentence mentioning perceptibles and s.t. $\mathcal{D} \models \varphi(\sigma)$ and

$$\mathcal{D} \models \varphi(p_i, \ldots, p_n, \sigma) \rightarrow \bigvee_{i \leq n} Mistaken(p_i, \sigma) \quad (9)$$

This sentence $\varphi(\sigma)$ is a *misperception*. As above (i.e. weak meaningful perception) we are looking for a sentence $\varphi'(\sigma')$ such that

$$\mathcal{D} \not\models \varphi'(p_i, \ldots, p_n, \sigma') \rightarrow \bigvee_{i \leq n} Mistaken(p_i, \sigma')$$

and $\mathcal{D} \models \varphi(\sigma) \rightarrow \varphi'(\sigma')$ and an hypothesis h that will explain the sentence $\psi(\sigma)$ obtained from $\varphi'(\sigma')$, by suitable transformation so as not to mention any perceptible. However we can now exploit the failure model, that is the event tree $tree(\sigma)$, generated by the situation σ, to find out both h and $\varphi'(\sigma')$.

Let $Rel(\mathcal{I}, \varphi(\sigma))$ be the set of transformed reliable implicates of $\varphi(\sigma)$, as we defined in Equation (8). For all $\psi(\sigma') \in \mathcal{I}(tr, \phi(\sigma))$ we know that if $\gamma \in Rel(\mathcal{I}, \varphi(\sigma))$ then $Prob(\gamma) \geq Prob(\psi(\sigma'))$. On this basis and considering the failure model given by the tree of σ we can now introduce the concept of hypothesis h that

explains the transformed reliable implicates of $\varphi(\sigma)$ and therefore explains what plausibly happened and where the agent is and what it has done.

Definition 4 (Hypotheses set) *Let $\varphi(\sigma)$ be a sentence mentioning perceptible and such that it is a misperception (see Equation 9). Let $tree(\sigma)$ be the failure model, that is, the event tree associated with all the possible failures of both control and sensing actions. Let $Rel(\mathcal{I}, \varphi(\sigma))$ be the set of transformed reliable implicates of $\varphi(\sigma)$. We define the set of hypotheses explaining a sentence $\varphi_i(\sigma') \in Rel(\mathcal{I}, \varphi(\sigma))$ as follows:*

$$\mathcal{H}(\sigma) = \{\sigma' | e \in tree(\sigma), \sigma' = trans(e) \text{ and } \sigma'(\mathbb{H}) \text{ is a safe history and } Prob(\mathcal{R}[\bigwedge tr(\sigma'(\mathbb{H}))]) > 0\}$$

Here $\bigwedge tr(\sigma'(\mathbb{H}))$ is the complete observed formula *relative to the sensing history $\sigma'(\mathbb{H})$.*

The connection between a sentence $\varphi_i(\sigma') \in Rel(\mathcal{I}, \varphi(\sigma))$ and the set of hypotheses $\mathcal{H}(\sigma)$ is given by the following lemma.

Lemma 1 *Let $\varphi_i(\sigma') \in Rel(\mathcal{I}, \varphi(\sigma))$ then $\sigma' \in \mathcal{H}(\sigma)$*

The above lemma tells us that the failures model captures through the tree all plausible histories, as defined with the minimality criterion. The *likelihood value* for σ' can be defined extending *Definition 3*:

Definition 5 *Given a sequence of actions σ the* likelihood value *for $\sigma' \in tree(\sigma)$ is:*

$$likelihood(\sigma') = Prob(\mathcal{R}[\bigwedge tr(\sigma'(\mathbb{H}))] \times \sum_{e_i \in Eset(\sigma', \sigma)} P(e_i)$$

here $\bigwedge tr(\sigma'(\mathbb{H}))$ denotes the complete observed formula *relative to a $\varphi'(\sigma') \in Rel(\mathcal{I}, \varphi(\sigma))$ and $Eset(\sigma', \sigma) = \{e | e \in tree(\sigma), \sigma' = trans(e)\}$*

In this setting a diagnosis is, as in (Finzi & Pirri, 2001a), the history $\sigma' \in \mathcal{H}(\sigma)$ which maximizes the likelihood value of the *complete observed formula.*

Definition 6 (Diagnosis) *Let $\varphi(\sigma)$ be a misperception and $\varphi'(\sigma') \in Rel(\mathcal{I}, \varphi(\sigma))$. Let $\bigwedge tr(\sigma'(\mathbb{H}))$ be the* complete observed formula *and $\sigma' \in \mathcal{H}(\sigma)$. σ' is a diagnosis for $\varphi'(\sigma')$ iff:*

$$likelihood(\sigma') = max\{likelihood(\sigma'') | \sigma'' \in \mathcal{H}(\sigma)\}$$

Example The agent is in front of a glass door which can be opened only from inside. We assume incomplete information about doors and obstacles in the initial situation, e.g. $\mathcal{D}_{S_0} \not\models Door(S_0)$, $\mathcal{D}_{S_0} \not\models \neg Door(S_0)$, $\mathcal{D}_{S_0} \not\models Free(S_0)$, $\mathcal{D}_{S_0} \not\models \neg Free(S_0)$ etc.. The agent has to pass through the door, therefore we assume he had executed the following sequence of actions σ:

$$[sense(isDoor, 1); sense(isOpen, 0); open; sense(isFree, 0)]$$

After this execution, a sentence $\varphi(\sigma)$ will be obtained. It turns out that:

$$\varphi(\sigma) \rightarrow Free(\sigma) \wedge Percept(isFree, 0, \sigma)$$

and

$$Free(\sigma) \wedge Percept(isFree, 0, \sigma) \rightarrow Mistaken(isFree, \sigma)$$

In order to interpret this misperception, we have to find $\varphi'(\sigma') \in \mathcal{I}(\varphi(\sigma))$ such that

$$\mathcal{D} \not\models \varphi'(\sigma') \rightarrow Mistaken(isFree, \sigma')$$

and such that the transformation $\psi(\sigma')$ of $\varphi'(\sigma')$ is minimal and in $Rel(\mathcal{I}, \varphi(\sigma))$. In order to obtain a coherent set of observations and the associated explanation we can exploit the event tree generated by σ (see figure 3). Consider the event

$$e' = \langle 1, sense(isFree, 0)\rangle \circ \langle 0, open\rangle \circ \langle 1, sense(isOpen, 0)\rangle \circ \langle 0, sense(isDoor, 1)\rangle \circ E_0$$

the relative sequence of actions is

$$\sigma' = trans(e') = do(sense(isFree, 0), do(sense(isOpen, 0), S_0))$$

and the regression of the associated *complete observed formula* $\mathcal{R}[\bigwedge tr(\sigma'(\mathbb{H}))]$ is equivalent to $\neg Open(\sigma') \wedge \neg Free(\sigma')$. Given our assumptions for $\mathcal{D}_{S_0}$, we have that $Prob(\bigwedge tr(\sigma'(\mathbb{H}))) > 0$, hence σ' belongs to $\mathcal{H}(\sigma)$. The sequence σ' corresponds to the following explanation: $sense(isDoor, 1)$ was a misperception, there is no door, hence the *open* action failed. Such an explanation is associated with a likelihood value, if such value is the maximum one in $\mathcal{H}(\sigma)$, then we consider σ' as the best explanation.

5 DISCUSSION

In this paper we have described a framework where the problem of incorporating and interpreting sensing information is considered in the context of a dynamic theory (Li & Pereira, 1996; Liberatore, 1999; Boutilier, 1995). The perception process that we define is based on a plausible *explanation* of *what happened* to a dynamic system given some observations,

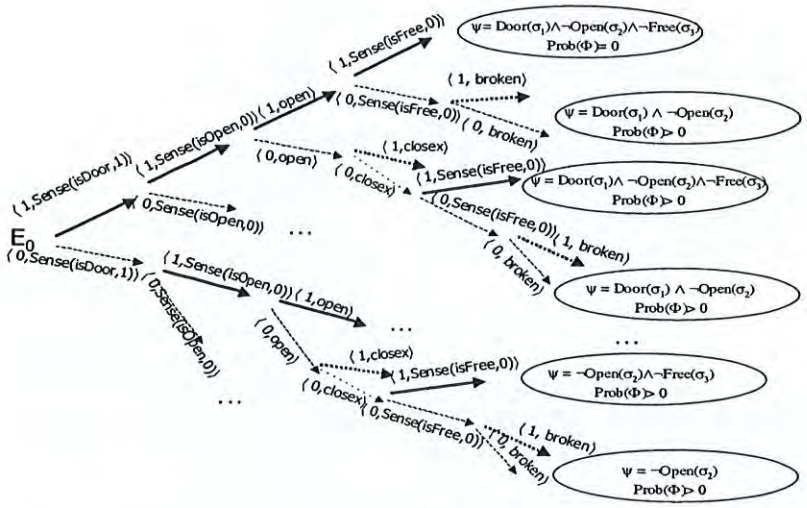


Figure 3: The tree $do(sense(isFree, 0), do(open, do(sense(isOpen, 0), do(sense(isDoor, 1), S_0))))$

under this point of view it can be related to *explanatory diagnosis* (McIlraith, 1999; Shananan, 1993). In (McIlraith, 1999) the *explanatory diagnosis* is thought as an *heavy diagnostic* process that extends the traditional one. In our framework the perspective is quite different. Our diagnostic process must be suitable for mobile robots tasks, therefore we are interested in a kind of *teleological diagnosis*: we are not concerned with discovering what is wrong in a complex system, rather we are interested in plausible explanations that we exploit to maintain the agent inner representation adherent to the external world as much as possible. It is important that the explanation resolves the uncertainties (as outcome of noisy sensing or failures) affecting the agent and limiting its behavior and its safeness. Analogously as in (Boutilier, 1995), we search for an explanation in the event space. Such an event space can be seen as a model of *what can go wrong*. Therefore, rather than exploiting the explanations to understand *what is wrong*, we exploit a model of *what can go wrong* to search for *what happened*.

Our formulation of the diagnostic process as a component of the perceptive process connects our model for sensing actions to the SDA model provided in (Shanahan, 1999). In fact, we share the perspective that the perceptive data should be filtered through a process taking place on the basis of a priori concepts (i.e. the initial information and the domain theory). In our system an explanation is related to an history extracted from our execution model, starting from an incomplete but truthful initial situation; in the SDA framework an explanation provides a new initial situation obtained exploiting abduction. The SDA definition of the diagnosing process requires a simpler representation: exogenous action are not introduced and no probabilistic model is exploited. Therefore the execution model is not distinguished from an idealized one, and the diagnostic process have to work with a quite detailed representation of the domain, as a matter of facts without a probabilistic representation the logical model must be very realistic. In this context the diagnostic process becomes quite hard, resembling the *what's wrong* diagnosis used for complex systems. In our approach we exploit an execution model modeling an explanation as sequence of actions, observe that this notion is different form the one in (Boutilier & Becher, 1995) and in (Williams, Pagnucco, Foo, & Sims, 1995).

Our process of revising sensing information can be considered in the light of an adjustment process (Benfehart, Kaci, Le Berre, & Williams, 2001; Williams, 1996; Dubois, Lang, & Prade, 1994)) in a dynamical settings, i.e. we have to minimally change the history in order to accommodate a maximal coherent set of observations. Our perspective is close to the one given by Boutilier in (Boutilier, 1995), where the agent, after observing φ, has to adjust its beliefs in order to find the most plausible φ-transition. In our framework, such transitions are represented by our model for failures (execution space); furthermore, as we allow for sensing action failure, the modification of the history σ' determines also a deviation of the sensing history $\sigma'(\mathbb{H})$. Observe that in (Boutilier, 1995) the failure of observations is not considered. In our framework, the diagnosis is exploited to gather both plausible ac-

tion executions and a plausible subset of observations. If we impose that our control actions do not fail (i.e. $\mathbf{P}(\langle 1, a\rangle|e) = 1$) and if we do not allow exogenous sensing actions, our selection of perceptible properties can be considered in the context of the *arbitration* problem (Liberatore & Schaerf, 1998; Revesz, 1993). However, observe that our iterated revision history process does not correspond to an iterated arbitration: in order to interpret new observations, the whole history has to be reinterpreted.

References

Benfehart, S., Kaci, S., Le Berre, D., & Williams, M.-A. (2001). Weakening conflicting information for iterated revision and knowledge integration. In *Proceedings of IJCAI'01.*

Biederman, I. (1987). Recognition by components. a theory of human image interpretation. *Psychological Review*, 115–147.

Boutilier, C. (1995). Generalized update: Belief change in dynamic settings. In *Proceedings of JCAI 1995*, pp. 1550–1556.

Boutilier, C., & Becher, V. (1995). Abduction as belief revision. *Artificial Intelligence.*

Carnap, R. (1950). *Logical Foundations of Probability.* University of Chicago Press, Chicago.

Dubois, D., Lang, J., & Prade, H. (1994). Possibilistic logic. In Gabbay, D., Hogger, C. J., & Robinson, J. A. (Eds.), *Handbook of Logic in Artificial Intelligence and Logic Programming, Volume 3: Nonmonotonic Reasoning and Uncertain Reasoning*, pp. 439–513. Oxford University Press, Oxford.

Finzi, A., Pirri, F., & Reiter, R. (2000). Open world planning in the situation calculus. In *Proceedings of AAAI-2000*, pp. 754–760.

Finzi, A., & Pirri, F. (2001a). Combining probability, failures and safety in robot control. In *IJCAI 2001*, pp. 1331–1336.

Finzi, A., & Pirri, F. (2001b). Diagnosing failures and predicting safe runs in robot control. In *Commonsense 2001*, pp. 105–112.

Li, R., & Pereira (1996). What is believed is what is explained. In *Proceedings of AAAI'96*, pp. 550–555.

Liberatore, P. (1999). Brels: a system for revising, updating, and merging knowledge bases. In *Workshop notes of the Third Workshop on Nonmonotonic Reasoning, Action, and Change (NRAC'99).*

Liberatore, P., & Schaerf, M. (1998). Arbitration (or how to merge knowledge bases). *IEEE Transactions on Knowledge and Data Engineering, 10*(1), 76–90.

McIlraith, S. (1999). Explanatory diagnosis: Conjecturing actions to explain observations. In *Logical Foundation for Cognitive Agents: Papers in Honour of Ray Reiter*, pp. 155–172. Springer-Verlag.

Pirri, F., & Finzi, A. (1999). An approach to perception in theory of actions: Part i. *ETAI, 4.*

Pirri, F., & Reiter, R. (1999). Some contributions to the metatheory of the situation calculus. *Journal of the ACM, 46*(3), 325–361.

Pirri, F., & Romano, M. (2002). A geon-bayes approach to object recognition.. Submitted for pubblication.

Reiter, R. (1991). The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression.. pp. 359–380. Academic Press.

Reiter, R. (1993). Proving properties of states in the situation calculus. Artificial Intelligence Journal, 337–351.

Reiter, R. (2001). *Knowledge in Action: logical foundations for describing and implementing dynamical systems. MIT press.*

Revesz, P. Z. (1993). On the semantics of theory change: Arbitration between old and new information.. pp. 71–82.

Rock, I. (1996). Indirect Perception. The MIT Press.

Roeper, P., & Leblanc, H. (1999). *Probability Theory and Probability Semantics.* University of Toronto Press.

Shanahan, M. (1999). What sort of computation mediates best between perception and action?. In Levesque, H. J., & (Eds.), F. P. (Eds.), *Logical Foundation for Cognitive Agents: Contributions in Honor of Ray Reiter*, pp. 352–369. Springer-Verlag.

Shananan, M. (1993). Explanation in situation calculus. In *In proceeding of IJCAI 1993*, pp. 160–165.

Williams, M.-A. (1996). A practical approach to belief revision: Reason based change. In *Proceedings of KR'96*, pp. 619–629.

Williams, M.-A., Pagnucco, M., Foo, N. Y., & Sims, B. (1995). Determining explanations using transmutations. In *IJCAI (1)*, pp. 822–829.

Two Representations for Iterative Non-prioritized Change

Alexander Bochman
Computer Science Department
Holon Academic Institute of Technology, Israel
email: bochmana@hait.ac.il

Abstract

We address a general representation problem for belief change, and describe two interrelated representations for iterative non-prioritized change: a logical representation in terms of persistent epistemic states, and a constructive representation in terms of flocks of bases.

1 Introduction

Two main kinds of representation for belief change have been suggested in the literature. In the AGM theory [Alchourrón *et al.*, 1985; Gärdenfors, 1988], belief change is considered as a change of *belief sets* that are taken to be deductively closed theories. According to the so-called base approach, however, a belief set should be seen as generated by some (finite) *base* (see, e.g., [Hansson, 1999]). Consequently, revisions of belief sets are determined on this approach by revisions of their underlying bases. This drastically reduces the set of alternatives and hence makes this approach constructive and computationally feasible.

The AGM theory and base approach constitute two seemingly incomparable representations for belief change, each with advantages of its own. Still, the framework of epistemic states suggested in [Bochman, 2001a; Bochman, 2001b] allows to formulate both as different species of a single general representation. The reformulation makes it possible, in particular, to reveal common constitutive principles behind them. These common principles embody, however, also common shortcomings that lead to a loss of information in iterated changes. This suggests epistemic states in their full generality as a justified representational alternative.

Epistemic states could be primarily seen as a generalization of the AGM models, namely, they form an abstract logical representation for belief change. Though the base-oriented models can be translated into epistemic states, the translation loses some important advantages of such models, namely their inherent finiteness and constructivity. Hence it would be desirable to have a constructive representation for epistemic states that would preserve these features of base models while avoiding their problems. Fortunately, such an alternative representation has already been suggested in [Fagin *et al.*, 1986], and it amounts to using sets, or *flocks*, of bases. As we will show, a certain modification of the original flock models will provide us with a constructive representation for an important class of epistemic states and belief change processes in them.

Remark. The theory described below, is different from that suggested in [Bochman, 1999; Bochman, 2000], since the latter has turned out to be flawed in its treatment of expansions. The shortcoming has created the need for shifting the level of representation from pure epistemic states used in the above papers to more general persistent epistemic states (see below).

2 Epistemic states

A common feature of the AGM and base representations is a preference structure on certain subsets of the belief set. In the case of the AGM paradigm, it is a preference structure on the maximal subtheories of the belief set, while for the base representation it is a preference structure on subtheories that are generated by the subsets of the base. The notion of an epistemic state, defined below, reflects and generalizes this common structure of the two representations.

Definition 2.1. An *epistemic state* $\mathbb{E}$ is a triple $(\mathcal{S}, l, \prec)$, where $\mathcal{S}$ is a set of *admissible belief states*, $\prec$ is a preference relation on $\mathcal{S}$, while l is a labelling function assigning a deductively closed theory (called

an admissible belief set) to every admissible belief state from $\mathcal{S}$.

Any epistemic state determines a unique set of propositions that are *believed* in it, namely, the set a propositions that hold in all maximally preferred admissible belief states. Thus, even if an epistemic state contains conflicting preferred belief states, we can still believe in propositions that hold in all of them. Still, the belief set associated with an epistemic state does not always constitute an admissible belief set by itself. The latter will hold, however, in an ideal situation when an epistemic state contains a unique most preferred admissible belief state. Such epistemic states will be called *determinate*. Both AGM and base models correspond to determinate epistemic states. Nevertheless, non-determinate epistemic states have turned to be essential for an adequate representation of both nonmonotonic reasoning and belief change. In particular, the necessity of accepting non-determination arises most clearly in the analysis of contractions (see below).

In what follows, we will concentrate on a special kind of persistent epistemic states that will provide a representation for a non-prioritized belief change.

Definition 2.2. An epistemic state will be called *persistent* if it satisfies

Persistence If $s \prec t$, then $l(s) \subseteq l(t)$.

For persistent epistemic states, the informational content of admissible belief states is always preserved (persist) in transitions to more preferred states. This property reflects the idea that the informational content of an admissible state is an essential (though presumably not the only) factor in determining the place of this state in the preference structure.

Persistent epistemic states are formally similar to a number of *information models* suggested in the logical literature, such as Kripke's semantics for intuitionistic logic, Veltman's data semantics, etc. (see [Benthem, 1988] for an overview). All such models consist of a partially ordered set of informational states. The relevant partial order represents possible ways of information growth, so it is assumed to satisfy the persistence requirement by its very meaning.

As will be shown, persistent epistemic states constitute a smallest natural class of epistemic states that is closed under belief change operations that do not involve prioritization. Moreover, it is precisely this class of epistemic states that admits a constructive representation in terms of flocks of bases.

An epistemic state will be called *pure* if it satisfies

Pure Monotonicity $s \preceq t$ if and only if $l(s) \subseteq l(t)$.

Pure epistemic states is a special kind of persistent states for which the preference relation is determined solely by the informational content of admissible belief states. Accordingly, a pure epistemic state can be defined simply as a set of deductively closed theories, with the intended understanding that the relation of set inclusion among such theories plays the role of a preference relation. In other words, for pure epistemic states, preference is given to maximal theories.

Pure epistemic states have been used in [Bochman, 1999; Bochman, 2000] as a basis for a foundational approach to belief change. As we mentioned, however, the approach has turned out to be flawed, since it does not provide an adequate representation of belief expansions. The present study stems from a more general approach to representing belief change suggested in [Bochman, 2001b].

3 Bases and flocks

Assume that a belief set $\mathbf{B}$ is generated by some base Δ with respect to a certain consequence relation Th (that is, $\mathbf{B} = \mathrm{Th}(\Delta)$). This structure is representable as an epistemic state of the form $(P(\Delta), l, \prec)$, where admissible belief states are the subsets of Δ, l assigns each such subset its deductive closure, while $\prec$ is a preference relation on $P(\Delta)$. In the simplest (non-prioritized) case, this preference relations is definable via set inclusion: $\Gamma \prec \Phi$ iff $\Gamma \subset \Phi$.

In the latter case base-generated epistemic states are equivalent to pure epistemic states consisting of theories of the form $\mathrm{Th}(\Gamma)$, where Γ ranges over the subsets of Δ. Notice that any such epistemic state will be determinate, since it contains a most preferred theory, namely $\mathrm{Th}(\Delta)$.

Unfortunately, we will see later that base-generated epistemic states (and bases themselves) are arguably inadequate for representing belief contractions. A generalization of bases that overcomes this shortcoming has been suggested in [Fagin *et al.*, 1986] and consists in using sets (or 'flocks') of bases.

By a *flock* we will mean an arbitrary set of sets of propositions $\mathcal{F} = \{\Delta_i\}$, for $i \in I$. Such a flock can be considered as a collection of bases Δ_i, and the following construction of the epistemic state generated by a flock can be seen as a natural generalization of base-generated epistemic states.

Any flock generates an epistemic state $\mathbb{E}_{\mathcal{F}} = \langle \mathcal{F}_{\downarrow}, l, \prec \rangle$ defined as follows:

- $\mathcal{F}_{\downarrow}$ is a set of all nonempty sets Γ such that $\Gamma \subseteq \Delta$, for some $\Delta \in \mathcal{F}$;
- $l(\Gamma) = \mathrm{Th}(\Gamma)$, for each $\Gamma \in \mathcal{F}_{\downarrow}$;
- $\Gamma \prec \Phi$ holds iff $\Gamma \subset \Phi$.

As can be seen, flocks constitute a generalization of bases. Namely, any base Δ can be identified with a singular flock $\{\Delta\}$.

As in our study, flocks were used in [Fagin *et al.*, 1986] as a framework for belief change operations. Our subsequent results will be different, however. The main difference can be described as follows.

Let us say that two flocks are *identical* if they generate the same epistemic state. Now, let $\mathcal{F}$ be a flock and Δ_0 a set of propositions such that $\Delta_0 \subseteq \Delta$, for some $\Delta \in \mathcal{F}$. Then it is easy to see that flocks $\mathcal{F}$ and $\mathcal{F} \cup \{\Delta_0\}$ produce the same epistemic state, and consequently they are identical in the above sense. This shows that a flock is determined, in effect, by its inclusion-maximal elements. According to [Fagin *et al.*, 1986], however, the above two flocks are distinct, and hence the validity of propositions with respect to a flock is determined, in effect, by *minimal* sets belonging to the flock. This makes the resulting theory less plausible and more complex than it could be.

The above feature, though plausible by itself, gives rise, however, to high sensitivity of flocks with respect to the syntactic form of the propositions occurring in them.

Example 3.1. Let us consider the flock $\mathcal{F} = \{\{A'\}, \{A, B\}\}$, where A' is logically equivalent to A. Replacing A' with A, we obtain a different flock $\mathcal{F}' = \{\{A\}, \{A, B\}\}$, which is identical to $\{\{A, B\}\}$. Note that $A \wedge B$ is believed in the epistemic state generated by the latter flock, though only A is believed in the epistemic state generated by the source flock $\mathcal{F}$.

The above example shows that flocks do not admit replacements of logically equivalent propositions, at least in cases when such a replacement leads to identification of propositions with other propositions appearing elsewhere in the flock. It should be kept in mind, however, that the epistemic state generated by a flock is a syntax-independent object, though purely syntactic differences in flocks may lead to significant differences in epistemic states generated by them.

Flock-generated epistemic states are always persistent; this follows immediately from the fact that $\Gamma \prec \Phi$ holds in such a state if and only if $\Gamma \subset \Phi$, and hence $\mathrm{Th}(\Gamma) \subseteq \mathrm{Th}(\Phi)$. Moreover, it has been shown in [Bochman, 2001b] that any finitary persistent epistemic state is representable by some flock. This means that flocks constitute an adequate syntactic formalism for representing persistent epistemic states. Unfortunately, this also shows that flocks are not representable by pure epistemic states, as has been suggested in [Bochman, 1999].

4 Changing epistemic states

Since belief sets are uniquely determined by epistemic states, operations on epistemic states will also determine corresponding operations on belief sets. Two kinds of operations immediately suggest themselves as most fundamental, namely removal and addition of information to epistemic states.

4.1 Contractions

Admissible belief states of an epistemic state constitute all potential alternatives that are considered as 'serious possibilities' by the agent. In accordance with this, the contraction of a proposition A from an epistemic state $\mathbb{E}$ is defined as an operation that removes all admissible belief states from $\mathbb{E}$ that support A. We will denote the resulting epistemic state by $\mathbb{E} - A$.

The contraction operation has quite regular properties, most important of which being *commutativity*: a sequence of contractions can be performed in any order yielding the same resulting epistemic state.

Let us compare the above contraction with base contraction. According to [Hansson, 1993], the first step in performing a base contraction of A consists in finding preferred (selected) subsets of the base that do not imply A. So far, this fits well with our construction, since the latter subsets exactly correspond to preferred admissible theories of the contracted epistemic state. Then our definition says, in effect, that the contracted belief set should be equal to the intersection of these preferred theories. Unfortunately, such a solution is unacceptable for the base paradigm, since we need to obtain a unique contracted base; only the latter will determine the resulting belief set. Accordingly, [Hansson, 1993] defines first the contracted base as the intersection of all preferred subsets of the base ('partial meet base contraction'), and then the contracted belief set is defined as the set of all propositions that are implied by the new base.

The problems arising in this approach are best illustrated by the following example (adapted from [Hansson, 1992]).

Example 4.1. Two equally good and reliable friends of a student say to her, respectively, that Niamey is

a Nigerian town, and that Niamey has a university. Our student should subsequently retract her acquired belief that Niamey is a university town in Nigeria.

Let A and B denote, respectively, propositions "Niamey is a town in Nigeria", and "There is a university in Niamey". Then the above situation can be described as a contraction of $A \wedge B$ from the (belief set generated by the) base $\{A, B\}$. As has been noted in [Gärdenfors and Rott, 1995], this small example constitutes a mayor stumbling block for the base approach to belief change. Actually, we will see that none of the current approaches handles satisfactorily this example.

To begin with, it seems reasonable to expect that the contracted belief set in the above situation should contain $A \vee B$, since each of the acceptable alternatives support this belief. This result is also naturally sanctioned by the AGM theory. Using the base contraction, however, we should retreat first to the two subbases $\{A\}$ and $\{B\}$ that do not imply $A \wedge B$, and then form their intersection which happens to be empty! In other words, we have lost all the information contained in the initial base, so all subsequent changes should start from a scratch.

Next, it seems also reasonable to require that if subsequent evidence rules out A, for example, we should believe that B. In other words, contracting first $A \wedge B$ and then A from the initial belief state should make us believe in B. This time the AGM theory cannot produce this result. The reason is that the first contraction gives the logical closure of $A \vee B$ as the contracted belief set, and hence the subsequent contraction of A will not have any effect on the corresponding belief state. Notice that this information loss is not 'seen' in one-step changes; it is revealed, however, in subsequent changes.

As we see it, the source of the above problem is that traditional approaches to belief change force us to choose in situations we have no grounds for choice. And our suggested solution here amounts to retaining all the preferred alternatives as parts of the new epistemic state, instead of transforming them into a single 'combined' solution. This means, in particular, that we should allow our epistemic states to be non-determinate. This will not prevent us from determining each time a unique current set of beliefs; but we should remember more than that.

Returning to the example, our contraction operation results in a new belief set $\mathrm{Th}(A \vee B)$, as well as a new epistemic state $\{\mathrm{Th}(A), \mathrm{Th}(B)\}$ consisting of two theories. This latter epistemic state, however, is not base-generated, though it will be generated by a flock $\{\{A\}, \{B\}\}$.

4.1.1 Contraction of flocks

Flocks emerge as a nearest counterpart of bases that will already be closed with respect to our contraction operation. Actually, the latter will correspond to the operation of deletion on flocks suggested in [Fagin *et al.*, 1986].

Definition 4.1. A *contraction of a flock* $\mathcal{F} = \{\Delta_i\}$ with respect to A (notation $\mathcal{F} - A$) is a flock consisting of all maximal subsets of each Δ_i that do not imply A.

The following result confirms that the above operation on flocks corresponds to contraction of associated epistemic states.

Lemma 4.1. *If $\mathbb{E}_{\mathcal{F}}$ is an epistemic state generated by a flock $\mathcal{F}$, then, for any A, $\mathbb{E}_{\mathcal{F}} - A = \mathbb{E}_{(\mathcal{F}-A)}$.*

Despite the similarity of the above definition of contraction with that given in [Fagin *et al.*, 1986], the resulting contraction operation will behave differently in our framework. The following example illustrates this.

Example 4.2. Contraction of the flock $\mathcal{F} = \{\{A\}, \{B\}\}$ with respect to A is a flock $\{\emptyset, \{B\}\}$, which is identical to $\{\{B\}\}$ according to our definition of identity. Consequently, B will be believed in the resulting epistemic state. This behavior seems also to agree with our intuitions, since eliminating A as an option will leave us with B as a single solution. In the representation of [Fagin *et al.*, 1986], however, $\{\emptyset, \{B\}\}$ is reducible to $\{\emptyset\}$. Consequently, nothing will be believed in the resulting epistemic state. Furthermore, this also makes the corresponding operation of deletion *non-commutative*: whereas deletion of $A \wedge B$ and then A from the base $\{A, B\}$ results in $\{\emptyset, \{B\}\}$, deletion of A first and then $A \wedge B$ will give a different flock, namely $\{\{B\}\}$.

4.2 Merge and expansion

The operation of expansion consists in adding information to epistemic states. In the AGM theory, this is achieved through a straightforward addition of the new proposition to the belief set, while in the base approach the new proposition is added directly to the base.

The framework of epistemic states drastically changes, however, the form and content of expansion operations. This stems already from the fact that adding a proposition to an epistemic state is no longer reducible to adding it to the belief set; it should deter-

mine also the place of the newly added proposition in the structure of the expanded epistemic state. This will establish the degree of firmness with which we should believe the new proposition, as well as its dependence and justification relations with other beliefs. As a way of modelling this additional information, we suggest to treat the latter as a special case of merging the source epistemic state with another epistemic state that will represent the added information. Accordingly, we will describe first some merge operation on epistemic states. Then an expansion will be defined roughly as a merge of $\mathbb{E}$ with a rudimentary epistemic state that is generated by the base $\{A\}$.

4.2.1 Merging epistemic states

Merge is a procedure of combining a number of epistemic states into a single epistemic state, in which we seek to combine information that is supported by the source epistemic states. It turns out that this notion of merging can be captured using a well-known algebraic construction of *product*. Roughly, a merge of two epistemic states $\mathbb{E}_1$ and $\mathbb{E}_2$ is an epistemic state in which the admissible states are all *pairs* of admissible states from $\mathbb{E}_1$ and $\mathbb{E}_2$, the labelling function assigns each such pair the deductive closure of the union of their corresponding labels, while the resulting preference relation agrees with the 'component' preferences (see [Bochman, 2001b] for a formal description).

Since the primary subject of this study is non-prioritized change, we will consider below only one kind of merge operations, namely a pure merge that treats the source epistemic states as having an equal 'weight'.

Definition 4.2. A *pure merge* of epistemic states $\mathbb{E}_1 = (\mathcal{S}_1, l_1, \prec_1)$ and $\mathbb{E}_2 = (\mathcal{S}_2, l_2, \prec_2)$ is an epistemic state $\mathbb{E}_1 \times \mathbb{E}_2 = (\mathcal{S}_1 \times \mathcal{S}_2, l, \prec)$ such that

- $l(s_1, s_2) = \mathrm{Th}(l_1(s_1) \cup l_2(s_2))$, for any $(s_1, s_2) \in \mathcal{S}_1 \times \mathcal{S}_2$;
- $(s_1, s_2) \preceq (t_1, t_2)$ iff $s_1 \preceq_1 t_1$ and $s_2 \preceq_2 t_2$.

A pure merge is a merge operation that treats the two component epistemic states as two equally reliable sources. It is easy to see, in particular, that it is a commutative operation.

Being applied to base-generated epistemic states, pure merge corresponds to a straightforward union of two bases:

Lemma 4.2. *If $\mathbb{E}_1$ and $\mathbb{E}_2$ are epistemic states generated, respectively, by bases Δ_1 and Δ_2, then $\mathbb{E}_1 \times \mathbb{E}_2$ is equivalent to an epistemic state generated by $\Delta_1 \cup \Delta_2$.*

4.2.2 Merging flocks

To begin with, the following result shows that pure merge preserves persistence of epistemic states.

Lemma 4.3. *A pure merge of two persistent epistemic states is also a persistent epistemic state.*

Since finitary persistent epistemic states are representable by flocks, a pure merge gives rise to a certain operation on flocks. This operation can be described as follows:

Let us consider two flocks $\mathcal{F}_1$ and $\mathcal{F}_2$ that have no propositions in common. Then a *merge* of $\mathcal{F}_1$ and $\mathcal{F}_2$ will be a flock

$$\mathcal{F}_1 \times \mathcal{F}_2 = \{\Delta_i \cup \Delta_j \mid \Delta_i \in \mathcal{F}_1, \Delta_j \in \mathcal{F}_2\}$$

Thus, the merge of two disjoint flocks is obtained by a pairwise combination of bases belonging to each flock. Note, however, that the assumption of disjointness turns out to be essential for establishing the correspondence between merge of flocks and pure merge of associated epistemic states. Still, this requirement is a purely syntactic constraint that can be easily met by replacing some of the propositions with logically equivalent, though syntactically different propositions. A suitable example will be given later when we will consider expansions of flocks that are based on the above notion of merge.

The following result shows that merge of flocks corresponds exactly to a pure merge of associated epistemic states.

Theorem 4.4. *If $\mathbb{E}_\mathcal{F}$ and $\mathbb{E}_\mathcal{G}$ are epistemic states generated, respectively, by disjoint flocks $\mathcal{F}$ and $\mathcal{G}$, then $\mathbb{E}_\mathcal{F} \times \mathbb{E}_\mathcal{G}$ is isomorphic to $\mathbb{E}_{\mathcal{F} \times \mathcal{G}}$.*

4.2.3 Pure expansions

For any proposition A, we will denote by $\mathbb{E}_A$ the epistemic state generated by a singular base $\{A\}$. This pure epistemic state consists of just two theories, namely $\mathrm{Th}(\emptyset)$ and $\mathrm{Th}(A)$. Accordingly, it gives a most 'pure' expression of the belief in A. Now the idea behind the definition below is that an expansion of an epistemic state with respect to A amounts to merging it with $\mathbb{E}_A$.

Definition 4.3. A *pure expansion* of an epistemic state $\mathbb{E}$ with respect to a proposition A (notation $\mathbb{E} + A$) is a pure merge of $\mathbb{E}$ and the epistemic state $\mathbb{E}_A$ that is generated by a base $\{A\}$.

In a pure expansion the new proposition is added as an independent piece of information, that is, as a proposi-

tion that is not related to others with respect to priority. Being applied to base-generated epistemic states, pure expansion corresponds to a straightforward addition of a new proposition to the base.

Corollary 4.5. *If $\mathbb{E}$ is generated by a base Δ, then $\mathbb{E}+A$ is isomorphic to an epistemic state generated by $\Delta \cup \{A\}$.*

Since pure merge is a commutative operation, pure expansions will also be commutative.

As any other kind of change in epistemic states, expansions generate corresponding changes in belief sets of epistemic states. It turns out that expansions generate in this sense precisely AGM belief expansion functions:

Lemma 4.6. *If $\mathbf{B}$ is a belief set of $\mathbb{E}$, then the belief set of $\mathbb{E}+A$ coincides with* $\mathrm{Th}(\mathbf{B}, A)$.

Thus, belief expansions generated by expansions of epistemic states behave just as AGM expansions: the underlying epistemic state plays no role in determining the resulting expanded belief set, since the latter can be obtained by a direct addition of new propositions to the source belief set. It should be kept in mind, however, that identical expansions of belief sets can be produced by expansions of different epistemic states, and even by different expansions of the same epistemic state. These differences will be revealed in subsequent contractions and revisions of the expanded belief set.

4.2.4 Expansions of flocks

As we have seen earlier, pure merge generates a corresponding merge operation on flocks. Consequently, pure expansion corresponds in this sense to a certain expansion operation on flocks.

Definition 4.4. An *expansion of a flock $\mathcal{F} = \{\Delta_i \mid i \in I\}$* with respect to a proposition A that does not appear in $\mathcal{F}$ is a flock $\mathcal{F}+A = \{\Delta_i \cup \{A\} \mid i \in I\}$.

Thus, an expansion of a flock is obtained simply by adding the new proposition to each base from the flock. Our earlier results immediately imply that this operation exactly corresponds to a pure expansion of the associated epistemic state with A:

Corollary 4.7. *If $\mathbb{E}_{\mathcal{F}}$ is an epistemic state generated by the flock $\mathcal{F}$, and A does not appear in $\mathcal{F}$, then $\mathbb{E}_{\mathcal{F}}+A$ is isomorphic to $\mathbb{E}_{\mathcal{F}+A}$.*

The above operation is quite similar in spirit to the operation of insertion into flocks used in [Fagin *et al.*, 1986], though the latter was intended to preserve consistency of the component bases, so they defined, in effect, the corresponding revision operation based on contraction and expansion in our sense. Note, however, that our flock expansion is defined only when the added proposition does not appear in the flock. The need for the restriction is illustrated by the following example.

Example 4.3. Let us return to the flock $\mathcal{F} = \{\{A\}, \{B\}\}$, where A and B denote, respectively, propositions "Niamey is a town in Nigeria", and "There is a university in Niamey". Recall that this flock is obtainable by contracting $A \wedge B$ from the base $\{A, B\}$. In other words, it reflects an informational situation in which we have reasons to believe in each of these propositions, but cannot believe in both.

Now let us expand the epistemic state $\mathbb{E}_{\mathcal{F}}$ with B. This expansion can be modeled by expanding $\mathcal{F}$ with some proposition B' that is logically equivalent to B. In other words, the epistemic state generated by the flock $\mathcal{F}+B' = \{\{B', A\}, \{B', B\}\}$ will be equivalent to the expansion of $\mathbb{E}_{\mathcal{F}}$ with B. This flock sanctions belief in B in full accordance with our intuitions. Actually, it can be shown that the latter flock is reducible to a flock $\{\{B', A\}, \{B\}\}$ in the sense that the latter flock will produce an equivalent epistemic state. However, B' cannot be replaced with B in these flocks: the flock $\{\{B\}, \{B, A\}\}$ is already reducible to a single base $\{\{A, B\}\}$ in which both A and B are believed, contrary to our intuitions about the relevant situation: receiving a new support for believing that there is a university in Niamey should not force us to believe also that Niamey is a town in Nigeria. This also shows most vividly that a straightforward addition of B to each base in the flock $\{\{A\}, \{B\}\}$ does not produce intuitively satisfactory results[1].

An additional interesting aspect of the above representation is that, though we fully believe in B in the flock $\{\{B\}, \{B', A\}\}$, the option A has not been forgotten; if we will contract now B from the latter flock, we will obtain the flock $\{\{A\}\}$ which supports belief in A. A little reflection shows that this is exactly what would be reasonable to believe in this situation.

5 Conclusions

The purpose of this study was to give a formal representation for iterative non-prioritized change. As has been shown, such a representation can be achieved in the framework of persistent epistemic states, with flocks providing the corresponding constructive representation. Moreover, these representations overcome shortcomings of both the AGM and base-oriented

[1] Exactly this was a by-product of the representation suggested in [Bochman, 1999].

models that incur loss of information in iterative changes.

Contraction and expansions are two basic operations on epistemic states that allow to define the majority of derived belief changes. Thus, a *revision* of an epistemic state is definable via Levi identity (on the level of epistemic states), namely as a combination of contraction and expansion. As can be shown, the resulting operation will be sufficiently expressive to capture any relational belief revision function in the sense of AGM.

To conclude the paper, we want to mention an interesting problem concerning expressivity of our belief change operations vis-a-vis flocks. Namely, it has been shown in [Bochman, 2001b] that there are flocks that are not constructible from simple primitive flocks using the contraction and expansion operations (a simplest example being the flock $\{\{p\},\{p \wedge q\}\}$). This apparently suggests that our stock of belief change operations is not complete and need to be extended with other operations that would provide functional completeness with respect to constructibility of flocks. The resulting theory would give then a truly complete constructive representation of non-prioritized belief change.

References

[Alchourrón *et al.*, 1985] C. Alchourrón, P. Gärdenfors, and D. Makinson. On the logic of theory change: Partial meet contraction and revision functions. *J. of Symbolic Logic*, 50:510–530, 1985.

[Benthem, 1988] J. Van Benthem. *A Manual of Intensional Logic.* CSLI, Stanford, second edition, 1988.

[Bochman, 1999] A. Bochman. A foundational theory of belief and belief change. *Artificial Intelligence*, 108:309–352, 1999.

[Bochman, 2000] A. Bochman. A foundationalist view of the AGM theory of belief change. *Artificial Intelligence*, 116:237–263, 2000.

[Bochman, 2001a] A. Bochman. Contraction of epistemic states: A general theory. In H. Rott and M.-A. Williams, editors, *Frontiers in Belief Revision*, pages 195–220. Kluwer, 2001.

[Bochman, 2001b] A. Bochman. *A Logical Theory of Nonomonotonic Inference and Belief Change.* Springer, 2001.

[Fagin *et al.*, 1986] R. Fagin, J. D. Ullman, G. M. Kuper, and M. Y. Vardi. Updating logical databases. In *Advances in Computing Research*, volume 3, pages 1–18, 1986.

[Gärdenfors and Rott, 1995] P. Gärdenfors and H. Rott. Belief revision. In D. M. Gabbay et al., editor, *Handbook of Logic in Artificial Intelligence and Logic Programming. Vol. 4*, pages 35–132. Clarendon Press, 1995.

[Gärdenfors, 1988] P. Gärdenfors. *Knowledge in Flux: Modeling the Dynamics of Epistemic States.* Bradford Books, MIT Press, 1988.

[Hansson, 1992] S. O. Hansson. In defense of base contraction. *Synthese*, 91:239–245, 1992.

[Hansson, 1993] S. O. Hansson. Theory contraction and base contraction unified. *J. of Symbolic Logic*, 58:602–625, 1993.

[Hansson, 1999] S. O. Hansson. *A Textbook of Belief Dynamics.* Kluwer, 1999.

Revising partially ordered beliefs

Salem Benferhat
IRIT, Univerité Paul Sabatier,
118, route de Narbonne,
31062 Toulouse Cedex 4, France
benferhat@irit.fr

Sylvain Lagrue
PRAXITEC
115, rue St Jacques,
13006 Marseille, France
lagrue@irit.fr

Odile Papini
SIS, Université de Toulon et du Var,
avenue de l'université BP 132,
83957 La Garde cedex, France
papini@univ-tln.fr

Abstract

This paper deals with the revision of partially ordered beliefs. It proposes a semantic representation of epistemic states by partial preorders on interpretations and a syntactic representation by partially ordered belief bases. Two revision operations, the revision stemming from the history of observations and the possibilistic revision, defined when the epistemic state is represented by a total preorder, are generalized, at a semantic level, to the case of a partial pre-order on interpretations, and at a syntactic level, to the case of a partially ordered belief base. The equivalence between the two representations is shown for the two revision operations.

1 Introduction

Most of the time, an intelligent agent faces incomplete, uncertain or inaccurate information. The arrival of a new item of information, more reliable or more certain leads the agent to refine (specify) his beliefs, to revise them. Belief revision is a well known problem in Artificial intelligence [1], [9], [13], in this context, an epistemic state encodes a set of beliefs about the real world (based on the available information). An epistemic state is generally interpreted as a plausibility ordering between possible states of the world, or as a preference relation between information sources from which an agent can derive his beliefs.

On a semantic level, epistemic states have been represented by a total pre-order on interpretations of the underlying logical language [5]. This total pre-order models the agent's preferences between several situations. This pre-order has been encoded according to several ways, ordinals [16], [17], possibilities [7], polynomials [15].

However, the agent has not always a total pre-order between situations at his disposal, but is only able to define a partial pre-order between situations. The arrival of successive items of information can help him to refine this partial pre-order in order to converge to a total pre-order between situations.

In other respects, total pre-orders are not suitable to model the case where decision making is impossible or not arbitrary. Suppose the agent has to make a decision on the cultivation of a new plot according to three rules given by experts. A first rule, $R1$ specifies that if some agronomical conditions hold (warm climate, deep soil, acidity, etc...), called *condition 1*, then *the cultivation of tobacco is feasible.* The second rule $R2$ specifies that if the agronomical conditions (*condition 1*) hold and the zone is mildewed (mildew parasite could ruin the plantation) called *condition 2*, then *the cultivation of tobacco is not feasible.* The third rule $R3$ specifies that according to the regulation of production of tobacco, if the area of the plot is not greater than the authorized area, called *condition 3*, then *the cultivation of tobacco is feasible.*

The question is to define a pre-order on the three rules in order to make a decision on the cultivation of tobacco. Since $R2$ is more specific than $R1$, it is natural to prefer R_2 over R_1 (namely $R2 < R1$ holds). However, *condition 3* is not related with *condition 1* nor with *condition 2*, then it seems reasonable to consider $R3$ incomparable with $R1$ and $R2$. On contrast, if we want to impose a total pre-oder on rules, we have to set $R3$ relatively to $R2$, therefore there are two intuitive choices, either $R1 \leq R3$ or $R3 \leq R2$. In the first case, the following total pre-order holds: $R2 < R1 \leq R3$ this means that if *condition 1*, *condition 2* and *condition 3* are both satisfied, according to the total pre-order we make the decision that *the cultivation of tobacco is not feasible.* In the second case, the following total pre-order holds: $R3 \leq R2 < R1$ this means that if *condition 1*, *condition 2* and *condition 3* are satisfied,

according to the total pre-order we make the decision that *the cultivation of tobacco is feasible.* These two total pre-orders lead to contradictory decisions, and there is no reason to choose the first one or the second one, the choice can only be arbitrary. Since we think that arbitrary choices have to be excluded, we think that a better solution is to consider $R3$ and $R2$ as incomparable (and hence all total pre-orders are considered), and thus to define a partial pre-order between rules. In such cases, an epistemic state has to be represented by a partial pre-order on interpretations and revision operations of partial pre-orders by formulas have to be defined. Partial pre-orders on interpretations have been used to represent update operations [12], this paper does not address updates but revisions.

In the present paper, we propose a generalization of two revision operators to the case of partial pre-orders. Section 2 presents the problematics of the representation of epistemic states by a partial pre-order and focuses on the difficulties of this generalization. Section 3 presents the generalization of the semantic revision operations to partial pre-orders and the generalization of the syntactic counterparts of these operations to partially ordered belief bases. These generalizations are rather direct. On contrast the generalization of the mapping from the syntactic level to the semantic level is more problematical as shown in Section 4, because the definition of a partial pre-order between formulas leads to two possible partial pre-orders between subsets of formulas. We choose one of them, however the results presented hold for the other one. We finally present the syntactic computation of the belief set corresponding to an epistemic state in Section 5 before a concluding discussion in Section 6.

2 Presentation of the problem

2.1 Basic definitions of partial pre-orders

In this paper we use propositional calculus, denoted by $\mathcal{L}_{PC}$, as knowledge representation language with usual connectives $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\equiv$ (logical equivalence). The lower case letters a, b, c, $\cdots$, are used to denote propositional variables, lower case Greeks letters ϕ, ψ, $\cdots$, are used to denote formulas, upper case letters A, B, C, are used to denote sets of formulas, and upper case Greeks letters Ψ, Φ $\cdots$, are used to denote epistemic states. We denote by $\mathcal{W}$ the set of interpretations of $\mathcal{L}_{PC}$ and by $Mod(\psi)$ the set of models of a formula ψ, that is $Mod(\psi) = \{\omega \in \mathcal{W}, \omega \models \psi\}$ where $\models$ denotes the inference relation used for drawing conclusions.

A partial pre-order, denoted by $\preceq$ on a set A is a reflexive and transitive binary relation. Let x and y be two members of A, the equality is defined by $x = y$ iff $x \preceq y$ and $y \preceq x$. The corresponding strict partial pre-order, denoted by $\prec$, is such that, $x \prec y$ iff $x \preceq y$ holds but $x \preceq y$ does not hold. We denote by $\sim$ the incomparability relation $x \sim y$ iff $x \preceq y$ does not hold nor $y \preceq x$.

$\preceq$ can equivalently be defined from $=$ and $\prec$: $a \preceq b$ iff $a \prec b$ or $a = b$.

Given $\preceq$ on a set A, the minimal elements of A are defined by: $min(A, \preceq) = \{x : x \in A, \nexists y \in A, y \prec x\}$

For the purpose of this paper, we need to define a partial pre-order on subsets of elements of a set A. According to Halpern's works [10], [11] (see also Cayrol et al. [4], Dubois et al. [6] and Lafage et al. [14]) there are two meaningful ways to compare subsets of A. We denote these two partial pre-orders on 2^A by $\preceq_{A,w}$ and by $\preceq_{A,s}$.

Let X and Y be two subsets of A, The sets X and Y are considered equals if for each preferred element in X there exists a so preferred element in Y and conversely, more formally:

Definition 1 *Let X and Y be two subsets of A and $\preceq_A$ a partial pre-order on A, $X = Y$ iff :*

$$\forall x \in min(X, \preceq_A),\ \exists y \in min(Y, \preceq_A) \text{ such that } x = y$$
$$\text{and}$$
$$\forall y \in min(Y, \preceq_A),\ \exists x \in min(X, \preceq_A) \text{ such that } x = y.$$

The first way to define a partial pre-order on subsets X and Y of A is to consider that X is preferred to Y if for each element of Y there exists at least one element in X, which is preferred to it, more formally (we assume that X and Y are not both empty):

Definition 2 *X is weakly preferred to Y, denoted by $X \prec_{A,w} Y$, iff $\forall y \in Min(Y, \preceq_A)$, $\exists x \in Min(X, \preceq_A)$ such that $x \prec_A y$.*

The second way to define a partial pre-order on subsets X and Y of A is to consider that X is preferred to Y if there exists at least one element in X which is preferred to all elements in Y, more formally:

Definition 3 *Let $\preceq_A$ be a partial pre-order on A and $X, Y \subseteq A$. X is strongly preferred to Y, denoted by $X \prec_{A,s} Y$ iff $\exists x \in Min(X, \preceq_A)$ such that $\forall y \in Min(Y, \preceq_A)$, $x \prec_A y$.*

It can be shown that the definition of $\prec_{A,s}$ implies the definition of $\prec_{A,w}$, namely, if $X \prec_{A,s} Y$ then $X \prec_{A,w} Y$. The converse does not hold.

Example 1 *Let $A = \{x_1, x_2, y_1, y_2\}$ and $\preceq_A$ be a partial pre-order on A such that $x_1 \preceq_A y_1$ and $x_2 \preceq_A y_2$. Let X and Y be two subsets of A, $X = \{x_1, x_2\}$ and $Y = \{y_1, y_2\}$, we have $X \prec_{A,w} Y$, indeed x_1 is preferred to y_1 and x_2 is preferred to y_2. However, $X \prec_{A,s} Y$ does not hold, there is no element in X which is preferred to all elements of Y.*

In the case where $\preceq_A$ is a total pre-order, the definition of $\prec_{A,s}$ is equivalent to the definition of $\prec_{A,w}$, more formally, $X \prec_{A,s} Y$ iff $X \prec_{A,w} Y$.

For lake of space, we will only focus on the weakly preference definition. But all provided results are valid for the strong preference.

2.2 Semantic representation of epistemic states

Let Ψ be an epistemic state, Ψ is first represented by a partial pre-order on interpretations, denoted by $\preceq_\Psi$. The interpretation ω is preferred (or more plausible than) to ω', denoted $\omega \preceq_\Psi \omega'$. $\omega \sim_\Psi \omega'$ denotes that the agent has no preference between ω and ω'. The belief set corresponding to Ψ, denoted by $Bel^{se}(\Psi)$, modeling the agent's current beliefs is such that $Mod(Bel^{se}(\Psi)) = min(\mathcal{W}, \preceq_\Psi)$. We illustrate this representation with the example informally described in the introduction.

Example 2 *The agent has to make a decision on the cultivation of a new plot according tree rules given by experts. A first rule, R1 specifies that if some agronomical conditions hold (warm climate, deep soil, acidity, etc...), called condition 1, then the cultivation of tobacco is feasible. The second rule R2 specifies that if the agronomical conditions (condition 1) hold and the zone is mildewed (mildew parasite could ruin the plantation) called condition 2, then the cultivation of tobacco is not feasible. The third rule R3 specifies that according to the regulation of the production of tobacco, if the area of the plot is not greater than the authorized area, called condition 3, then the cultivation of tobacco is feasible. More formally, the three rules can be represented as follows: R1: $b \rightarrow a$, R2: $b \wedge c \rightarrow \neg a$, R3: $d \rightarrow a$, where a encodes the cultivation of tobacco is feasible, b encodes condition 1, c encodes condition 2 and d encodes condition 3.*

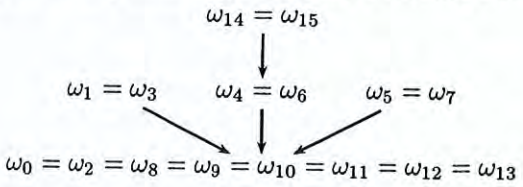
There are four propositional variables a, b, c and d. The sixteen interpretations are $\omega_0 = \{\neg a, \neg b, \neg c, \neg d\}$, $\omega_1 = \{\neg a, \neg b, \neg c, d\}$, $\omega_2 = \{\neg a, \neg b, c, \neg d\}$, $\cdots$, $\omega_{14} = \{a, b, c, \neg d\}$, $\omega_{15} = \{a, b, c, d\}$. Let Ψ be an epistemic state which corresponding belief set is $Bel^{se}(\Psi) = (b \rightarrow a) \wedge (b \wedge c \rightarrow \neg a) \wedge (d \rightarrow a)$, we represent the epistemic state by a partial pre-order, denoted $\preceq_\Psi$ as follows:

Since R2 is more specific than R1, and since R2 and R3 are incomparable, then:

- *the interpretations satisfying all constraints are preferred to all other interpretations,*
- *the interpretations which falsify R_2 are preferred to the interpretations falsifying R_1,*
- *the interpretations which falsify R_2 and the ones which falsify R_3 are uncomparable.*

The partial pre-order $\preceq_\Psi$ is represented by Figure 1 (an arrow $x \longleftarrow y$ means $x \prec y$, the transitivity and the reflexivity are not represented for sake of clarity).

Figure 1: Representation of initial epistemic state Ψ



2.3 Syntactic representation of epistemic states

An epistemic state Ψ is syntactically represented by a partially ordered belief base, denoted by $\preceq_\Sigma$, where Σ is a set of propositional formulas, and $\preceq_\Sigma$ is a partial pre-order on Σ. Let ϕ and $\phi' \in \Sigma$, $\phi \preceq_\Sigma \phi'$ means that ϕ is preferred (more important than) to ϕ' and $\phi \sim_\Sigma \phi'$ denotes that the agent has no preference between ϕ and ϕ'. We illustrate this representation with the example informally described in the introduction.

Example 3 *Let Ψ be the epistemic state, where $\Sigma = \{b \rightarrow a, b \wedge c \rightarrow \neg a, d \rightarrow a\}$, we represent the epistemic state by a partial pre-order on Σ, denoted by $\preceq_\Sigma$, as follows: Since $b \wedge c \rightarrow \neg a$ is more specific than $b \rightarrow a$, and $d \rightarrow a$ and $b \wedge c \rightarrow \neg a$ are incomparable, the following partial pre-order on formulas holds:*

$b \wedge c \rightarrow \neg a \prec_\Sigma b \rightarrow a$ and $b \wedge c \rightarrow \neg a \sim_\Sigma d \rightarrow a$ and $b \rightarrow a \sim_\Sigma d \rightarrow a$.

The generalization of the representation and revision of an epistemic state to a partial pre-order leads to the following diagram:

$$\begin{array}{ccc} \preceq_\Sigma & \rightarrow & \preceq_\Psi \\ \downarrow & & \downarrow \\ \preceq_{\Sigma \circ^{sy} \mu} & \rightarrow & \preceq_{\Psi \circ^{se} \mu} \\ \downarrow & & \downarrow \\ Bel^{sy}(\Psi \circ^{sy} \mu) & \equiv & Bel^{se}(\Psi \circ^{se} \mu) \end{array}$$

In the special case of a total pre-order, the diagram has been shown valid and the equivalence between the syntactic approach and the semantic approach has been proved [2]. The question is what does remain true when this diagram is extended to the representation by a partial pre-order. We show in Section 3 that we get the mappings $\preceq_\Sigma \rightarrow \preceq_{\Sigma \circ^{sy} \mu}$, $\preceq_\Psi \rightarrow \preceq_{\Psi \circ^{se} \mu}$ and $\preceq_{\Psi \circ^{se} \mu} \rightarrow Bel^{se}(\Psi \circ^{se} \mu)$ rather directly. On contrast, the mappings $\preceq_\Sigma \rightarrow \preceq_\Psi$ and $\preceq_{\Sigma \circ^{sy} \mu} \rightarrow Bel^{sy}(\Psi \circ^{sy} \mu)$ are less straightforward.

We now present the revision extended to a partial pre-order.

3 Semantic and syntactic revision of partial pre-orders

3.1 Extension of the revision stemming from the history of observations

We extend the revision operation, defined in [2, 15], to the case of partial pre-orders. The underlying intuition stems from the fact that the agent remembers all his previous observations. However these observations are not at the same level, according to whether there are more plausible or not in the next epistemic state. The general philosophy is that an old assertion is less reliable than a new one. In prediction problems, it seems reasonable to decrease the confidence that one has in an item of information, as time goes by. However, this revision operation attempts to satisfy as many previous observations as possible. That is, an old observation persists until it becomes contradictory with a more recent one. The revision operation uses the history of the sequence of previous observations to perform revision.

3.1.1 Semantic extension

When an epistemic state, Ψ, is represented by a partial pre-order on interpretations, the revision of Ψ by a propositional formula μ leads to a revised epistemic state $\Psi \circ^{se}_{\triangleright} \mu$, represented by a partial pre-order on interpretations. This new epistemic state is such that the relative ordering between models of μ is preserved, the relative ordering between counter-models of μ is preserved, and the models of μ are preferred to its counter-models. More formally:

Definition 4 *Let Ψ be an epistemic state and μ be a propositional formula, the revised epistemic state $\Psi \circ^{se}_{\triangleright} \mu$ corresponds to the following partial pre-order:*

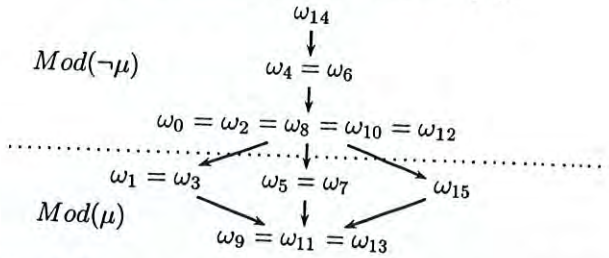
- *if $\omega, \omega' \in Mod(\mu)$ then $\omega \preceq_{\Psi \circ^{se}_{\triangleright} \mu} \omega'$ iff $\omega \preceq_\Psi \omega'$,*
- *if $\omega, \omega' \notin Mod(\mu)$ then $\omega \preceq_{\Psi \circ^{se}_{\triangleright} \mu} \omega'$ iff $\omega \preceq_\Psi \omega'$,*
- *if $\omega \in Mod(\mu)$ and $\omega' \notin Mod(\mu)$ then $\omega \prec_{\Psi \circ^{se}_{\triangleright} \mu} \omega'$.*

According to this definition it is easy to check that $Mod(Bel^{se}(\Psi \circ^{se}_{\triangleright} \mu)) = min(Mod(\mu), \preceq_\Psi)$.

Example 4 *We come back to example 2, where the initial epistemic state Ψ is represented by the Figure 1.*

The corresponding belief set $Bel^{se}(\Psi)$ is such that $Mod(Bel^{se}(\Psi)) = \{\omega_0, \omega_2, \omega_8, \omega_9, \omega_{10}, \omega_{11}, \omega_{12}, \omega_{13}\}$. Suppose we learn that condition 3 holds, namely we revise Ψ by the propositional formula $\mu = d$. According to the definition 4 the revised epistemic state $\Psi \circ^{se}_{\triangleright}$ is represented by the partial pre-order on interpretations $\preceq_{\Psi \circ^{se}_{\triangleright}}$ graphically represented on Figure 2.

Figure 2: Representation of $\Psi \circ^{se}_{\triangleright} \mu$



The belief set corresponding to $\Psi \circ^{se}_{\triangleright} \mu$ is such that $Mod(Bel^{se}(\Psi \circ^{se}_{\triangleright} \mu)) = \{\omega_9, \omega_{11}, \omega_{13}\}$ and since $Mod(\mu) = \{\omega_1, \omega_3, \omega_5, \omega_7, \omega_9, \omega_{11}, \omega_{13}, \omega_{15}\}$, it can be checked that
$Mod(Bel^{se}(\Psi \circ^{se}_{\triangleright} \mu)) = min(Mod(\mu), \preceq_\Psi)$.

3.1.2 Syntactic extension

We now present the syntactic extension of this revision operation to partial pre-orders. Let Ψ be an epistemic state, Ψ is syntactically represented by a partially ordered belief base, denoted by $\preceq_\Sigma$, where Σ is a set of propositional formulas, and $\preceq_\Sigma$ is a partial pre-order on Σ. The revision of $\preceq_\Sigma$ by a propositional formula μ leads to a partially ordered belief base denoted by $\preceq_{\Sigma \circ^{sy}_{\triangleright} \mu}$ as follows:

Let us denote by U the set of the disjunctions between μ and the formulas of Σ, more formally, $U = \{\phi \vee \mu$, such that $\phi \in \Sigma$ and $\phi \vee \mu \not\equiv \top\}$.

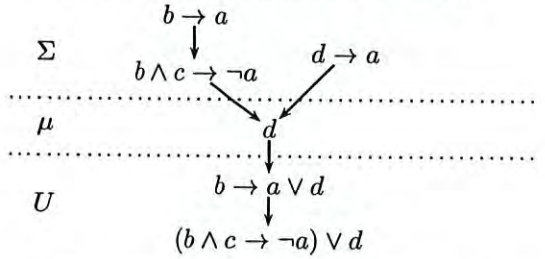
Definition 5 *Let Ψ be an epistemic state, represented by a partially ordered belief base $\preceq_\Sigma$, the revision of Ψ by μ leads to a revised epistemic state $\Psi \circ^{sy}_{\triangleright} \mu$ represented by a partially ordered belief base where $\Sigma \circ^{sy}_{\triangleright} \mu = \Sigma \cup U \cup \{\mu\}$ and $\preceq_{\Sigma \circ^{sy}_{\triangleright} \mu}$ is such that:*

- $\forall \phi \vee \mu \in U : \ \phi \vee \mu \prec_{\Sigma \circ_\flat^{sy} \mu} \mu$,
- $\forall \phi \in \Sigma : \ \mu \prec_{\Sigma \circ_\flat^{sy} \mu} \phi$,
- $\forall \phi, \phi' \in \Sigma : \phi \preceq_\Sigma \phi'$ *iff* $\phi \preceq_{\Sigma \circ_\flat^{sy} \mu} \phi'$,
- $\forall \phi, \phi' \in \Sigma : \phi \preceq_\Sigma \phi'$ *iff* $\phi \vee \mu \preceq_{\Sigma \circ_\flat^{sy} \mu} \phi' \vee \mu$.

Example 5 *We come back to example 3. Let Ψ be the epistemic state, where $\Sigma = \{b \rightarrow a, b \wedge c \rightarrow \neg a, d \rightarrow a\}$, $\preceq_\Sigma$ is such that it only contains one constraint: $b \wedge c \rightarrow \neg a \prec_\Sigma b \rightarrow a$.*

Let us revise Ψ by the propositional formula $\mu = d$. According to the definition of the revision operation $\circ_\flat^{sy}$, the revised epistemic state $\Psi \circ_\flat^{sy} \mu$ is represented by the following partial pre-order on Figure 3.

Figure 3: Representation of $\Psi \circ_\flat^{sy} \mu$



Remark that since $(d \rightarrow a) \vee d$ is a tautology, we do not take it into account.

3.2 Extension of possibilistic revision

We now present the extension of the possibilistic revision to partial pre-orders. In a possibility theory framework [6] an epistemic state Ψ is represented by a possibility distribution π. Each interpretation is assigned with a real number belonging to the interval $[0,1]$. The value 1 means that the interpretation is totally possible, whereas the value 0 means that the interpretation is totally impossible. A possibility distribution induces a total pre-order $\leq_\pi$ on interpretations in the following way: $\omega \leq_\pi \omega'$ iff $\pi(\omega) \geq \pi(\omega')$. For more details on possibility theory and the revision of possibility distributions, see [7, 8].

3.2.1 Semantic extension

Let Ψ be an epistemic state represented by a partial pre-order $\leq_\Psi$. The possibilistic revision of Ψ by a propositional formula μ leads to a revised epistemic state $\Psi \circ_\pi^{se} \mu$, represented by a partial pre-order on interpretations, denoted by $\preceq_{\Psi \circ_\pi^{se} \mu}$ which considers that all the counter-models of the new item of information μ as impossible and preserves the relative ordering between the models of μ. More formally:

Definition 6 *Let Ψ be an epistemic state and μ be a propositional formula, the revised epistemic state $\Psi \circ_\pi^{se} \mu$ corresponds to the following partial pre-order:*

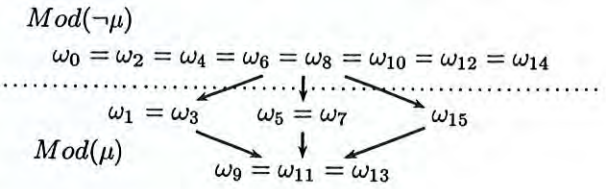
- *if $\omega, \omega' \in Mod(\mu)$ then $\omega \preceq_{\Psi \circ_\pi^{se} \mu} \omega'$ iff $\omega \preceq_\Psi \omega'$,*
- *if $\omega, \omega' \notin Mod(\mu)$ then $\omega =_{\Psi \circ_\pi^{se} \mu} \omega'$,*
- *if $\omega \in Mod(\mu)$ and $\omega' \notin Mod(\mu)$ then $\omega \prec_{\Psi \circ_\pi^{se} \mu} \omega'$.*

According to this definition it is easy to check that $Mod(Bel^{se}(\Psi \circ_\pi^{se} \mu)) = min(Mod(\mu), \preceq_\Psi)$.

Example 6 *Let us consider again example 2.*

We revise Ψ by the propositional formula $\mu = d$. According to the definition 6, the revised epistemic state $\Psi \circ_\pi^{se} \mu$ is represented by the following partial pre-order on interpretations $\preceq_{\Psi \circ_\pi^{se} \mu}$, graphically represented on Figure 4.

Figure 4: Representation of $\Psi \circ_\pi^{se} \mu$



The belief set corresponding to $\Psi \circ_\pi^{se} \mu$ is such that $Mod(Bel^{se}(\Psi \circ_\pi^{se} \mu)) = \{\omega_9, \omega_{11}, \omega_{13}\}$.

3.2.2 Syntactic extension

We now present the syntactic extension of the possibilistic revision operation to partial pre-orders. Let Ψ be an epistemic state, syntactically represented by a partially ordered belief base. The revision of $\preceq_\Sigma$ by a propositional formula μ leads to a partially ordered belief base denoted by $\preceq_{\Sigma \circ_\pi^{sy} \mu}$ as follows:

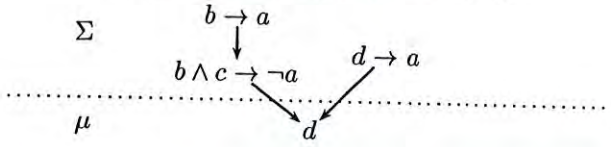
Definition 7 *The revision of Ψ by μ leads to a revised epistemic state $\Psi \circ_\pi^{sy} \mu$ represented by a partially ordered belief base $\Sigma \circ_\pi^{sy} \mu = \Sigma \cup \{\mu\}$ where $\preceq_{\Sigma \circ_\pi^{sy} \mu}$ is such that:*

- $\forall \phi \in \Sigma: \quad \mu \prec_{\Sigma \circ_\pi^{sy} \mu} \phi$,
- $\forall \phi, \phi' \in \Sigma$: $\phi \preceq_\Sigma \phi'$ *iff* $\phi \preceq_{\Sigma \circ_\pi^{sy} \mu} \phi'$.

Example 7 *We come back to example 3, where $\Sigma = \{b \rightarrow a, b \wedge c \rightarrow \neg a, d \rightarrow a\}$, and $\preceq \Sigma$ only contains one constraint: $b \wedge c \rightarrow \neg a \prec_\Sigma b \rightarrow a$.*

Let us revise Ψ by the propositional formula $\mu = d$. According to the definition 7, xthe revised epistemic state $\Psi \circ_\pi^{sy} \mu$ is represented by the following partial pre-order on $\Sigma \circ_\pi^{sy} \mu$ denoted by $\preceq_{\Sigma \circ_\pi^{sy} \mu}$ and graphically represented by Figure 5.

Figure 5: Representation of $\Psi \circ_\pi^{sy} \mu$



Note that revising a partial pre-order, with $\circ_\pi^{se}$ and $\circ_\triangleright^{se}$, carry away some incomparabilities. Hence, after a certain number of successive revisions the resulting partial pre-order on interpretations converges to a total pre-order on interpretations, more formally:

Proposition 1 *Let $\preceq_\Psi$ be a partial pre-order on interpretations, there exists a sequence of formulas $(\mu_1, \mu_2, \cdots, \mu_n)$ such that the resulting partial pre-order after successive revisions*

- $(((\preceq_\Psi \circ_\triangleright^{se} \mu_1) \circ_\triangleright^{se} \mu_2) \circ_\triangleright^{se} \cdots \circ_\triangleright^{se} \mu_n)$ *is a total pre-order, and*
- $(((\preceq_\Psi \circ_\pi^{se} \mu_1) \circ_\pi^{se} \mu_2) \circ_\pi^{se} \cdots \circ_\pi^{se} \mu_n)$ *is a total pre-order.*

The interest of such a result stems from the fact that starting from total ignorance about a topic, successive revisions lead to a partial pre-order on interpretations, and we now know how to perform these revisions. Moreover, after a certain number of revision the partial pre-order converges to a total pre-order that can be revised according to the results previously obtained in [2], [7].

4 From syntax to semantics

We now present the mapping from a partially ordered belief base $\preceq_\Sigma$ to a partial pre-order on interpretation $\preceq_\Psi$.

Definition 8 *Let Σ be a partially ordered belief base and ω be an interpretation. We denote by $\lceil \omega, \Sigma \rceil$ the set of preferred formulas of Σ falsified by ω. We define a partial pre-order on interpretations as follow:*

$$\omega \preceq_{\Psi,w} \omega' \text{ iff } \lceil \omega', \Sigma \rceil \preceq_{\Sigma,w} \lceil \omega, \Sigma \rceil.$$

Where $\preceq_{\Sigma,w}$ is given by definition 2

Example 8 *We come back to example 3,where $\Sigma = \{b \to a, b \wedge c \to \neg a, d \to a\}$, and $\preceq_\Sigma$ is defined as $b \wedge c \to \neg a \prec_\Sigma b \to a$.*

The sets of preferred formulas of Σ falsified by the interpretations are the following:

$$\lceil \omega_0, \Sigma \rceil = \lceil \omega_2, \Sigma \rceil = \lceil \omega_8, \Sigma \rceil = \lceil \omega_9, \Sigma \rceil = \emptyset,$$
$$\lceil \omega_{10}, \Sigma \rceil = \lceil \omega_{11}, \Sigma \rceil = \lceil \omega_{12}, \Sigma \rceil = \lceil \omega_{13}, \Sigma \rceil = \emptyset,$$
$$\lceil \omega_1, \Sigma \rceil = \lceil \omega_3, \Sigma \rceil = \{d \to a\},$$
$$\lceil \omega_4, \Sigma \rceil = \lceil \omega_6, \Sigma \rceil = \{b \to a\},$$
$$\lceil \omega_5, \Sigma \rceil = \lceil \omega_7, \Sigma \rceil = \{b \to a, d \to a\},$$
$$\lceil \omega_{14}, \Sigma \rceil = \lceil \omega_{15}, \Sigma \rceil = \{b \wedge c \to \neg a\}.$$

According to definition of $\preceq_{\Sigma,w}$, it can be easily checked that the computation of $\preceq_{\Sigma,w}$ leads to the same partial pre-order than the one used in the semantic representation of Ψ in example 2, namely $\preceq_{\Sigma,w} = \preceq_\Psi$.

We are now able to establish the equivalence between the syntactic representation of epistemic states by means of partially ordered belief bases and the semantic representation of epistemic states by means partial pre-orders on interpretations.

Theorem 1 *Let Ψ be an epistemic state represented, on one hand by a partially ordered belief base $\preceq_\Sigma$ and on the other hand by a partial pre-orders on interpretations $\preceq_\Psi$. Let $\circ_\triangleright^{sy}$ and $\circ_\triangleright^{se}$ be the syntactic and semantic revision operators stemming from the history of the observations, and $\circ_\pi^{sy}$ and $\circ_\pi^{se}$ be the syntactic and semantic possibilistic revision operators. Let we be the mapping from a partially ordered belief base $\preceq_\Sigma$ to a partial pre-order on interpretation $\preceq_\Psi$. The following result holds:*

- $we(\preceq_\Sigma \circ_\triangleright^{sy} \mu) = we(\preceq_\Sigma) \circ_\triangleright^{se} \mu$,
- $we(\preceq_\Sigma \circ_\pi^{sy} \mu) = we(\preceq_\Sigma) \circ_\pi^{se} \mu$.

We illustrate this theorem by the following example.

Example 9 *Let Ψ be the epistemic state of example 3, where $\Sigma = \{b \to a, b \wedge c \to \neg a, d \to a\}$, and $\preceq_\Sigma$ is such that: $b \wedge c \to \neg a \prec_\Sigma b \to a$.*

According to example 5 the revised epistemic state $\Psi \circ_\triangleright^{sy} \mu$ is represented by the partial pre-order on $\Sigma \circ_\triangleright^{sy} \mu$ given by figure 3.

The sets of preferred formulas of $\Sigma \circ_\triangleright^{sy} \mu = \{b \to a, b \wedge c \to \neg a, d \to a, d, (b \to a) \vee d, (b \wedge c \to \neg a) \vee d, (d \to a) \vee d\}$ falsified by the interpretations are the following:

$\lceil\omega_9, \Sigma \circ_\triangleright^{sy} \mu\rceil = \lceil\omega_{11}, \Sigma \circ_\triangleright^{sy} \mu\rceil = \lceil\omega_{13}, \Sigma \circ_\triangleright^{sy} \mu\rceil = \emptyset$,
$\lceil\omega_1, \Sigma \circ_\triangleright^{sy} \mu\rceil = \lceil\omega_3, \Sigma \circ_\triangleright^{sy} \mu\rceil = \{d \rightarrow a\}$,
$\lceil\omega_5, \Sigma \circ_\triangleright^{sy} \mu\rceil = \lceil\omega_7, \Sigma \circ_\triangleright^{sy} \mu\rceil = \{b \rightarrow a, d \rightarrow a\}$,
$\lceil\omega_{15}, \Sigma \circ_\triangleright^{sy} \mu\rceil = \{b \wedge c \rightarrow \neg a\}$,
$\lceil\omega_0, \Sigma \circ_\triangleright^{sy} \mu\rceil = \lceil\omega_2, \Sigma \circ_\triangleright^{sy} \mu\rceil = \lceil\omega_8, \Sigma \circ_\triangleright^{sy} \mu\rceil =$
$= \lceil\omega_{10}, \Sigma \circ_\triangleright^{sy} \mu\rceil = \lceil\omega_{12}, \Sigma \circ_\triangleright^{sy} \mu\rceil = \{d\}$,
$\lceil\omega_4, \Sigma \circ_\triangleright^{sy} \mu\rceil = \lceil\omega_6, \Sigma \circ_\triangleright^{sy} \mu\rceil = \{(b \rightarrow a) \vee d\}$,
$\lceil\omega_{14}, \Sigma \circ_\triangleright^{sy} \mu\rceil = \{(b \wedge c \rightarrow \neg a) \vee d\}$.

According to definition 4 it can be checked that $\preceq_{\Psi \circ_\triangleright^{se} \mu, w}$ *is identical to the one given by figure 3.*

Since from example 8 $we(\preceq_\Sigma) = \preceq_\Psi$ *we have* $we(\preceq_\Sigma) \circ_\triangleright^{se} \mu = \preceq_{\Psi \circ_\triangleright^{se} \mu}$ *and* $we(\preceq_\Sigma \circ_\triangleright^{sy} \mu) = \preceq_{\Psi \circ_\triangleright^{se} \mu, s}$ *it can be checked that* $\preceq_{\Psi \circ_\triangleright^{se} \mu, s} = \preceq_{\Psi \circ_\triangleright^{se} \mu}$, *therefore* $we(\preceq_\Sigma \circ_\triangleright^{sy} \mu) = we(\preceq_\Sigma) \circ_\triangleright^{se} \mu$ *and the equivalence between syntactic representation and semantic representation holds.*

5 Syntactic computation of $Bel^{sy}(\Psi)$

Let Ψ be an epistemic state represented by a partially ordered belief base $\preceq_\Sigma$, we now present the mapping from Σ to $Bel^{sy}(\Psi)$. The computation of the corresponding belief set $Bel^{sy}(\Psi)$ is more complex and it is closely linked to the ability of making syntactic inferences from $\preceq_\Sigma$ in order to deduce the agent's current beliefs. This involves the definition of a partial pre-order between consistent subsets of Σ, denoted by $\mathcal{C}$ and we generate a partial pre-order between consistent subsets of Σ using $\preceq_\Sigma$ described in definition 2. or

Definition 9 *Let* $C, C' \in \mathcal{C}$: $C \preceq_{\mathcal{C},w} C'$ *iff* $\{\phi' : \phi' \notin C'\} \preceq_{\Sigma,w} \{\phi : \phi \notin C\}$,

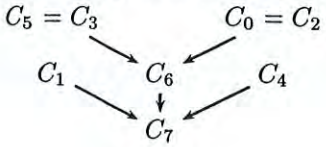
Intuitively, C is preferred to C' if the best formulas outside C are less preferred than the best formulas outside C'. We denote by $CONS_w(\Sigma)$ the set of preferred consistent subsets of Σ with respect to $\preceq_{\mathcal{C},w}$, namely: $CONS_w(\Sigma) = min(\mathcal{C}, \preceq_{\mathcal{C},w})$.

Example 10 *We come back to example 3, the following array illustrates the set of consistent subsets of* Σ:

C_i	$\{\phi \in C_i\}$	$Min(\{\phi \notin C_i\}, \preceq_\Sigma)$
C_0	$\emptyset$	$b \wedge c \rightarrow \neg a, d \rightarrow a$
C_1	$b \wedge c \rightarrow \neg a$	$b \rightarrow a, d \rightarrow a$
C_2	$b \rightarrow a$	$b \wedge c \rightarrow \neg a, d \rightarrow a$
C_3	$d \rightarrow a$	$b \wedge c \rightarrow \neg a,$
C_4	$b \rightarrow a, b \wedge c \rightarrow \neg a$	$d \rightarrow a$
C_5	$b \rightarrow a, d \rightarrow a$	$b \wedge c \rightarrow \neg a$
C_6	$b \wedge c \rightarrow \neg a, d \rightarrow a$	$b \rightarrow a$
C_7	$b \rightarrow a, b \wedge c \rightarrow \neg a, d \rightarrow a$	$\emptyset$

According to the definition of $\preceq_{\mathcal{C},w}$, *we obtain the partial pre-order graphically represented by Figure 6.*

Figure 6: Representation of $\preceq_{\mathcal{C}}, w$



The syntactic definition of Bel is:

$$Bel^{se}[\Psi] = \bigvee_{C \in Cons_w(\Sigma)} C.$$

This means that the syntactic inference can be defined as: ϕ is inferred from $\preceq_\Sigma$ iff $\forall C \in CONS_s$, $C \cup \neg\phi$ is inconsistent.

Theorem 2 *Let* Ψ *be the epistemic state, let* $\circ_\triangleright^{sy}$ *and* $\circ_\triangleright^{se}$ *be the syntactic and semantic revision operators stemming from the history of observations, and* $\circ_\pi^{sy}$ *and* $\circ_\pi^{se}$ *be the syntactic and semantic possibilistic revision operators. The following result holds:*

- $Bel^{sy}(\Psi \circ_\triangleright^{sy} \mu) \equiv Bel^{se}(\Psi \circ_\triangleright^{se} \mu)$,
- $Bel^{sy}(\Psi \circ_\pi^{sy} \mu) \equiv Bel^{se}(\Psi \circ_\pi^{se} \mu)$.

We illustrate this theorem with an example:

Example 11 *Let* Ψ *be the epistemic state defined in example 4 where the revision by* $\mu = d$ *leads, to the belief set* $Bel^{se}(\Psi \circ_\triangleright^{se} \mu)$ *such that* $Mod(Bel^{se}(\Psi \circ_\triangleright^{se} \mu)) = \{\omega_9, \omega_{11}, \omega_{13}\}$.

On the syntactic level, we can check that, since $\Sigma \cup \{\mu\} \cup U$ *is consistent, that the preferred elements for* $\preceq_{\mathcal{C},w}$, $Cons_w(\Sigma \circ_\triangleright^{sy} \mu)$ *is simply the subset composed of all elements of* $\Sigma \circ_\triangleright^{sy} \mu$. *Namely we have* $Cons_w(\Sigma_{\circ_\triangleright^{sy} \mu}) = \{\{b \rightarrow a, b \wedge c \rightarrow \neg a, d \rightarrow a, d, (b \rightarrow a) \vee s, (b \wedge c \rightarrow \neg a) \vee d, (d \rightarrow a) \vee d\}\}$ *and hence* $Mod(Cons_w) = \{\omega_9, \omega_{11}, \omega_{13}\}$, *as it is excepted.*

6 Concluding discussion

Since in certain situations an agent faces incomplete information and has to deal with partially ordered information, this paper proposed a semantic representation of an epistemic state by a partial pre-order on interpretations as well as a syntactic representation by a partially ordered belief base. The extension to partial

pre-orders of two revision strategies already defined for total pre-orders are presented, and the equivalence between the representations is shown. We showed that after a certain number of successive revisions the partial pre-order convergences to a total pre-order.

In a future work, we have to investigate the properties of these revision operators and the presented approach could be generalized to the revision of a partial pre-order by a partial pre-order, generalizing the approach proposed in [3]. Moreover, in order to provide reversibility the encoding by polynomials of partial pre-orders on interpretations and partially ordered belief base could be investigated.

Another future work is to develop algorithms for computing the belief set at the syntactical case, and to apply them in geographical information systems where available information is often partially ordered.

7 Acknowledgments

This work was supported by European Community with the REV!GIS project ♯ IST-1999-14189. http://www.cmi.uni-mrs.fr/REVIGIS

References

[1] C. Alchourron, P. Gärdenfors, and D.Makinson. On the Logic of Theory Change: Partial Meet Functions for Contraction and Revision. *Journal of Symbolic Logic*, 50:510–530, 1985.

[2] S. Benferhat, D. Dubois, and O. Papini. A sequential reversible belief revision method based on polynomials. In *Proc. of the sixtenth national conference on artificial intelligence (AAAI1999)*, pages 733–738, 1999.

[3] S. Benferhat, K. Konieczny, O. Papini, and R. P. Perez. Iterated Revision by epistemic states: axioms, semantics and syntax. In *Proc. of the 14 th European Conference on Artificial Intelligence (ECAI'2000)*, pages 13–17, 2000.

[4] C. Cayrol, V. Royer, and C.Saurel. Management of preferences in assumption-based reasoning. In *Proc. of Information Processing and the Management of Uncertainty in Knowledge Based Systems (IPMU'92)*, pages 13–22. Springer, 1992.

[5] A. Darwiche and J. Pearl. On the logic of iterated revision. *Artificial Intelligence*, 89:1–29, 1997.

[6] D. Dubois, J. Lang, and H. Prade. Timed possibilistic logic. *fundamenta informaticae*, XV:211–234, 1992.

[7] D. Dubois and H. Prade. Belief change and possibility theory. In P. G. Eds, editor, *Belief Revision*, pages 142–182. Cambridge University Press, 1992.

[8] D. Dubois and H. Prade. A synthetic view of belief revision with uncertain inputs in the framework of possibility theory. *Int. J. Approx. Reasoning*, 17:295–324, 1997.

[9] P. Gärdenfors. *Knowledge in Flux: Modeling the Dynamics of Epistemic States*. Bradford Books. MIT Press, Cambridge, 1988.

[10] J. Y. Halpern. Defining relative likehood in partially ordered preferential structures. In *Journal of AI Research*, volume 7, pages 1–24, 1997.

[11] J. Y. Halpern. Plausibility measures: A general approach for representing incertainty. In *Proc. of the 17th International Joint Conference on AI (IJCAI 2001)*, pages 1474–1483, 2001.

[12] H. Katsuno and A. Mendelzon. On the difference between updating a knowledge base and revising it. In *Proc. of the 2nd inter. Conf. on Principles of Knowledge Representation and Reasoning*, pages 387–394, 1991.

[13] H. Katsuno and A. Mendelzon. Propositional Knowledge Base Revision and Minimal Change. *Artificial Intelligence*, 52:263–294, 1991.

[14] C. Lafage, J. Lang, and R. Sabbadin. A logic of supporters. In B. Bouchon-Meunier, R. Yager, and A. Zadeh, editors, *Information, uncertainty and fusion*, pages 381–392. Kluwer Academic, 1999.

[15] O. Papini. Iterated revision operations stemming from the history of an agent's observations. *Frontiers of Belief Revision*, page 27, 2001.

[16] W. Spohn. Ordinal conditiona functions: a dynamic theory of epistemic states. *Causation in Decision, Belief Change, and Statistics*, pages 105–134, 1988.

[17] M. A. Williams. Transmutations of Knowledge Systems. In *Proc. of 4th Principle of Knowledge Representation and Reasoning Conference*, pages 619–629, 1994.

Considerations on Compositional Update Operators

James Delgrande and Matthew Suderman
School of Computing Science, Simon Fraser University,
Burnaby, B.C., Canada, V5A 1S6
{jim,mjsuderm}@cs.sfu.ca

Abstract

Two specific operators are proposed for modifying a knowledge base: one for updating a knowledge base with a formula, and a complementary operator that removes a formula. The approach differs from other belief change operators in that the definition of the operators is compositional with respect to the sentence added or removed. The goal is to provide update operators that are intuitive (in that their definitions are based on a recursive decomposition) and that may be reasonably implemented. It proves to be the case that the resulting approach has better complexity characteristics than other operators in some cases. However, while the operators satisfy a core group of the benchmark Katsuno-Mendelzon update postulates, not all of the postulates are satisfied. The full set of Katsuno-Mendelzon postulates are obtained by suitably restricting the syntactic form of the sentence for update. In this fashion we also come up with a hierarchy of update operators with, it turns out, Winslett's *standard semantics* as the weakest approach captured.

1 Introduction

A knowledge base, consisting of a set of facts or assertions is, of course, not a static entity, but rather evolves over time. New information may be added, and old or out-of-date information deleted. A fundamental question concerns how such change should be effected. A major body of research addresses this question via the proposal of various *rationality postulates*, or standards that an adequate change operator may be expected to satisfy. These postulates describe belief change at the *knowledge level*, independent of how beliefs are represented and manipulated. In the *AGM approach* [AGM85, Gär88], standards for revision and contraction functions are given, wherein it is assumed that a knowledge base is receiving information concerning a static domain. Subsequently, Katsuno and Mendelzon [KM92] explored a distinct notion of belief change, with functions for belief *update* and *erasure*, wherein an agent changes its beliefs in response to changes in the environment. See [KM92] for a comparison between revision and update. Various researchers, including [Bor85, Dal88, For89, Sat88, Web86, Win88] have proposed specific change operators. These approaches are formulated in terms of the distance between models of the knowledge base and a sentence for revision or update. In general there has been less work dealing with systems that may be readily implementable (but

see for example [Wil96, DS00]).

In this paper we propose specific update and erasure operators. Our point of departure from previous work is that we develop operators intended to be *compositional* in the sentence μ representing information to be added or removed. For example, if a conjunction $\mu = a \wedge b$ is to be erased from a knowledge base ψ, the intuition is that this can be effected by the erasure of a disjoined with the erasure of b.

Arguably, the results obtained from these operators will be intuitive, since they are based on the recursive decomposition of a formula; hence the (generally abstract) notion of update is here anchored in a more familiar computational setting. Second, the hope is that these operators will be efficiently implementable in some cases, by exploiting restrictions to the syntactic form of the formula. As well, the focus here is on the form of the formula for update; presumably the approach described may be combined with one in which the knowledge base is divided into relevant and irrelevant parts for an update [Par99].

These goals are generally realised. First the approach leads to a straightforward algorithm for implementation. It is efficient in some cases, in particular when the input sentence is restricted to conjunctive normal form or when its size is bounded by a constant. As well the operators have reasonable properties: many of the Katsuno and Mendelzon benchmark properties are satisfied, including those deemed essential by [HR99]. Further we show how the remaining Katsuno and Mendelzon properties may be obtained (by restricting the sentence for update), and we relate a restriction of our approach to one by Winslett.

The next section reviews belief update and erasure, and describes two specific approaches to update. The following section describes our approach after which, in the next section, we give a discussion and analysis. The last section contains concluding remarks; proofs of theorems are given in the full paper.

2 Background

2.1 Belief Update and Erasure

A formula is said to be *complete* just if it implies the truth or falsity of every other formula. In the approach of [KM92], an *update* function $\diamond$ is a function from $L \times L$ to L satisfying the following postulates.

(U1) $\psi \diamond \mu \vdash \mu$.

(U2) If $\psi \vdash \mu$ then $\psi \diamond \mu \equiv \psi$.

(U3) If μ and ψ are satisfiable then so is $\psi \diamond \mu$.

(U4) If $\psi_1 \equiv \psi_2$ and $\mu_1 \equiv \mu_2$ then $\psi_1 \diamond \mu_1 \equiv \psi_2 \diamond \mu_2$.

(U5) $(\psi \diamond \mu) \wedge \phi$ implies $\psi \diamond (\mu \wedge \phi)$.

(U6) If $\psi \diamond \mu_1 \vdash \mu_2$, and $\psi \diamond \mu_2 \vdash \mu_1$ then $\psi \diamond \mu_1 \equiv \psi \diamond \mu_2$.

(U7) If ψ is complete then $(\psi \diamond \mu_1) \wedge (\psi \diamond \mu_2)$ implies $\psi \diamond (\mu_1 \vee \mu_2)$.

(U8) $(\psi_1 \vee \psi_2) \diamond \mu \equiv (\psi_1 \diamond \mu) \vee (\psi_2 \diamond \mu)$

An erasure function $\blacksquare$ is a function from $L \times L$ to L satisfying the following postulates.

(E1) $\psi \vdash \psi \blacksquare \mu$.

(E2) If $\psi \vdash \neg\mu$ then $\psi \blacksquare \mu \equiv \psi$.

(E3) If ψ is satisfiable and $\not\vdash \mu$ then $\psi \blacksquare \mu \not\vdash \mu$.

(E4) If $\psi_1 \equiv \psi_2$ and $\mu_1 \equiv \mu_2$ then $\psi_1 \blacksquare \mu_1 \equiv \psi_2 \blacksquare \mu_2$.

(E5) $(\psi \blacksquare \mu) \wedge \mu \vdash \psi$.

(E8) $(\psi_1 \vee \psi_2) \blacksquare \mu$ is equivalent to $(\psi_1 \blacksquare \mu) \vee (\psi_2 \blacksquare \mu)$

These postulates are not, however, uncontentious. In [HR99] a discussion of the plausibility of the postulates is given, wherein it is concluded that **U2**, **U5**, and **U6** are undesirable, while **U7** is unimportant. This leaves (according to the authors) **U1**, **U3**, **U4**, and **U8** as being desirable.

Update and erasure are often interdefinable by means of the following identities, analogous to the *Levi* and *Harper* identities relating revision and contraction:

$$\psi \diamond \mu \equiv (\psi \bullet \neg\mu) \wedge \mu \quad (1)$$

$$\psi \bullet \mu \equiv \psi \vee (\psi \diamond \neg\mu). \quad (2)$$

The first case asserts that update by μ corresponds to erasing $\neg\mu$ followed by an expansion by μ. The second asserts that erasing μ from ψ corresponds to disjoining ψ with the result of ψ updated by $\neg\mu$.

There have been various specific update (and revision) operators proposed based on the distance between models. We focus on two, both due to Winslett. The first, the *Possible Models Approach (PMA)* [Win88] is a well-known example of an update operator satisfying the Katsuno and Mendelzon update postulates. The second, Winslett's *standard semantics* [Win90] is a weak approach to update that is captured in a variant of ours. We denote these operators by $\diamond_{pma}$ and $\diamond_{ss}$ respectively.

For $\psi \diamond_{pma} \mu$, we have that, for each interpretation I of ψ, $\diamond_{pma}$ selects from the interpretations of μ that are "closest" to I. The update is determined by the set of these closest interpretations. The notion of "closeness" between two interpretations I and J is the Hamming distance, given as follows:

> $diff(I, J)$ = The set of all propositional letters on which I and J differ.

Interpretation J_1 is closer to I than J_2, expressed $J_1 \leq_I J_2$, just if $diff(I, J_1) \subseteq diff(I, J_2)$. The $\leq_I$-minimal set with respect to μ is designated $Incorporate(Mod(\mu), I)$. From this we can specify the PMA update operator:

$$Mod(\psi \diamond_{pma} \mu) = \bigcup_{I \in Mod(\psi)} Incorporate(Mod(\mu), I).$$

The update $\psi \diamond_{ss} \mu$ is defined so that for each model of ψ, those models of μ that retain the truth values of atoms not in μ are chosen. That is, if $atom(\mu)$ is the set of atoms in μ:

$$Mod(\psi \diamond_{ss} \mu) =$$

$$\cup_{I \in Mod(\psi)} \{J \in Mod(\mu) \mid diff(I, J) \subseteq atom(\mu)\}.$$

The following example (from [KM92]) illustrates these update operators. Let $L = \{b, m\}$ and $\psi \equiv (b \wedge \neg m) \vee (\neg b \wedge m)$. Consider updating with μ as b. Under the PMA approach we have the following. The interpretations of ψ are: $I_1 = (\neg b, m)$, $I_2 = (b, \neg m)$; and the interpretations of μ are: $J_1 = (b, m)$, $J_2 = (b, \neg m)$. $diff(I_1, J_1) = \{b\}$ and $diff(I_1, J_2) = \{b, m\}$, hence $J_1 \leq_{I_1, pma} J_2$, and so $Incorporate(Mod(\mu), I_1) = b$. Similarly, $Incorporate(Mod(\mu), I_2) = \{\}$. Hence, $\psi \diamond_{pma} \mu \equiv b$. The same result obtains for $\diamond_{ss}$. For a second example, $(\neg a \wedge \neg b) \diamond_{pma} (a \vee b)$ is equivalent to $a \equiv \neg b$, whereas $\diamond_{ss}$ applied to the same arguments yields $a \vee b$.

3 The Approach

3.1 Preliminaries

The underlying logic will be classical propositional logic. We consider a propositional language L, over a finite set of atoms, or propositional letters, $\mathbf{P} = \{a, b, c, \ldots\}$, and truth-functional connectives $\neg$, $\wedge$, $\vee$, $\supset$, and $\equiv$. *Lits* is the set of literals $\mathbf{P} \cup \{\neg l \mid l \in \mathbf{P}\}$. For a literal l, we use $\bar{l}$ to denote $\neg l$ if $l \in P$ or $l' \in P$ if $\neg l' = l$. Similarly, for a set of literals Γ, we use $\bar{\Gamma}$ to denote the set $\{\bar{l} \mid l \in \Gamma\}$. An *interpretation* of L is a function from $\mathbf{P}$ to $\{T, F\}$.

A *model* of a sentence α is an interpretation that makes α true, according to the usual definition of truth. A model can be equated with its defining set of literals. $Mod(\alpha)$ denotes the set of models of sentence α. For simplicity, and in common with Katsuno and Mendelzon, we assume that knowledge bases are finite – that is, knowledge bases are expressible by a finite language. For interpretation ω we write $\omega \models \alpha$ for α is true in ω. For interpretation ω and set of literals Γ, we write $\omega \downarrow \Gamma$ to denote the set of literals in ω but containing neither l nor $\bar{l}$ for each $l \in \Gamma$. For example, if $L = \{a, b, c\}$ and $\omega = \{a, \neg b, c\}$ then we have $\omega \downarrow \{b, \neg c\} = \{a\}$.

We denote the conjunctive normal form of a sentence μ by $cnf(\mu)$ and the disjunctive normal form of μ by $dnf(\mu)$. For the most part we will work with a specific representation of $cnf(\mu)$ (respectively $dnf(\mu)$), called *normal conjunctive normal form (disjunctive normal form)* and denoted $ncnf(\mu)$ ($ndnf(\mu)$). $ncnf(\mu)$ is obtained by converting μ to negation normal form, and then distributing disjunctions over conjunctions insofar as possible. $ndnf(\mu)$ is obtained in the obvious dual manner. $ncnf(\mu)$, $ndnf(\mu)$, $cnf(\mu)$, and $dnf(\mu)$ will each consist of sets of sets of literals. Members of $cnf(\mu)$ ($ncnf(\mu)$) are implicitly conjoined, and the literals in a member of $cnf(\mu)$ ($ncnf(\mu)$) are implicitly disjoined. The analogous, dual, convention holds for $dnf(\mu)$ and $ndnf(\mu)$. We will also use $\bigvee \Gamma$ to denote the disjunction and $\bigwedge \Gamma$ the conjunction of the sentences in Γ.

3.2 Intuitions

Two operators are defined for changing the state of a knowledge base, one for update and the other for erasure. These operators are defined in a compositional fashion so that, for updating or erasing sentence μ, the operator is defined in terms of the components of μ. For an update $\psi \diamond \mu$, the idea is that each model of ψ is replaced by its closest model(s) in μ [KM92]. Here, the notion of "close" for each model of ψ is determined in part by the syntactic structure of μ. That is, μ is recursively decomposed, and the resulting (base case) literals are used to determine models of the update. Erasure is treated analogously. We begin by considering how this can be effected for an erasure operator, denoted $\blacksquare_c$.

We are given a knowledge base ψ and a sentence μ, and we wish to determine a new knowledge base in which μ is not believed. For the base case, suppose $\mu = l$ is a literal, and we wish to remove l from knowledge base ψ. If the knowledge base already implies $\neg l$ then we need do nothing. If l is believed, then we wish to arrive at a knowledge base in which l is not believed. That is, we want to change the knowledge base only enough so that it does not entail l. We can do this by adding, for each model ω of ψ such that $\omega \models l$ an interpretation $\omega' = (\omega \downarrow \{l\}) \cup \{\bar{l}\}$ to the knowledge base. Thus, we would have both ω' and ω in the resulting knowledge base. The intuition is that for each model ω in which l is true, we add an interpretation ω' exactly like ω, except that $\omega' \models \neg l$.

Next consider removing a conjunction of literals $\mu = l_1 \wedge l_2$ from a knowledge base. A knowledge base in which $l_1 \wedge l_2$ is not believed will of course be one in which at least l_1 is not believed or l_2 is not believed. So (in terms of interpretations) we can take the result of erasing $l_1 \wedge l_2$ to be the union of erasing l_1 and erasing l_2. We can do this in turn by adding, for each $\omega \in Mod(\psi)$ such that $\omega \models l_1 \wedge l_2$, an interpretation $\omega'_1 = (\omega \downarrow \{l_1\}) \cup \{\bar{l_1}\}$ and an interpretation $\omega'_2 = (\omega \downarrow \{l_2\}) \cup \{\bar{l_2}\}$.

Finally, to remove a disjunction of literals $\mu = l_1 \vee l_2$ from a knowledge base we want to add "sufficient" interpretations in which neither l_1 nor l_2 are true. We can accomplish this by adding, for each $\omega \in Mod(\psi)$ such that $\omega \models$

$l_1 \vee l_2$, an interpretation $\omega' = (\omega \downarrow \{l_1, l_2\}) \cup \{\overline{l_1}, \overline{l_2}\}$.

We generalise $\blacksquare_c$ to handle arbitrary propositional sentences by generalising how it handles conjunctions and disjunctions. For conjunctions, we extend the definition to conjunctions of sentences rather than just literals. So to erase a conjunction of sentences, we recursively erase each conjunct and then take the union of the results. For disjunctions, we similarly extend the definition of the operator to handle arbitrary disjunctions.

3.3 Update and Erasure Operators

Based on these intuitions, we define erasure and update operators, $\blacksquare_c$ and $\diamond_c$ respectively. We begin with some preliminary definitions. In the following, $SetL$ sets the value of select literals of interpretation ω, given a set of formulas Γ. This formalises the procedure given in the discussion above.

Definition 1 *For interpretation ω and $\Gamma \subseteq L$, define $SetL(\omega, \Gamma)$ as follows:*

1. *If $\Gamma \subseteq Lits$ then $\Gamma' = \Gamma \setminus \{l, \bar{l} \mid l \in Lits\}$ and $SetL(\omega, \Gamma) = \{(\omega \downarrow \Gamma') \cup \overline{\Gamma'}\}$.*
2. *If $\Gamma = \{\alpha \wedge \beta\} \cup \Gamma'$ then $SetL(\omega, \Gamma) = SetL(\omega, \{\alpha\} \cup \Gamma') \cup SetL(\omega, \{\beta\} \cup \Gamma')$*
3. *If $\Gamma = \{\alpha \vee \beta\} \cup \Gamma'$ then $SetL(\omega, \Gamma) = SetL(\omega, \{\alpha, \beta\} \cup \Gamma')$*
4. *If $\Gamma = \{\neg(\alpha \vee \beta)\} \cup \Gamma'$ then $SetL(\omega, \Gamma) = SetL(\omega, \{\neg\alpha\} \cup \Gamma') \cup SetL(\omega, \{\neg\beta\} \cup \Gamma')$*
5. *If $\Gamma = \{\neg(\alpha \wedge \beta)\} \cup \Gamma'$ then $SetL(\omega, \Gamma) = SetL(\omega, \{\neg\alpha, \neg\beta\} \cup \Gamma')$*
6. *If $\Gamma = \{\neg\neg\alpha\} \cup \Gamma'$ then $SetL(\omega, \Gamma) = SetL(\omega, \{\alpha\} \cup \Gamma')$*

For knowledge base ψ, $New(\psi, \mu)$ is the set of (potentially) new interpretations added to ψ required to erase the sentence μ.

Definition 2

$New(\psi, \mu) = \{\omega' \mid \omega' \in SetL(\omega, \{\mu\}), \omega \in Mod(\psi)\}$.

From this we can define our operator $\blacksquare_c$:

Definition 3 $Mod(\psi \blacksquare_c \mu) = Mod(\psi) \cup New(\psi, \mu)$.

That is, $\psi \blacksquare_c \mu$ is any sentence whose models satisfy the definition. The next section gives an algorithm which directly constructs a formula for $\psi \blacksquare_c \mu$.

The following lemma gives an alternative definition for New in terms of the normal conjunctive normal form of the sentence to erase. The results are straightforward, but are useful for the penultimate section where we consider properties of the approach.

Lemma 1 *For $\psi, \mu \in L$, we have* $New(\psi, \mu) =$

$$\{(\omega \downarrow \Gamma) \cup \overline{\Gamma} \mid \omega \in Mod(\psi) \text{ and } \Gamma \in ncnf(\mu)\}.$$

The next lemma follows from Lemma 1.

Lemma 2 *For $\psi, \mu \in L$, we have*

$Mod(\psi) \cap Mod(\mu) \subseteq New(\psi, \neg\mu) \subseteq Mod(\mu)$.

From Definition 2 and Equation 1, we can define two update operators based on $\blacksquare_c$, called $\diamond_c$ and $\diamond'_c$:

$$Mod(\psi \diamond_c \mu) = New(\psi, \neg\mu) \quad (3)$$
$$Mod(\psi \diamond'_c \mu) = Mod(\psi \blacksquare_c \neg\mu) \cap Mod(\mu) \quad (4)$$

We obtain:

Theorem 3 *For $\psi, \mu \in L$, we have $\psi \diamond_c \mu \equiv \psi \diamond'_c \mu$.*

Hence we need consider just $\diamond_c$. The following theorem is based on Lemma 1. Like that lemma, it gives an alternative definition for $\diamond_c$ based on the normal disjunctive normal form of the update sentence.

Theorem 4 *For* $\psi, \mu \in L$, *we have* $Mod(\psi \diamond_c \mu)$

$= \{(\omega \downarrow \Gamma) \cup \Gamma \mid \omega \in Mod(\psi)$ *and* $\Gamma \in ndnf(\mu)\}$.

4 Properties of the Operators

To start, we consider which of the Katsuno-Mendelzon postulates our operators satisfy.

Theorem 5 $\bullet_c$ *satisfies* **E1**, **E3**, **E5**, **E8**.

The counterexample to **E2** is analogous to the one for **U2** given below. Intuitively, **E2** is not satisfied because $\bullet_c$ may add interpretations not in $Mod(\psi)$ that satisfy $\neg\mu$. Therefore, if $Mod(\psi)$ already contains all interpretations that satisfy $\neg\mu$ then the knowledge base will not change. A counterexample to **E4** is given by $(a \wedge b) \bullet_c ((a \vee \neg b) \wedge \neg b) \equiv (a \wedge b) \vee (\neg a \wedge b) \not\equiv (a \wedge b) \bullet_c \neg b \equiv a \wedge b$.

The postulates that $\diamond_c$ fails to satisfy are directly related to those that $\bullet_c$ fails to satisfy.

Theorem 6 $\diamond_c$ *satisfies* **U1**, **U3**, **U5**, **U7**, **U8**.

For a counterexample to **U2** consider the second example given illustrating Winslett's approaches, where $\psi = (b \wedge \neg m) \vee (\neg b \wedge m)$ and $\mu = b \vee m$. In our approach, the models of the update sentence are $\{b, m\}$, $\{b, \neg m\}$, and $\{\neg b, m\}$. Hence $\psi \diamond_c (b \vee m) \equiv b \vee m$. **U2** would dictate that the result be ψ. The operator $\diamond_c$, on the other hand, includes an additional model. This appears to make some sense because by updating by $b \vee m$, we are really telling the knowledge base that the world has changed so that one of $b \wedge m$ or $b \wedge \neg m$ or $\neg b \wedge m$ is true.

A counterexample for **U4** is provided by $a \diamond_c ((\neg a \wedge b) \vee b) \equiv (\neg a \wedge b) \vee (a \wedge b) \not\equiv a \diamond_c b \equiv a \wedge b$. The reason that **U4** (and **E4**) are not satisfied is that in our compositional approach, parts of a sentence may interact to provide implicit results not explicit in the sentence. Consider $(\neg a \vee b) \wedge (\neg b \vee c)$ for example. Updating by this sentence is effected by updating by the individual components, viz $(\neg a \vee b)$ and $(\neg b \vee c)$. However, implicit in these parts is the fact that $(\neg a \vee c)$ is also true, and the presence of this sentence would affect the result of the update. We consider this behaviour further below.

A counterexample for **U6** is given by $(a \wedge b) \diamond_c ((a \wedge c) \vee (\neg a \wedge b \wedge c)) \equiv (a \wedge b \wedge c) \vee (\neg a \wedge b \wedge c) \not\equiv (a \wedge b) \diamond_c ((b \wedge c) \vee (a \wedge \neg b \wedge c)) \equiv (a \wedge b \wedge c) \vee (a \wedge \neg b \wedge c)$. **U2** and **U6** are not satisfied for exactly the same reason that $\bullet_c$ does not satisfy **E2**; that is, $\psi \diamond_c \mu$ includes in some cases interpretations not in $Mod(\psi)$ that satisfy μ. We pursue this behaviour further in the next section, where we use this as a point of contrast with Winslett's approach. Similarly, **U4** does not hold for the same reason that **E4** doesn't hold.

Despite failing to satisfy some postulates, $\bullet_c$ and $\diamond_c$ do exhibit a nice property that operators appearing in the literature and satisfying the Katsuno and Mendelzon postulates fail to satisfy. Let $\mu \wedge \phi$ be a satisfiable propositional sentence. The following are straightforward.

Proposition 1 $\psi \bullet_c (\mu \wedge \phi) \equiv (\psi \bullet_c \mu) \vee (\psi \bullet_c \phi)$

Proposition 2 $\psi \diamond_c (\mu \vee \phi) \equiv (\psi \diamond_c \mu) \vee (\psi \diamond_c \phi)$

Our update operator also satisfies those postulates deemed desirable by [HR99], with the exception of **U4**. As discussed above, **U4** is not satisfied due to the interaction of parts of a sentence. It would seem that if we could "compile out" the implicit information in a sentence then we would obtain substitution of equivalents, as expressed in **E4** and **U4**. One way to satisfy **E4** and **U4** then is to redefine $\diamond_c$ and $\bullet_c$ so that we first "compile out" implicit information. We do this by defining operators that consider the set of *prime implicants* of a

sentence.[1] We call these modified operators $\bullet_c^{pi}$ and $\diamond_c^{pi}$. Let $PI(\mu)$ be the set of prime implicants of μ.

Definition 4

$\psi \bullet_c^{pi} \mu = \psi \bullet_c PI(\mu)$

$\psi \diamond_c^{pi} \mu = New(\psi, PI(\neg\mu))$

Although $\diamond_c^{pi}$ satisfies **U4** (and $\bullet_c^{pi}$ **E4**), we lose **U7** with $\diamond_c^{pi}$. A counter-example for **U7** is given by $\mu_1 = (a \wedge d) \vee (\neg c \wedge d)$ and $\mu_2 = (\neg a \wedge d) \vee (\neg c \wedge d)$ and $\psi = a \wedge b \wedge c \wedge d$.

We can further pursue this direction as follows. For μ let $ModL(\mu)$ be the models of μ, over the language of μ, expressed in disjunctive normal form. For example $ModL((a \vee b) \wedge c)$ would be $(a \wedge b \wedge c) \vee (a \wedge \neg b \wedge c) \vee (\neg a \wedge b \wedge c)$.

Definition 5 $\psi \diamond_c^{ss} \mu = New(\psi, ModL(\neg\mu))$

We obtain:

Proposition 3 $\psi \diamond_c^{ss} \mu = \psi \diamond_{ss} \mu$.

Thus in this case we capture the basic Winslett approach [Win90].[2] Hence we obtain a hierarchy of operators, based on the extent to which information in μ is made explicit.

Finally, **E2** and **U2** can also be satisfied by adding a condition to $\diamond_c$ and $\bullet_c$. **E2** states that if we wish to erase μ from ψ and $\psi \vdash \neg\mu$, then ψ is unchanged. Similarly, **U2** states that if we wish to update by μ and $\psi \vdash \mu$, then ψ is unchanged. The corresponding modified operators, $\bullet_c'$ and $\diamond_c'$, are defined as follows:[3]

Definition 6

$$\psi \bullet_c' \mu = \begin{cases} \psi & \text{if } \psi \vdash \neg\mu \\ \psi \bullet_c \mu & \text{otherwise} \end{cases}$$

$$\psi' \diamond_c \mu = \begin{cases} \psi & \text{if } \psi \vdash \mu \\ \psi \diamond_c \mu & \text{otherwise} \end{cases}$$

[1] A consistent set of literals Γ is a *prime implicant* of μ iff: $\Gamma \vdash \mu$ and for $\Gamma' \subset \Gamma$ we have $\Gamma' \nvdash \mu$.

[2] More accurately, we capture a syntax-independent variant of the Winslett approach [HR99].

[3] [Bor85] employs a similar definition with respect to a revision operator.

Last, we have already noted that the base approach given by $\diamond_c$ is distinct from the Winslett approaches. It is also distinct from all other approaches that we are aware of and that have appeared in the literature. The full paper contains examples and more details.

4.1 Algorithms and Complexity

In this section we provide algorithms for our operators. We also analyse the complexity of these algorithms under a variety of assumptions. Specifically, we analyse the complexity of the algorithms when applied to any general propositional sentences, any sentences in conjunctive normal form, any sentences in disjunctive normal form, and any sentences whose sizes are bounded by some specified constant.

In the following algorithms defined for $\bullet_c$, $\diamond_c$ and New, let $\psi, \mu \in L$:

Algorithm $EraseAlg(\psi, \mu)$
1. **return** $\psi \vee NewAlg(\psi, \mu)$

Algorithm $UpdateAlg(\psi, \mu)$
1. **return** $NewAlg(\psi, \neg\mu)$

Algorithm $NewAlg(\psi, \mu)$
1. $\psi' \leftarrow False$
2. **for each** clause $\Gamma \in ncnf(\mu)$
3. $\quad \psi' \leftarrow \psi' \vee \left(FlipAlg(\psi, \Gamma) \bigwedge \overline{\Gamma}\right)$
4. **return** ψ'

Algorithm $FlipAlg(\psi, \Gamma)$
1. $\psi'' \leftarrow nnf(\psi)$
2. **for each** $l \in \Gamma$
3. $\quad$ replace each l in ψ'' by $\bar{l}$
4. **return** ψ''

Note that $nnf(\psi)$ is just the negation normal form of ψ, and that $FlipAlg$ is not meant to

mirror the behaviour of *SetL*. *FlipAlg* modifies the composition of the input knowledge base by modifying its negation normal form.

The following theorems state that these algorithms are complete and sound with respect to the operators.

Lemma 7 *For $\psi, \psi' \in L$ and $\Gamma \subseteq Lits$ we have*

1. $FlipAlg(\psi \wedge \psi', \Gamma) = FlipAlg(\psi, \Gamma) \wedge FlipAlg(\psi', \Gamma)$
2. $FlipAlg(\psi \vee \psi', \Gamma) = FlipAlg(\psi, \Gamma) \vee FlipAlg(\psi', \Gamma)$
3. $FlipAlg(\neg(\psi \vee \psi'), \Gamma) = FlipAlg(\neg\psi, \Gamma) \wedge FlipAlg(\neg\psi', \Gamma)$
4. $FlipAlg(\neg(\psi \wedge \psi'), \Gamma) = FlipAlg(\neg\psi, \Gamma) \vee FlipAlg(\neg\psi', \Gamma)$
5. $FlipAlg(\neg\neg\psi, \Gamma) = FlipAlg(\psi, \Gamma)$

Corollary 8 *For $\psi \in L$ and $\Gamma \subseteq Lits$ we have*

$$ndnf(FlipAlg(\psi, \Gamma)) = FlipAlg(ndnf(\psi), \Gamma).$$

Lemma 9 *For $\psi, \mu \in L$, $ndnf(NewAlg(\psi, \mu))$ is equivalent to*

$$\bigvee \left\{ \bigwedge (\Gamma' \downarrow \Gamma) \cup \overline{\Gamma} \mid \Gamma \in ncnf(\mu) \text{ and } \Gamma' \in ndnf(\psi) \right\}$$

Theorem 10

$$New(\psi, \mu) = Mod(NewAlg(\psi, \mu)).$$

Corollary 11

$$\psi \bullet_c \mu \equiv EraseAlg(\psi, \mu)$$

$$\psi \diamond_c \mu \equiv UpdateAlg(\psi, \mu)$$

We also give an alternative definition for *UpdateAlg*.

Theorem 12 *For $\psi, \mu \in L$, $ndnf(UpdateAlg(\psi, \mu))$ is equivalent to*

$$\bigvee \left\{ \bigwedge (\Gamma' \downarrow \Gamma) \cup \Gamma \mid \Gamma \in ndnf(\mu) \text{ and } \Gamma' \in ndnf(\psi) \right\}$$

We now give the complexity of the algorithms. Let $\psi, \mu \in L$ and let $|\delta|$ be the size of $\delta \in L$.

Theorem 13 *The complexity of evaluating $EraseAlg(\psi, \mu)$ is:*

1. $O(|\psi| \times 2^{|\mu|})$ *for* $\mu \in L$;
2. $O(|\psi| \times |\mu|)$ *for* μ *in cnf; and*
3. $O(|\psi|)$ *for* $|\mu| < c$ *for some constant* c.

Theorem 14 *The complexity of evaluating $UpdateAlg(\psi, \mu)$ is:*

1. $O(|\psi| \times 2^{|\mu|})$ *for* $\mu \in L$;
2. $O(|\psi| \times |\mu|)$ *for* μ *in dnf; and*
3. $O(|\psi|)$ *for* $|\mu| < c$ *for some constant* c.

These theorems show that, like most other model-based change operators, the complexity of the algorithm is in the size of the update sentence [EG92]. Also, because of how *SetL* is defined, we see that *EraseAlg* is quite efficient when the sentence to remove is in conjunctive normal form and *UpdateAlg* is quite efficient when the sentence to add is in disjunctive normal form. According to [EG92] the major model-based operators are at least co-NP-complete when the update sentences are in either of these two forms. Since it is reasonable for practical knowledge base systems to put limits on the size or form of the update sentences, these are positive results.

5 Conclusion

We have presented belief change operators for updating a knowledge base and for erasing a formula from a knowledge base. The definition of the operators is compositional with respect to the sentence added or removed; the intent is to provide operators with transparent definitions, based on the structure of the formula for belief change. As a result we lose some of the

standard postulates for update and erasure, although we satisfy a core group of the standard postulate set. We achieve full irrelevance of syntax if the sentence for update is replaced by the disjunction of its prime implicants and conversely the sentence for removal is replaced by the conjunction of its prime implicants. We argue that the approach is interesting first, because it is founded on differing intuitions than other operators, in that it is based on a decomposition of the formula, and second, that it allows a straightforward, and in some cases efficient, implementation. While distinct from previous update operators that have appeared in the literature, we capture the basic Winslett approach to update in a restriction of our approach

An open question concerns how the approach may be combined with one that is designed to exploit the structure of the knowledge base (such as discussed in [Par99]). A second, technical question that is not fully explored concerns the behaviour of $\diamond_c$ as an erasure operator. For example, let $\psi = (a \vee b) \wedge (\neg a \vee \neg b)$. Then, while $\psi \vdash a \vee b$, we get that $\psi \diamond_c (a \vee b) \equiv a \vee b$. Thus, in updating the knowledge base with a formula already implied by the knowledge base, we have actually removed information. This, as argued earlier, is not unreasonable if one considers that an update (in contrast to a revision) by $a \vee b$ asserts that the world has changed so that one of $\{a, b\}$, $\{\neg a, b\}$, $\{a, \neg b\}$ is now true. In general, $\diamond_c$ appears to behave as an erasure operator when it is used to add disjunctions of formulas already in the knowledge base.

References

[AGM85] C.E. Alchourrón, P. Gärdenfors, and D. Makinson. On the logic of theory change. functions. *Journal of Symbolic Logic*, 50(2):510–530, 1985.

[Bor85] A. Borgida. Language features for flexible handling of exceptions in information systems. *ACM Transactions on Database Systems*, 10, 1985.

[Dal88] M. Dalal. Investigations into theory of knowledge base revision. In *Proc. AAAI*, pages 449–479, St. Paul, Minnesota, 1988.

[DS00] J.P. Delgrande and T. Schaub. A consistency-based model for belief change: Preliminary report. In *Proc. AAAI*, pages 392–398, Austin, TX, 2000.

[EG92] T. Eiter and G. Gottlob. On the complexity of propositional knowledge base revision, updates, and counterfactuals. *Artificial Intelligence*, 57(2-3):227–270, 1992.

[For89] K.D. Forbus. Introducing actions into qualitative simulation. In *Proc. IJCAI*, pages 1273–1278, 1989.

[Gär88] P. Gärdenfors. *Knowledge in Flux: Modelling the Dynamics of Epistemic States.* The MIT Press, Cambridge, MA, 1988.

[HR99] A. Herzig and O. Rifi. Propositional belief update and minimal change. *Artificial Intelligence*, 115(1):107–138, 1999.

[KM92] H. Katsuno and A. Mendelzon. On the difference between updating a knowledge base and revising it. In P. Gärdenfors, editor, *Belief Revision*, pages 183–203. Cambridge University Press, 1992.

[Par99] R. Parikh. Beliefs, belief revision, and splitting languages. In L.S. Moss, J. Ginzburg, and M. de Rijke, editors, *Logic, Language, and*

Computation, Vol II, pages 266–278. CSLI Publications, 1999.

[Sat88] K. Satoh. Nonmonotonic reasoning by minimal belief revision. In *Proceedings of the International Conference on Fifth Generation Computer Systems*, pages 455–462, Tokyo, 1988.

[Web86] A. Weber. Updating propositional formulas. *Proc. First Conference on Expert Database Systems*, pages 487–500, 1986.

[Wil96] M.-A. Williams. Towards a practical approach to belief revision: Reason-based change. In L. Aiello, J. Doyle, and S. Shapiro, editors, *Proc. KR*, pages 412–421, Cambridge, MA, November 1996.

[Win88] M. Winslett. Reasoning about action using a possible models approach. In *Proc. AAAI*, pages 89–93, St. Paul, Minnesota, 1988.

[Win90] M. Winslett. *Updating Logical Databases*. Cambridge University Press, 1990.

Repairing Databases with Annotated Predicate Logic

Pablo Barceló
P. Universidad Católica de Chile
Depto. Ciencia de Computación
Santiago, Chile.
pbarcelo@ing.puc.cl

Leopoldo Bertossi
Carleton University
School of Computer Science
Ottawa, Canada.
bertossi@scs.carleton.ca

Abstract

Consistent answers from a relational database that violates a given set of integrity constraints are characterized [Arenas et al. 1999] as ordinary answers that can be obtained from *every* repaired version of the database. In this paper we address the problem of specifying the repairs of a database as the minimal models of a theory written in *Annotated Predicate Logic* [Kifer et al. 1992a]. The specification is then transformed into a disjunctive logic program with annotation arguments and a stable model semantics. From the program, consistent answers to first order queries are obtained.

1 Introduction

Integrity constraints (ICs) are important in the design and use of a relational database. They embody the semantics of the application domain and help maintain the correspondence between that application domain and its model provided by the database. Nevertheless, it is not strange for a database instance to become inconsistent with respect to a given, expected set of ICs. This could happen due to different factors, being one of them the integration of several data sources. The integration of consistent databases may easily lead to an inconsistent integrated database.

An important problem in databases consists in retrieving answers to queries that are "consistent" with the given ICs, even when the database as a whole does not satisfy those ICs. Very likely "most" of the data is still consistent. The notion of consistent answers to a first order (FO) query was defined in [Arenas et al. 1999], where also a computational mechanism for obtaining them was presented. Intuitively speaking, an ground tuple $\bar{t}$ is a consistent answer to a first order query $Q(\bar{x})$ in a, possibly inconsistent, relational database instance DB if it is an (ordinary) answer to $Q(\bar{x})$ in every minimal repair of DB, that is in every database instance over the same schema that differs from DB by a minimal (under set inclusion) set of inserted or deleted tuples.

That mechanism presented in [Arenas et al. 1999] has some limitations in terms of the ICs and queries that can be handled. In [Arenas et al. 2000b], a more general methodology based on logic programs with an stable model semantics was introduced. More general queries could be considered, but ICs were restricted to be "binary", i.e. universal with at most two database literals (plus built-in formulas).

For consistent query answering we need to deal with *all* the repairs of a database. In consequence, a natural approach consists in providing a manageable logical specification of the class of database repairs. The specification must include information about (from) the database and the information contained in the ICs. Since these two pieces of information are mutually inconsistent, we need a logic that does not collapse in the presence of contradictions. A logic like *Annotated Predicate Logic* (*APC*) [Kifer et al. 1992a], for which a classically inconsistent set of premises can still have a model, is a natural candidate.

In [Arenas et al. 2000a], a new declarative semantic framework was introduced for studying the problem of query answering in databases that are inconsistent with integrity constraints. This was done by embedding both the database instance and the integrity constraints into a single theory written in *APC*,

with an appropriate non classical truth-values lattice *Latt*. It was shown that, for universal ICs, there is a one to one correspondence between some minimal models of the annotated theory and the repairs of the inconsistent database. In this way, a logical specification of the database repairs was achieved. The annotated theory was used to obtain some algorithms for obtaining consistent answers to some simple first order queries.

This paper extends the results presented in [Arenas et al. 2000a] in several ways. First, in section 3, we show how to annotate and integrate referential ICs (that contain existential quantifiers) in addition to universal ICs into the annotated theory. The correspondence between minimal models of the theory and the database repairs is established. Next, in section 4, we show how to annotate queries and the formulation of the problem of consistent query answering as a problem of non-monotonic (minimal) entailment from the annotated theory. Then, in section 5.1, on the basis of the generated annotated theory, disjunctive logic programs with annotation arguments are derived in such a way that they specify the database repairs. After that, in section 5.2, we show how to use those programs to obtain consistent answers to first order queries. In section 5.3 the logic programs are transformed into classical disjunctive normal programs with a stable model semantics; and the *coherent* stable models become the database repairs. In section 2 we present the basic framework, and in section 6 we draw some conclusions, mention ongoing work, and consider related work.

The methodology presented here works for arbitrary first order queries and arbitrary universal ICs, what considerable extends the cases that could be handled in [Arenas et al. 1999, Arenas et al. 2000b, Arenas et al. 2000a].

2 Preliminaries

2.1 Database repairs and consistent answers

In the context of relational databases, we will consider a fixed relational schema $\Sigma = (D, P \cup B)$ that determines a first order language. It consists of a fixed, possibly infinite, database domain $D = \{c_1, c_2, ...\}$, a fixed set of database predicates $P = \{p_1, \ldots, p_n\}$, and a fixed set of built-in predicates $B = \{e_1, \ldots, e_m\}$.

A database instance over Σ is a finite collection DB of facts of the form $p(c_1, ..., c_n)$, where p is a predicate in P and $c_1, ..., c_n$ are constants in D. Built-in predicates have a fixed and same extension in every database instance, not subject to changes.

An *integrity constraint* (IC) is an implicitly quantified clause of the form

$$q_1(\bar{t}_1) \vee \cdots \vee q_n(\bar{t}_n) \vee \neg p_1(\bar{s}_1) \vee \cdots \vee \neg p_m(\bar{s}_m) \quad (1)$$

in the *FO* language of Σ, where each p_i, q_j is a predicate in $P \cup B$ and the $\bar{t}_i, \bar{s}_j$ are tuples containing constants and variables. We assume we have a fixed set IC of ICs.

We will assume that DB and IC, separately, are consistent theories. Nevertheless, it may be the case that $DB \cup IC$ is inconsistent. Equivalently, if we associate to DB a first order structure, also denoted with DB, in the natural way, i.e. by applying the closed world assumption that makes false any ground atom not explicitly appearing in the set of atoms DB, it may happen that DB, as a structure, does not satisfy the IC. We denote with $DB \models_\Sigma IC$ the fact that the database satisfies IC. In this case we say that DB is consistent wrt IC; otherwise we say DB is inconsistent.

As in [Arenas et al. 1999], we define the *distance* between two database instances DB_1 and DB_2 as their symmetric difference $\Delta(DB_1, DB_2) = (DB_1 - DB_2) \cup (DB_2 - DB_1)$.

Now, given a database instance DB, possibly inconsistent wrt IC, we say that the instance DB' is a *repair* of DB iff $DB' \models_\Sigma IC$ and $\Delta(DB, DB')$ is minimal under set inclusion in the class of instances that satisfy IC and conform to schema Σ [Arenas et al. 1999].

Example 1. Consider the relational schema $Book(author, name, publYear)$, a database instance $DB = \{Book(kafka, metamorph, 1915), Book(kafka, metamorph, 1919)\}$; and the functional dependency $FD : author, name \rightarrow publYear$, that can be expressed by $IC : \neg Book(x, y, z) \vee \neg Book(x, y, w) \vee z = w$. Instance DB is inconsistent with respect to IC. The original instance has two possible repairs: $DB_1 = \{Book(kafka, metamorph, 1915)\}$, and $DB_2 = \{Book(kafka, metamorph, 1919)\}$. □

Let DB be a database instance, possibly not satisfying a set IC of integrity constraints. Given a query $Q(\bar{x})$ to DB, we say that a tuple of constants $\bar{t}$ is a *consistent answer* to $Q(\bar{x})$ in DB

[Arenas et al. 1999], denoted $DB \models_c Q(\bar{t})$, if for every repair DB' of DB, $DB' \models_\Sigma Q(\bar{t})$. If Q is a closed formula, *i.e.* a sentence, then *true* is a *consistent answer* to Q, denoted $DB \models_c Q$, if for every repair DB' of DB, $DB' \models_\Sigma Q$.

Example 2. (example 1 continued) The query Q_1 : $Book(kafka, metamorph, 1915)$ does not have *true* as a consistent answer, because it is not true in every repair. Query $Q_2(y)$: $\exists x \exists z Book(x, y, z)$ has $y = metamorph$ as a consistent answer. Query $Q_3(x)$: $\exists z Book(x, metamorph, z)$ has $x = kafka$ as a consistent answer. □

2.2 Annotating DBs and ICs

Annotated Predicate Calculus was introduced in [Kifer et al. 1992a] and also studied in [Blair et al. 1989] and [Kifer et al. 1992b]. It constitutes a non classical logic, where classically inconsistent information does not unravel logical inference, reasoning about causes of inconsistency becomes possible, making one of its goals to study the differences in the contribution to the inconsistency made by the different literals in a theory, what is related to the the problem of consistent query answers.

The syntax of *APC* is similar to that of classical logic, except for the fact that the atoms are annotated with values drawn from a *truth-values lattice.* The lattice *Latt* we will use throughout this paper is shown in Figure 1, first introduced in [Arenas et al. 2000a].

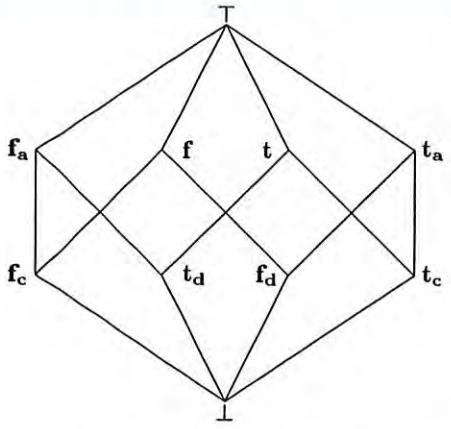


Figure 1: *Latt* with *constraints values*, *database values* and *advisory values*

Intuitively, we can think of values $\mathbf{t_c}$ and $\mathbf{f_c}$ as specifying what is needed for constraint satisfaction. The values $\mathbf{t_d}$ and $\mathbf{f_d}$ represent the truth values according to the original database. Finally, $\mathbf{t_a}$ and $\mathbf{f_a}$ are considered *advisory* truth values. These are intended to solve conflicts between the original database and what is needed for the satisfaction of the integrity constraints. Notice that $lub(\mathbf{t_d}, \mathbf{f_c}) = \mathbf{f_a}$ and $lub(\mathbf{f_d}, \mathbf{t_c}) = \mathbf{t_a}$. The intuition behind is that, in case of a conflict between the constraints and the database, we should obey the constraints, because the database instance only can be changed to restore consistency. This lack of symmetry between data and ICs is captured by the lattice. Advisory value $\mathbf{t_a}$ is an indication that the atom annotated with it must be inserted into the DB; and deleted from the DB when annotated with $\mathbf{f_a}$.

Herbrand interpretations are now sets of annotated ground atoms. The notion of formula satisfaction in an *Herbrand interpretation* I is defined classically, except for atomic formulas p, where we say that $I \models p:\mathbf{s}$, with $\mathbf{s} \in Latt$, iff for some $\mathbf{s}'$ such that $\mathbf{s} \leq \mathbf{s}'$ we have that $p:\mathbf{s}' \in I$ [Kifer et al. 1992a].

Given an *APC* theory $\mathcal{T}$, we say that an Herbrand interpretation I is a Δ-minimal model of $\mathcal{T}$, with $\Delta = \{\mathbf{t_a}, \mathbf{f_a}\}$, if I is a model of $\mathcal{T}$ and no other model of $\mathcal{T}$ has a proper subset of atoms annotated with elements in Δ, *i.e.* the set of atoms annotated with $\mathbf{t_a}$ or $\mathbf{f_a}$ in I is minimal under set inclusion.

Given a database instance DB and a set of integrity constraints IC of the form (1), an embedding $\mathcal{T}(DB, IC)$ of DB and IC into a new *APC* theory can be defined [Arenas et al. 2000a] in order to restore consistency using the annotations in the lattice. It was shown in [Arenas et al. 2000a] that there is a one-to-one correspondence between the Δ-minimal models (we will simply say "minimal" in the rest of the paper) of theory $\mathcal{T}(DB, IC)$ and the repairs of the original database instance. Repairs can be obtained from minimal models as follows:

Definition 1. Given a minimal model $\mathcal{M}$ of $\mathcal{T}(DB, IC)$, the corresponding DB instance is defined by $DB_{\mathcal{M}} = \{p(\bar{a}) \mid \mathcal{M} \models p(\bar{a}):\mathbf{t} \vee p(\bar{a}):\mathbf{t_a}\}$. □

Example 3. (example 1 cont.) The embedding $\mathcal{T}(DB)$ of DB into *APC* is given by the following formulas:

1. $Book(kafka, metamorph, 1915):\mathbf{t_d}$
   $Book(kafka, metamorph, 1919):\mathbf{t_d}$.

2. Predicate closure axioms:

   $((x = kafka)\text{:}\mathbf{t_d} \wedge (y = metamorph)\text{:}\mathbf{t_d} \wedge (z = 1915)\text{:}\mathbf{t_d}) \vee$
   $((x = kafka)\text{:}\mathbf{t_d} \wedge (y = metamorph)\text{:}\mathbf{t_d} \wedge (z = 1919)\text{:}\mathbf{t_d}) \vee$
   $Book(x, y, z)\text{:}\mathbf{f_d}$.

   Every ground atom that is not in DB is (possibly implicitly) annotated with $\mathbf{f_d}$.

The embedding $\mathcal{T}(IC)$ of IC into APC is given by:

3. $Book(x, y, z)\text{:}\mathbf{f_c} \vee Book(x, y, w)\text{:}\mathbf{f_c} \vee (z = w)\text{:}\mathbf{t_c}$.

4. $Book(x, y, z)\text{:}\mathbf{f_c} \vee Book(x, y, z)\text{:}\mathbf{t_c}$
   $\neg Book(x, y, z)\text{:}\mathbf{f_c} \vee \neg Book(x, y, z)\text{:}\mathbf{t_c}$.[1]

   These formulas specify that every fact must have one and just one constraint value.

Furthermore

5. For every true *built-in* atom φ we include φ:**t** in $\mathcal{T}(\mathcal{B})$, and φ:**f** for every false built-in atom, e.g. $(1915 = 1915)$:**t**, but $(1915 = 1919)$:**f**.

The Δ-minimal models of $\mathcal{T}(DB, IC) = \mathcal{T}(DB) \cup \mathcal{T}(IC) \cup \mathcal{T}(\mathcal{B})$ are:

$\mathcal{M}_1 = \{Book(kafka, metamorph, 1915)\text{:}\mathbf{t},$
$Book(kafka, metamorph, 1919)\text{:}\mathbf{f_a}\},$

$\mathcal{M}_2 = \{Book(kafka, metamorph, 1915)\text{:}\mathbf{f_a},$
$Book(kafka, metamorph, 1919)\text{:}\mathbf{t}\},$

plus annotated false DB atoms and built-ins in both cases . The correspondent database instances, $DB_{\mathcal{M}_1}, DB_{\mathcal{M}_2}$ are the repairs of DB shown in example 1. □

From the definition of the lattice and the fact that no atom from the database is annotated with both $\mathbf{t_d}$ and $\mathbf{f_d}$, it is possible to show that, in the minimal models of the annotated theory, a DB atom may get the annotations either **t** or $\mathbf{f_a}$ if the atom was annotated with $\mathbf{t_d}$, and either **f** or $\mathbf{t_a}$ if the atom was annotated with $\mathbf{f_d}$. In the transition from the annotated theory to its minimal models, the annotations $\mathbf{t_d}, \mathbf{f_d}$ "disappear", as we wished the atoms to be annotated in the highest possible layer in the lattice, except for $\top$ if possible. Actually, in the minimal models $\top$ can always be avoided.

[1] Since only atomic formulas are annotated, the non atomic formula $\neg p(\bar{x})$:s is to be read as $\neg(p(\bar{x})$:s). We will omit the parenthesis though.

3 Annotating Referential ICs

Referential integrity constraints (RICs) like

$$p(\bar{x}) \rightarrow \exists y q(\bar{x}', y), \tag{2}$$

where the variables in $\bar{x}'$ are a subset of the variables in $\bar{x}$, cannot be expressed as an equivalent clause of the form (1). RICs are important and common in databases. For that reason, we need to extend our embedding methodology. Actually, we embed (2) into APC by means of

$$p(\bar{x})\text{:}\mathbf{f_c} \vee \exists y(q(\bar{x}', y)\text{:}\mathbf{t_c}). \tag{3}$$

Now we allow the given set of ICs to contain, in addition to ICs of the form (1), RICs like (2). The one-to-one correspondence between minimal models of the new theory $\mathcal{T}(DB, IC)$ and the repairs of DB still holds. Most important for us is to obtain repairs from minimal models.

Proposition 1. Let $\mathcal{M}$ be a model of $\mathcal{T}(DB, IC)$. If $\mathcal{M}$ is minimal and $DB_{\mathcal{M}}$ is finite, then $DB_{\mathcal{M}}$ is a repair of DB with respect to IC. □

Example 4. Consider the relational schema of Example 1 extended with table $Author(name, citizenship)$. Now, IC also contains the RIC: $Book(x, y, z) \rightarrow \exists w Author(x, w)$, expressing that every writer of a book in the database instance must be registered as an author. The theory $\mathcal{T}(IC)$ now also contains:

$$Book(x, y, z)\text{:}\mathbf{f_c} \vee \exists w(Author(x, w)\text{:}\mathbf{t_c}),$$
$$Author(x, w)\text{:}\mathbf{f_c} \vee Author(x, w)\text{:}\mathbf{t_c},$$
$$\neg Author(x, w)\text{:}\mathbf{f_c} \vee \neg Author(x, w)\text{:}\mathbf{t_c}.$$

We might also have the functional dependency $FD : name \rightarrow citizenship$, that in conjunction with the RIC, produces a foreign key constraint. The database instance $\{Book(neruda, 20\,lovepoems, 1924)\}$ is inconsistent wrt the given RIC. If we have the following subdomain $D(Author.citizenship) = \{chilean, canadian\}$ for the attribute "citizenship", we obtain the following database theory:

$\mathcal{T}(DB) = \{Book(neruda, 20\,lovepoems, 1924)\text{:}\mathbf{t_d},$
$Author(neruda, chilean)\text{:}\mathbf{f_d},$
$Author(neruda, canadian)\text{:}\mathbf{f_d}, \ldots\}.$

The minimal models of $\mathcal{T}(DB, IC)$ are:

$\mathcal{M}_1 = \{Book(neruda, 20\,lovepoems, 1924)\text{:}\mathbf{f_a},$
$Author(neruda, chilean)\text{:}\mathbf{f},$

$$\mathcal{M}_2 = \{Author(neruda, canadian):\mathbf{f}, \dots\}$$
$$\mathcal{M}_2 = \{Book(neruda, 20lovepoems, 1924):\mathbf{t},\ Author(neruda, chilean):\mathbf{t_a},\ Author(neruda, canadian):\mathbf{f}, \dots\}$$
$$\mathcal{M}_3 = \{Book(neruda, 20lovepoems, 1924):\mathbf{t},\ Author(neruda, chilean):\mathbf{f},\ Author(neruda, canadian):\mathbf{t_a}, \dots\}.$$

We obtain $DB_{\mathcal{M}_1} = \emptyset$, $DB_{\mathcal{M}_2} = \{Book(neruda, 20lovepoems, 1924), Author(neruda, chilean)\}$ and $DB_{\mathcal{M}_3}$ similar to $DB_{\mathcal{M}_2}$, but with a Canadian Neruda. According to proposition 1, these are repairs of the original database instance, actually the only ones. □

As in [Arenas et al. 2000a], it can be proved that when the original instance is consistent, then it is its only repair and it corresponds to a unique minimal model of the APC theory.

4 Annotation of Queries

According to proposition 1, a ground tuple $\bar{t}$ is a consistent answer to a FO query $Q(\bar{x})$ iff $Q(\bar{t})$ is true of every minimal model of $\mathcal{T}(DB, IC)$. However, if we want to pose the query directly to the theory, it is necessary to reformulate it as an annotated formula.

Definition 2. Given a FO query $Q(\bar{x})$ in language Σ, we denote by $Q^{an}(\bar{x})$ the annotated formula obtained from Q by simultaneously replacing, for $p \in P$, the negative literal $\neg p(\bar{s})$ by the APC formula $p(\bar{s}):\mathbf{f} \vee p(\bar{s}):\mathbf{f_a}$, and the positive literal $p(\bar{s})$ by the APC formula $p(\bar{s}):\mathbf{t} \vee p(\bar{s}):\mathbf{t_a}$. For $p \in B$, the literal $p(\bar{s})$ is replaced by the APC formula $p(\bar{s}):\mathbf{t}$. □

According to this definition, logically equivalent versions of a query could have different annotated versions, but it can be shown (proposition 2), that they retrieve the same consistent answers.

Example 5. (example 1 cont.) If we want the consistent answers to the query $Q(x)$: $\neg\exists y\exists z\exists w\exists t(Book(x, y, z) \wedge Book(x, w, t) \wedge y \neq w)$, asking for those authors that have at most one book in DB, we generate the annotated query $Q^{an}(\bar{x})$: $\neg\exists y\exists z\exists w\exists t((Book(x, y, z) : \mathbf{t} \vee Book(x, y, z) : \mathbf{t_a}) \wedge (Book(x, w, t) : \mathbf{t} \vee Book(x, w, t) : \mathbf{t_a}) \wedge (y \neq w) : \mathbf{t})$, to be posed to the annotated theory with its minimal model semantics. □

Definition 3. If φ is a sentence in the language of $\mathcal{T}(DB, IC)$, we say that $\mathcal{T}(DB, IC)$ Δ-minimally entails φ, written $\mathcal{T}(DB, IC) \models_\Delta \varphi$, iff every Δ-minimal model $\mathcal{M}$ of $\mathcal{T}(DB, IC)$, such that $DB_{\mathcal{M}}$ is finite, satisfies φ, *i.e.* $\mathcal{M} \models \varphi$. □

Now we characterize consistent query answers wrt the annotated theory.

Proposition 2. Let DB be a database instance, IC a set of integrity constraints and $Q(\bar{x})$ a query in FO language Σ. It holds:

$$DB \models_c Q(\bar{t}) \text{ iff } \mathcal{T}(DB, IC) \models_\Delta Q^{an}(\bar{t}). \quad \square$$

Example 6. (example 5 continued) For consistently answering the query $Q(x)$, we pose the query $Q^{an}(x)$ to the minimal models of $\mathcal{T}(DB, IC)$. The answer we obtain from *every* minimal model is $x = kafka$. □

According to this proposition, in order to consistently answer queries, we are left with the problem of evaluating minimal entailment wrt the annotated theory. In [Arenas et al. 2000a] some limited FO queries were evaluated, but no annotated queries were generated. The original query were answered using ad hoc algorithms that were extracted from theory $\mathcal{T}(DB, IC)$. No advantage was taken from a characterization of consistent answers in terms of minimal entailment from $\mathcal{T}(DB, IC)$. In the next section we will address this issue by taking the original DB instance with the ICs into a logic program that is generated taking advantage of the annotations provided by $\mathcal{T}(DB, IC)$. The query to be posed to the logic program will be built from Q^{an}.

5 Query Answering

In this section we will consider ICs of the form (1), more precisely of the form

$$\bigvee_{i=1}^{n} \neg p_i(\bar{t}_i) \vee \bigvee_{j=1}^{m} q_j(\bar{s}_j) \vee \varphi, \qquad (4)$$

where, for every i and j, p_i and q_j are predicates in P, and φ is a formula containing predicates in B only.

5.1 Logic programming specification of repairs

In order to generate a first order logic program that gives an account of annotations, for each predicate

$p(\bar{x}) \in P$, we introduce a new, predicate $p(\bar{x}, \cdot)$, with an extra argument for annotations. This defines a new FO language, Σ^{ext}, for extended Σ. The repair logic program, $\Pi(DB, IC)$, for DB and IC, is written with predicates from Σ^{ext} and contains the following clauses:

1. For every atom $p(\bar{a}) \in DB$, $\Pi(DB, IC)$ contains the fact $p(\bar{a}, \mathbf{t_d}) \leftarrow$.

2. For every predicate $p \in P$, $\Pi(DB, IC)$ contains the clauses:

   $p(\bar{x}, \mathbf{t}^\star) \leftarrow p(\bar{x}, \mathbf{t_d})$

   $p(\bar{x}, \mathbf{t}^\star) \leftarrow p(\bar{x}, \mathbf{t_a})$

   $p(\bar{x}, \mathbf{f}^\star) \leftarrow p(\bar{x}, \mathbf{f_a})$,

   where $\mathbf{t}^\star, \mathbf{f}^\star$ are new, auxiliary elements in the domain of annotations.

3. For every constraint of the form (4), $\Pi(DB, IC)$ contains the clause:

   $$\bigvee_{i=1}^{n} p_i(\bar{t}_i, \mathbf{f_a}) \vee \bigvee_{j=1}^{m} q_j(\bar{s}_j, \mathbf{t_a}) \longleftarrow \bigwedge_{i=1}^{n} p_i(\bar{t}_i, \mathbf{t}^\star) \wedge \bigwedge_{j=1}^{m} q_j(\bar{s}_j, \mathbf{f}^\star) \wedge \bar{\varphi},$$

   where $\bar{\varphi}$ represents the negation of φ.

Intuitively, the clauses in 3. say that when the IC is violated (the body), then the DB has to be repaired according to one of the alternatives shown in the head. Since there may be interactions between constraints, these single repairing steps may not be enough to restore the consistency of the DB. We have to make sure that the repairing process continues and stabilizes in a state where all the ICs hold[2]. This is the role of the clauses in 2. containing the new annotations $\mathbf{t}^\star$, that groups together those atoms annotated with $\mathbf{t_d}$ and $\mathbf{t_a}$, and $\mathbf{f}^\star$, that does the same with $\mathbf{f_d}$ and $\mathbf{f_a}$ (with the help of Definition 4 below).

The following example shows the interaction of a FD and an inclusion dependency. When atoms are deleted in order to satisfy the FD, the inclusion dependency could be violated, and in a second step it should be repaired. At that second step, the annotations $\mathbf{t}^\star$ and $\mathbf{f}^\star$, computed at the first step where the FD was repaired, will detect the violation of the inclusion dependency and perform the corresponding repairing process.

[2] In [Arenas et al. 2000b] a direct specification of database repairs by means of disjunctive logic programs with a stable model semantics was presented. Those programs contained both repair triggering rules and "stabilizing" rules.

Example 7. (example 1 cont.) We extend the schema with table $Eurbook(author, name, publYear)$, for European books. Now, DB also contains the literal $Eurbook(kafka, metamorph, 1919)\}$. If in addition to the ICs we had before, we include in IC the set inclusion dependency $\forall xyz\ (Eurbook(x, y, z) \rightarrow Book(x, y, z))$, we obtain the following program $\Pi(DB, IC)$:

1. $EurBook(kafka, metamorph, 1919, \mathbf{t_d}) \leftarrow$

   $Book(kafka, metamorph, 1919, \mathbf{t_d}) \leftarrow$

   $Book(kafka, metamorph, 1915, \mathbf{t_d}) \leftarrow$.

2. $Book(x, y, z, \mathbf{t}^\star) \leftarrow Book(x, y, z, \mathbf{t_d})$

   $Book(x, y, z, \mathbf{t}^\star) \leftarrow Book(x, y, z, \mathbf{t_a})$

   $Book(x, y, z, \mathbf{f}^\star) \leftarrow Book(x, y, z, \mathbf{f_a})$

   $Eurbook(x, y, z, \mathbf{t}^\star) \leftarrow Eurbook(x, y, z, \mathbf{t_d})$

   $Eurbook(x, y, z, \mathbf{t}^\star) \leftarrow Eurbook(x, y, z, \mathbf{t_a})$

   $Eurbook(x, y, z, \mathbf{f}^\star) \leftarrow Eurbook(x, y, z, \mathbf{f_a})$.

3. $Book(x, y, z, \mathbf{f_a}) \vee Book(x, y, w, \mathbf{f_a}) \leftarrow Book(x, y, z, \mathbf{t}^\star) \wedge Book(x, y, w, \mathbf{t}^\star) \wedge z \neq w$

   $Eurbook(x, y, z, \mathbf{f_a}) \vee Book(x, y, z, \mathbf{t_a}) \leftarrow Eurbook(x, y, z, \mathbf{t}^\star) \wedge Book(x, y, z, \mathbf{f}^\star)$. □

In order to have a semantics for our repair programs, we define their models. Since the negative information in a database instance is only implicitly available and we want to avoid explicitly representing it, we need to specify when negative information of the form $p(\bar{t}, \mathbf{f}^\star)$ is true of a model.

Definition 4. (a) Let I be an Herbrand interpretation for Σ^{ext} and φ a FO formula in Σ^{ext}. The definition of $\star$-satisfaction of φ by I, denoted $I \models_\star \varphi$, is as usual, except that for a ground atomic formula $p(\bar{a}, \mathbf{f}^\star)$ it holds: $I \models_\star p(\bar{a}, \mathbf{f}^\star)$ iff $p(\bar{a}, \mathbf{f}^\star) \in I$ or $p(\bar{a}, \mathbf{t_d}) \notin I$.
(b) An Herbrand interpretation $\mathcal{M}$ is a *$\star$-model* of $\Pi(DB, IC)$ if for every (ground instantiation of a) clause $(\bigvee_{i=1}^{n} a_i \leftarrow \bigwedge_{j=1}^{m} b_j) \in \Pi(DB, IC)$, $\mathcal{M} \not\models_\star \bigwedge_{j=1}^{m} b_j$ or $\mathcal{M} \models_\star \bigvee_{i=1}^{n} a_i$. □

Definition 5. (a) An atom $p(\bar{a})$ in a model $\mathcal{M}$ of a program is *plausible* if it belongs to the head of a clause in $\Pi(DB, IC)$ such that $\mathcal{M}$ $\star$-satisfies the body of the clause. A model of a program is plausible if every atom in it is plausible.
(b) A model is *coherent* if it does not contain both $p(\bar{a}, \mathbf{t_a})$ and $p(\bar{a}, \mathbf{f_a})$. □

We will be interested only in the Herbrand $\star$-models of the program that are minimal wrt set inclusion[3] and plausible and *coherent*. Notice that in a coherent model we may still find both atoms $p(\bar{a}, \mathbf{t}^\star)$ and $p(\bar{a}, \mathbf{f}^\star)$. Notice also that a plausible atom may belong to a supported disjunctive head [Lobo 1998], without the other disjuncts being forced to be false.

It is possible to prove that every minimal, plausible and coherent $\star$-model of $\Pi(DB, IC)$ is a model of $\Pi(DB, IC)$ (in the usual sense). Furthermore, it is easy to see from the definition of a $\star$-model of a program that we could keep the classical notion of satisfaction by including in the program the additional clauses $p(\bar{x}, \mathbf{f}^\star) \leftarrow \mathit{not}\ p(\bar{x}, \mathbf{t_d})$, that would include in the models all the negative information we usually keep implicit via the closed world assumption. Moreover, we would be left with a normal disjunctive program, for which a stable model semantics could be used [Gelfond et al. 1988] (see section 5.3 below).

Example 8. (example 7 cont.) The coherent plausible minimal $\star$-models of the program presented in example 7 are:

$$\begin{aligned}\mathcal{M}_1 = \{&\mathit{Eurbook}(\mathit{kafka}, \mathit{metamorph}, \mathit{1919}, \mathbf{t_d}),\\ &\mathit{Eurbook}(\mathit{kafka}, \mathit{metamorph}, \mathit{1919}, \mathbf{t}^\star),\\ &\mathit{Book}(\mathit{kafka}, \mathit{metamorph}, \mathit{1919}, \mathbf{t_d}),\\ &\mathit{Book}(\mathit{kafka}, \mathit{metamorph}, \mathit{1919}, \mathbf{t}^\star),\\ &\mathit{Book}(\mathit{kafka}, \mathit{metamorph}, \mathit{1915}, \mathbf{t_d}),\\ &\mathit{Book}(\mathit{kafka}, \mathit{metamorph}, \mathit{1915}, \mathbf{t}^\star),\\ &\mathit{Book}(\mathit{kafka}, \mathit{metamorph}, \mathit{1915}, \mathbf{f_a}),\\ &\mathit{Book}(\mathit{kafka}, \mathit{metamorph}, \mathit{1915}, \mathbf{f}^\star)\}.\end{aligned}$$

$$\begin{aligned}\mathcal{M}_2 = \{&\mathit{Eurbook}(\mathit{kafka}, \mathit{metamorph}, \mathit{1919}, \mathbf{t_d}),\\ &\mathit{Eurbook}(\mathit{kafka}, \mathit{metamorph}, \mathit{1919}, \mathbf{t}^\star),\\ &\mathit{Eurbook}(\mathit{kafka}, \mathit{metamorph}, \mathit{1919}, \mathbf{f_a}),\\ &\mathit{Eurbook}(\mathit{kafka}, \mathit{metamorph}, \mathit{1919}, \mathbf{f}^\star),\\ &\mathit{Book}(\mathit{kafka}, \mathit{metamorph}, \mathit{1919}, \mathbf{t_d}),\\ &\mathit{Book}(\mathit{kafka}, \mathit{metamorph}, \mathit{1919}, \mathbf{t}^\star),\\ &\mathit{Book}(\mathit{kafka}, \mathit{metamorph}, \mathit{1919}, \mathbf{f_a}),\\ &\mathit{Book}(\mathit{kafka}, \mathit{metamorph}, \mathit{1919}, \mathbf{f}^\star),\\ &\mathit{Book}(\mathit{kafka}, \mathit{metamorph}, \mathit{1915}, \mathbf{t_d}),\\ &\mathit{Book}(\mathit{kafka}, \mathit{metamorph}, \mathit{1915}, \mathbf{t}^\star)\}.\end{aligned}$$

□

Notice that, in contrast to the minimal models of the annotated theory $\mathcal{T}(DB, IC)$, the $\star$-models of the program will include the database contents with its original annotations ($\mathbf{t_d}$). Every time there is an atom in a model annotated with $\mathbf{t_d}$ or $\mathbf{t_a}$, it will appear annotated with $\mathbf{t}^\star$. From these models we should be able to "read" database repairs. Every $\star$-model of the logic program has to be interpreted.

Definition 6. Given a coherent plausible $\star$-model $\mathcal{M}$ of $\Pi(DB, IC)$, its *interpretation*, $i(\mathcal{M})$, is a new Herbrand interpretation obtained from $\mathcal{M}$ as follows:

1. If $p(\bar{a}, \mathbf{f_a})$ belongs to $\mathcal{M}$, then $p(\bar{a}, \mathbf{f}^{\star\star})$ belongs to $i(\mathcal{M})$.
2. If neither $p(\bar{a}, \mathbf{t_d})$ nor $p(\bar{a}, \mathbf{t_a})$ belongs to $\mathcal{M}$, then $p(\bar{a}, \mathbf{f}^{\star\star})$ belongs to $i(\mathcal{M})$.
3. If $p(\bar{a}, \mathbf{t_d})$ belongs to $\mathcal{M}$ and $p(\bar{a}, \mathbf{f_a})$ does not belong to $\mathcal{M}$, then $p(\bar{a}, \mathbf{t}^{\star\star})$ belongs to $i(\mathcal{M})$.
4. If $p(\bar{a}, \mathbf{t_a})$ belongs to $\mathcal{M}$, then $p(\bar{a}, \mathbf{t}^{\star\star})$ belongs to $i(\mathcal{M})$. □

Notice that the interpreted models contain two new annotations, $\mathbf{t}^{\star\star}, \mathbf{f}^{\star\star}$, in the last arguments. The first one groups together those atoms annotated either with $\mathbf{t_a}$ or with $\mathbf{t_d}$ but not $\mathbf{f_a}$. Intuitively, the latter correspond to those annotated with $\mathbf{t}$ in the models of $\mathcal{T}(DB, IC)$. A similar role plays the other new annotation wrt the "false" annotations. These new annotations will simplify the expression of the queries to be posed to the program (see section 5.2). Without them, instead of simply asking $p(\bar{x}, \mathbf{t}^{\star\star})$ (for the tuples in a repair), we would have to ask for $p(\bar{x}, \mathbf{t_a}) \vee (p(\bar{x}, \mathbf{t_d}) \wedge \neg p(\bar{x}, \mathbf{f_a}))$.

Example 9. (example 8 cont.) The interpreted models are:

$$\begin{aligned}i(\mathcal{M}_1) = \{&\mathit{Eurbook}(\mathit{kafka}, \mathit{metamorph}, \mathit{1919}, \mathbf{t}^{\star\star}),\\ &\mathit{Book}(\mathit{kafka}, \mathit{metamorph}, \mathit{1919}, \mathbf{t}^{\star\star}),\\ &\mathit{Book}(\mathit{kafka}, \mathit{metamorph}, \mathit{1915}, \mathbf{f}^{\star\star})\}\end{aligned}$$

$$\begin{aligned}i(\mathcal{M}_2) = \{&\mathit{Eurbook}(\mathit{kafka}, \mathit{metamorph}, \mathit{1919}, \mathbf{f}^{\star\star}),\\ &\mathit{Book}(\mathit{kafka}, \mathit{metamorph}, \mathit{1919}, \mathbf{f}^{\star\star}),\\ &\mathit{Book}(\mathit{kafka}, \mathit{metamorph}, \mathit{1915}, \mathbf{t}^{\star\star})\}.\end{aligned}$$

□

The interpreted models could be easily obtained by adding new rules to the program $\Pi(DB, IC)$. This will be shown in section 5.2. From an interpreted model of the program we can obtain a database instance:

Definition 7.

$$DB^{\Pi}_{i(\mathcal{M})} = \{p(\bar{a}) \mid i(\mathcal{M}) \models p(\bar{a}, \mathbf{t}^{\star\star})\}.$$ □

Example 10. (example 9 cont.) The following database instances obtained from definition 7 are the repairs of DB:

[3] To distinguish them from the Δ-minimal model of the annotated theory.

$DB^{\Pi}_{i(\mathcal{M}_1)} = \{Eurbook(kafka, metamorph, 1919), Book(kafka, metamorph, 1919)\}$,
$DB^{\Pi}_{i(\mathcal{M}_2)} = \{Book(kafka, metamorph, 1915)\}$. □

Theorem 1. If $\mathcal{M}$ is a coherent minimal plausible $\star$-model of $\Pi(DB, IC)$, and $DB^{\Pi}_{i(\mathcal{M})}$ is finite, then $DB^{\Pi}_{i(\mathcal{M})}$ is a repair of DB with respect to IC. Furthermore, the repairs obtained in this way are all the repairs of DB. □

5.2 The query program

Given a first order query Q, we want the consistent answers from DB. In consequence, we need those atoms that are simultaneously true in every interpreted coherent minimal plausible $\star$-model of the program $\Pi(DB, IC)$. They are obtained through the query $Q^{\star\star}$, obtained from Q by replacing, for $p \in P$, every positive literal $p(\bar{s})$ by $p(\bar{s}, \mathbf{t}^{\star\star})$ and every negative literal $\neg p(\bar{s})$ by $p(\bar{s}, \mathbf{f}^{\star\star})$. This query corresponds to the annotated version Q^{an} of Q (see section 4). Now $Q^{\star\star}$ can be transformed into a query program $\Pi(Q^{\star\star})$ by a standard transformation [Lloyd 1987, Abiteboul et al. 1995]. This query program will be run in combination with a program, Π^{int}, that specifies the interpreted models. This program can be obtained extending $\Pi(DB, IC)$ with the following rules:

$p(\bar{x}, \mathbf{t}^{\star\star}) \leftarrow p(\bar{x}, \mathbf{t_a})$,
$p(\bar{x}, \mathbf{t}^{\star\star}) \leftarrow p(\bar{x}, \mathbf{t_d})$, *not* $p(\bar{x}, \mathbf{f_a})$,
$p(\bar{x}, \mathbf{f}^{\star\star}) \leftarrow p(\bar{x}, \mathbf{f_a})$,
$p(\bar{x}, \mathbf{f}^{\star\star}) \leftarrow$ *not* $p(\bar{x}, \mathbf{t_d})$, *not* $p(\bar{x}, \mathbf{t_a})$.

The extended program, that now contains weak negation, is basically stratified due to the different ground annotations in the predicates. Thus, its models can be computed by extending the models of the original program in a uniform manner.

Example 11. (example 7 cont.) The program Π^{int} is obtained adding to the program $\Pi(DB, IC)$ of example 7 the following clauses:

$Eurbook(x, y, z, \mathbf{t}^{\star\star}) \leftarrow Eurbook(x, y, z, \mathbf{t_a})$

$Eurbook(x, y, z, \mathbf{f}^{\star\star}) \leftarrow Eurbook(x, y, z, \mathbf{f_a})$

$Eurbook(x, y, z, \mathbf{t}^{\star\star}) \leftarrow Eurbook(x, y, z, \mathbf{t_d})$, *not* $Eurbook(x, y, z, \mathbf{f_a})$

$Eurbook(x, y, z, \mathbf{f}^{\star\star}) \leftarrow$ *not* $Eurbook(x, y, z, \mathbf{t_d})$, *not* $Eurbook(x, y, z, \mathbf{t_a})$

$Book(x, y, z, \mathbf{t}^{\star\star}) \leftarrow Book(x, y, z, \mathbf{t_a})$

$Book(x, y, z, \mathbf{f}^{\star\star}) \leftarrow Book(x, y, z, \mathbf{f_a})$

$Book(x, y, z, \mathbf{t}^{\star\star}) \leftarrow Book(x, y, z, \mathbf{t_d})$, *not* $Book(x, y, z, \mathbf{f_a})$

$Book(x, y, z, \mathbf{f}^{\star\star}) \leftarrow$ *not* $Book(x, y, z, \mathbf{t_d})$, *not* $Book(x, y, z, \mathbf{t_a})$.

For the query $Q(y) : \exists z Book(kafka, y, z)$, we generate $Q^{\star\star}(y) : \exists z Book(kafka, y, z, \mathbf{t}^{\star\star})$, that is transformed into the query program $\Pi(Q^{\star\star})$: $Answer(y) \leftarrow Book(kafka, y, z, \mathbf{t}^{\star\star})$.

The coherent minimal plausible $\star$-models of $\Pi^{int} \cup \Pi(Q^{\star})$ are $\mathfrak{M}_1 = \mathcal{M}_1 \cup i(\mathcal{M}_1) \cup \{Answer(metamorph)\}$ and $\mathfrak{M}_2 = \mathcal{M}_2 \cup i(\mathcal{M}_2) \cup \{Answer(metamorph)\}$, where $\mathcal{M}_1, \mathcal{M}_2$ and $i(\mathcal{M}_1), i(\mathcal{M}_2)$ are given in examples 8 and 9, resp. We can see that $y = metamorph$ is a consistent answer. □

5.3 Computing from the program

The repair programs $\Pi(DB, IC)$ introduced in section 5.1 are based on a non classical notion of satisfaction (definition 4). In order to compute from the program using a stable model semantics for disjunctive programs, we build a new program $\Pi^{\neg}(DB, IC)$, obtained from the original one by adding the clause $p(\bar{x}, \mathbf{f}^{\star}) \leftarrow$ *not* $p(\bar{x}, \mathbf{t_d})$ that gives an account of the closed world assumption. It holds:

Proposition 3. If $\mathcal{M}$ is a coherent stable model of $\Pi^{\neg}(DB, IC)$, then $DB^{\Pi}_{\mathcal{M}}$ is a repair of DB with respect to IC; and every repair can be obtained in this way. □

In consequence, the database repairs can be specified using disjunctive logic programs with a stable model semantics [Gelfond et al. 1988, Gelfond et al. 1991], and an implementation of this semantics, like *DLV* [Eiter et al. 2000], can be used to compute both repairs and consistent answers. Notice that *DLV* implements denial constraints [Buccafurri et al. 2000], which can be used to keep the coherent stable models only, by pruning those models that do not satisfy $\leftarrow p(\bar{x}, \mathbf{t_a}), p(\bar{x}, \mathbf{f_a})$.

Example 12. Consider the database instance $\{p(a)\}$ that is inconsistent wrt the set inclusion dependency $\forall x\ (p(x) \rightarrow q(x))$. The program $\Pi^{\neg}(DB, IC)$ contains the following clauses:

1. The following rules do not depend on ICs
   $p(x, \mathbf{f}^{\star}) \leftarrow p(x, \mathbf{f_a})$
   $p(x, \mathbf{t}^{\star}) \leftarrow p(x, \mathbf{t_a})$

$p(x, \mathbf{t}^\star) \leftarrow p(x, \mathbf{t_d})$

$q(x, \mathbf{f}^\star) \leftarrow q(x, \mathbf{f_a})$

$q(x, \mathbf{t}^\star) \leftarrow q(x, \mathbf{t_a})$

$q(x, \mathbf{t}^\star) \leftarrow q(x, \mathbf{t_d})$

2. A single rule capturing the IC

   $p(x, \mathbf{f_a}) \vee q(x, \mathbf{t_a}) \leftarrow p(x, \mathbf{t}^\star), q(x, \mathbf{f}^\star).$

3. Database contents

   $p(a, \mathbf{t_d}) \leftarrow$

4. The new rules for the closed world assumption

   $p(x, \mathbf{f}^\star) \leftarrow \mathit{not}\ p(x, \mathbf{t_d})$

   $q(x, \mathbf{f}^\star) \leftarrow \mathit{not}\ q(x, \mathbf{t_d}).$

5. Denial constraints for coherence

   $\leftarrow p(\bar{x}, \mathbf{t_a}), p(\bar{x}, \mathbf{f_a})$

   $\leftarrow q(\bar{x}, \mathbf{t_a}), q(\bar{x}, \mathbf{f_a}).$

6. Rules for interpreting the models

   $p(x, \mathbf{t}^{\star\star}) \leftarrow p(x, \mathbf{t_a})$

   $p(x, \mathbf{t}^{\star\star}) \leftarrow p(x, \mathbf{t_d}),\ \mathit{not}\ p(x, \mathbf{f_a})$

   $p(x, \mathbf{f}^{\star\star}) \leftarrow p(x, \mathbf{f_a})$

   $p(x, \mathbf{f}^{\star\star}) \leftarrow \mathit{not}\ p(x, \mathbf{t_d}),\ \mathit{not}\ p(x, \mathbf{t_a})$

   $q(x, \mathbf{t}^{\star\star}) \leftarrow q(x, \mathbf{t_a})$

   $q(x, \mathbf{t}^{\star\star}) \leftarrow q(x, \mathbf{t_d}),\ \mathit{not}\ q(x, \mathbf{f_a})$

   $q(x, \mathbf{f}^{\star\star}) \leftarrow q(x, \mathbf{f_a})$

   $q(x, \mathbf{f}^{\star\star}) \leftarrow \mathit{not}\ q(x, \mathbf{t_d}),\ \mathit{not}\ q(x, \mathbf{t_a}).$ □

The programs with annotations we have obtained are very simple in terms of their dependency on the ICs. Furthermore, the programs contain only negation as failure and become head-cycle free [Ben-Eliyahu 1994], and thus can be transformed into non disjunctive normal programs with a stable model semantics [Gelfond et al. 1988], for which more efficient and appropriate implementations can be used, in particular, for answering open queries.

As mentioned before, consistent answers can be obtained "running" the query program introduced in section 5.2 in combination with the repair program $\Pi^{\neg}(DB, IC)$, under the skeptical stable model semantics, that sanctions as true what is true of all stable models.

6 Conclusions

Extending work presented in [Arenas et al. 2000a], we have shown how to annotate referential ICs in order to obtain a specification in annotated predicate logic of the class of repairs of a relational database. The correspondence between the minimal models of the annotated first order specification and the database repairs is established. Ongoing work considers the extension of the annotated embedding methodology to the class of all ICs found in DB praxis [Abiteboul et al. 1995, chap. 10].

We formulated the problem of consistent query answering as a problem of non monotonic entailment of a modified query from the annotated theory.

We have presented a general treatment of consistent query answering for first order queries and universal ICs. This is done by means of disjunctive logic programs with a stable model semantics, where annotations are now arguments, that specify the database repairs in the case of universal ICs. In consequence, consistent query answers can be obtained by "running" the program. Ongoing work considers the extension of the logic programs to include referential (and more general) ICs.

In [Greco et al. 2001], a general methodology for specifying database repairs wrt universal ICs is presented. There, disjunctive logic programs with stable model semantics are used. They also consider the problem of specifying preferences between possible repairs. Independently, [Arenas et al. 2000b] also presents a specification of database repairs by means of disjunctive logic programs with a stable model semantics. The programs capture the repairs for binary universal ICs. The programs presented here also work for the whole class of universal ICs, but they are much simpler than those presented in [Greco et al. 2001, Arenas et al. 2000b]; this is due to the simplicity of the stabilizing rules, that take full advantage of the relationship between the annotations. The simplicity is expressed in a much smaller number of rules and the syntactic properties of the programs.

In [Blair et al. 1989, Kifer et al. 1992b, Leach et al. 1996] paraconsistent and annotated logic programs are introduced. In particular, in [Subrahmanian 1994] those programs are used to integrate databases, a problem closely related to inconsistency handling. It is not clear how to use those definitive non disjunctive programs to capture database repairs. Furthermore, notice that the

programs presented in this paper have a completely classical semantics.

In [Gaasterland 1994], annotated logic (programs) were used to specify and obtain answers matching user needs and preferences. Deeper relationships to our work deserve to be explored.

Acknowledgments: Work supported by DIPUC, FONDECYT Grant 1000593, and Carleton University Start-Up Grant 9364-01. We are grateful to Alberto Mendelzon and Marcelo Arenas for stimulating and useful conversations, and comments.

References

[Abiteboul et al. 1995] Abiteboul, S.; Hull, R. and Vianu, V. "Foundations of Databases". Addison-Wesley, 1995.

[Arenas et al. 1999] Arenas, M.; Bertossi, L. and Chomicki, J. "Consistent Query Answers in Inconsistent Databases". In *Proc. ACM Symposium on Principles of Database Systems (ACM PODS'99), Philadelphia)*, pages 68-79, 1999.

[Arenas et al. 2000a] Arenas, M.; Bertossi, L. and Kifer, M. "Applications of Annotated Predicate Calculus to Querying Inconsistent Databases". In *'Computational Logic - CL2000' Stream: 6th International Conference on Rules and Objects in Databases (DOOD'2000)*. Springer Lecture Notes in Artificial Intelligence 1861, pages 926-941.

[Arenas et al. 2000b] Arenas, M.; Bertossi, L. and Chomicki, J. "Specifying and Querying Database Repairs using Logic Programs with Exceptions". In *Flexible Query Answering Systems. Recent Developments*, H.L. Larsen, J. Kacprzyk, S. Zadrozny, H. Christiansen (eds.), Springer, 2000, pp. 27–41.

[Ben-Eliyahu 1994] Ben-Eliyahu, R.; and Dechter, R. "Propositional Semantics for Disjunctive Logic Programs". *Annals of Mathematics in Artificial Intelligence*, 1994, 12:53-87.

[Blair et al. 1989] Blair, H.A. and Subrahmanian, V.S. "Paraconsistent Logic Programming". *Theoretical Computer Science*, 68:135-154, 1989.

[Buccafurri et al. 2000] Buccafurri, F.; Leone, N.; Rullo, P. "Enhancing Disjunctive Datalog by Constraints". *IEEE Transactions on Knowledge and Data Engineering*, 2000, 12(5) : 845-860.

[Eiter et al. 2000] Eiter, T.; Faber, W.; Leone, N.; Pfeifer, G. "Declarative Problem-Solving in DLV". In *Logic-Based Artificial Intelligence*, J. Minker (ed.), Kluwer, 2000, pp. 79–103.

[Gaasterland 1994] Gaasterland, T. and Lobo, J. "Qualified Answers That Reflect User Needs and Preferences". In Proc. 20th International Conference of Very Large Databases (VLDB'94). Morgan Kaufmann Publishers, 1994, pages 309–320.

[Gelfond et al. 1988] Gelfond, M.; and Lifschitz, V. "The Stable Model Semantics for Logic Programming". In *Logic Programming, Proceedings of the Fifth International Conference and Symposium*, R. A. Kowalski and K. A. Bowen (eds.), MIT Press 1988, pp. 1070–1080.

[Gelfond et al. 1991] Gelfond, M.; and Lifschitz, V. "Classical Negation in Logic Programs and Disjunctive Databases". *New Generation Computing*, 1991, 9:365–385.

[Greco et al. 2001] Greco, G.; Greco, S.; and Zumpano, E. "A Logic Programming Approach to the Integration, Repairing and Querying of Inconsistent Databases". In Proc. 17th International Conference on Logic Programming, ICLP'01, Ph. Codognet (ed.), LNCS 2237, Springer, 2001, pp. 348–364.

[Kifer et al. 1992a] Kifer, M. and Lozinskii, E.L. "A Logic for Reasoning with Inconsistency". *Journal of Automated reasoning*, 9(2):179-215, November 1992.

[Kifer et al. 1992b] Kifer, M. and Subrahmanian, V.S. "Theory of Generalized Annotated Logic Programming and its Applications". *Journal of Logic Programming*, 12(4):335-368, April 1992.

[Leach et al. 1996] Leach, S.M. and Lu, J.J. "Query Processing in Annotated Logic Programming: Theory and Implementation". *Journal of Intelligent Information Systems*, 6, January 1996, pp. 33–58.

[Lloyd 1987] Lloyd, J.W. "Foundations of Logic Programming". Springer Verlag, 1987.

[Lobo 1998] Lobo, J.; Minker, J. and Rajasekar, A. "Semantics for Disjunctive and Normal Disjunctive Logic Programs". In *Handbook of Logic*

in Artificial Intelligence and Logic Programming, Vol. 5, D. Gabbay et al. (eds.). Oxford University Press, 1998.

[Subrahmanian 1994] Subrahmanian, V.S. "Amalgamating Knowledge Bases". *ACM Transactions on Database Systems*, 1994, 19(2):291–331.

Dynamical Revision Operators with Memory

Sébastien Konieczny
IRIT
Université Paul Sabatier
31062 Toulouse - France
konieczny@irit.fr

Ramón Pino Pérez
Departamento de Matemáticas
Facultad de Ciencias
Universidad de los Andes
Mérida - Venezuela
pino@ciens.ula.ve

Abstract

We have proposed in previous works [Konieczny and Pino Pérez, 2000; Konieczny and Pino Pérez, 2001] a construction that allows to define operators for iterated revision from classical AGM revision operators. We called those operators *revision operators with memory* and show that the operators obtained have nice logical properties. But those operators can be considered as too conservative, since the revision policy of the agent, encoded as a faithful assignment, does not change during her life. In this paper we propose an extension of those operators, that aims to add more dynamics in the revision process.

1 Introduction

One of the predominant approaches to modelling belief change was proposed by Alchourrón, Gärdenfors and Makinson and is known as the AGM framework [Alchourrón *et al.*, 1985; Gärdenfors, 1988]. The core of this framework is a set of logical properties that a revision operator has to satisfy to guarantee a nice behaviour.

A drawback of AGM definition of revision is that it is a static one, which means that, with this definition of revision operators, one can have a rational one step revision but the conditions for the iteration of the process are very weak. The problem is that AGM postulates state conditions only between the initial knowledge base, the new evidence and the resulting knowledge base. But the way to perform further revisions on the new knowledge base does not depend on the way the old knowledge base was revised.

Numerous proposals have tried to state a logical characterization that adequately models iterated belief change behaviour [Darwiche and Pearl, 1997; Darwiche and Pearl, 1994; Boutilier, 1996; Lehmann, 1995; Nayak *et al.*, 1996; Nayak, 1994; Konieczny and Pino Pérez, 2000] [1]. The more famous one seems to be [Darwiche and Pearl, 1997]. The main idea that is common to all of these works is that the belief base framework is not sufficient to encompass iterated revision, since one needs some additional information for coding the revision policy of the agent. So the need of *epistemic states* to encode the agent's "state of mind" is widely accepted. An epistemic state allows to code the agent's beliefs but also to code her relative confidence in alternative possible states of the world. Epistemic states can be represented by several means: pre-orders on interpretations [Darwiche and Pearl, 1997; Lehmann, 1995], conditionals [Boutilier, 1996; Darwiche and Pearl, 1997], epistemic entrenchments [Williams, 1994; Nayak, 1994], prioritized belief bases [Benferhat *et al.*, 1999; Benferhat *et al.*, 2000], etc. In this work we will focus on the representation of epistemic states in terms of pre-orders on interpretations.

In [Konieczny and Pino Pérez, 2001; Konieczny and Pino Pérez, 2000], we define a family of revision operators that we have called *revision operators with memory*. Those operators can be defined from any classical AGM revision operator [Alchourrón *et al.*, 1985; Katsuno and Mendelzon, 1991] and they have good properties for iterated revision.

In fact revision operators with memory use the faithful assignment provided by the classical AGM revision operator as an *a priori* information. This *a priori* information is attached to the new evidence, and the completed information obtained is then incorporated

[1] see also [Spohn, 1987; Williams, 1994; Williams, 1995]. We do not adress this kind of operators in this paper since they require an additional numerical information with the new evidence.

to the old epistemic state with the usual *primacy of update* requirement.

The ontology for this pre-processing step, associating an additional information to the incoming new evidence is the following. Suppose that the agent has no information (no belief) about the world and learns a (first) new evidence. Then, this new evidence alone can provide more change in the agent's mind than just the addition of a belief.

As an example, suppose that the agent learns $\varphi = a \wedge b \wedge c \wedge d$, where a, b, c, d are atomic formulae. Then her preferred worlds (the ones she finds the more plausible) will be the ones where the four atomic formulae are true. But it can be sensible for her to find the worlds where three of the atomic formulae are true more plausible than the ones where only two are, etc.

So the new evidence does not simply imply a partition between the believed worlds and the unbelieved ones, but defines several stratas, depending of the plausibility of each world, given the new evidence. We call this property *strong primacy of update*.

This induced preferential information was given here by a "Dalal distance" policy[2], but more complex or realistic policies can be also used depending on the particular context.

The point in the definition of revision operators with memory, is that this *a priori* information, carried by a new evidence depends only on the new evidence by itself, and does not depend on the current agent's beliefs. Going back to the previous example, the fact that the worlds where three of the four atomic formulae are true are preferred to the ones where only two are, does not depend on any other information than the new evidence itself. So this *a priori* information has to be added to the new evidence before incorporating it in the agent's epistemic state.

More precisely, the revision policy of revision operators with memory is the following: the revision of the current epistemic state Φ – represented by a pre-order over possible worlds – by a new piece of information α – a formula – is the epistemic state (pre-order) obtained after the following two steps:

- First, take the pre-order $\leq_\alpha$ associated to α by the AGM revision operator (faithful assignment [Katsuno and Mendelzon, 1991]) given at the beginning of the process.
- Second, take the lexicographical pre-order associated to $\leq_\alpha$ and Φ. The pre-order obtained in this way is the new epistemic state.

Note that there is a very static feature in this process: the way in which we associate a pre-order to the new piece of information is always the same; it is given by the *fixed* AGM operator from which we start all the process. In some sense this is contrary to the principle of *priority of the new information.*

In this work we solve this problem. In order to do that we take the revision policy as an epistemic state and naturally this revision policy will change progressively with the successive revisions. The new process can be described in the following manner: first of all, an epistemic state is composed by a faithful assignment, say f, and a distinguished formula ϕ. When α, the new evidence, arrives, we revise as follows:

(i) the new distinguished formula ϕ' will be a formula having as models the minimal models of α with respect to the $f(\phi)$ pre-order.

(ii) The new assignment f' will coincide with f on the formulas not equivalent to ϕ'. On formulas equivalent to ϕ' it will be the lexicographical pre-order associated to $f(\alpha)$ and $f(\phi)$.

Thus, this method allows to incorporate the changes step by step in a very natural way. This process agrees with the postulate of primacy of the new information. Unlike our first revision operators with memory that mix the new piece of information with the oldest information (which is static), our latter operators mix the new piece of information with the current epistemic state.

The rest of the paper is organized as follows: in section 2, we recall the logical characterization of iterated revision operators of Darwiche and Pearl. In section 3, we recall the definition of revision operators with memory and state the general logical results. Then, in section 4, we show how to add more dynamics to revision operators with memory. We conclude in section 5 with some general remarks.

2 Iterated Revision Postulates

We give here a formulation of AGM postulates for belief revision *à la* Katsuno and Mendelzon [Katsuno and Mendelzon, 1991]. More exactly, we give a formulation of these postulates in terms of epistemic states [Darwiche and Pearl, 1997]. The epistemic states framework is an extension of the belief bases one. Intuitively

[2] The Dalal distance [Dalal, 1988] is a Hamming distance between interpretations.

an epistemic state can be seen as a composed information: the beliefs of the agent, plus all the information that the agent needs about how to perform revision (preference ordering, conditionals, etc.). Then we give the additional iteration postulates proposed by Darwiche and Pearl [Darwiche and Pearl, 1997].

2.1 Formal Preliminaries

We will work in the finite propositional case. A belief base φ is a set of formulae, which can be considered as the formula that is the conjunction of its formulae. The set of all interpretations is denoted $\mathcal{W}$. Let φ be a formula, $Mod(\varphi)$ denotes the set of models of φ, *i.e.* $Mod(\varphi) = \{I \in \mathcal{W} : I \models \varphi\}$.

A pre-order $\leq$ is a reflexive and transitive relation, and $<$ is its strict counterpart, i.e $I < J$ if and only if $I \leq J$ and $J \not\leq I$. As usual, $\simeq$ is defined by $I \simeq J$ iff $I \leq J$ and $J \leq I$. A pre-order is total if and only if $\forall I, J, I \leq J$ or $J \leq I$.

To each epistemic state Ψ is associated a belief base $Bel(\Psi)$ which is a propositional formula representing the objective (logical) part of Ψ. The models of Ψ are the models of its associated belief base, thus $Mod(\Psi) = Mod(Bel(\Psi))$. Let Ψ be an epistemic state and μ be a sentence denoting the new information. $\Psi \circ \mu$ denotes the epistemic state resulting of the revision of Ψ by μ. For reading convenience we will write respectively $\Psi \vdash \mu$, $\Psi \wedge \mu$ and $I \models \Psi$ instead of $Bel(\Psi) \vdash \mu$, $Bel(\Psi) \wedge \mu$ and $I \models Bel(\Psi)$.

Two epistemic states are equivalent, noted $\Psi \equiv \Psi'$, if and only if their objective parts are equivalent formulae, *i.e.* $Bel(\Psi) \leftrightarrow Bel(\Psi')$. Two epistemic states are equal, noted $\Psi = \Psi'$, if and only if they are identical. Thus equality is stronger than equivalence.

2.2 AGM Postulates for Epistemic States

Let Ψ be an epistemic state and μ and φ be formulae. An operator $\circ$ that maps an epistemic state Ψ and a formula μ to an epistemic state $\Psi \circ \mu$ is said to be a revision operator on epistemic states if it satisfies the following postulates [Darwiche and Pearl, 1997]:

(R*1) $\Psi \circ \mu \vdash \mu$

(R*2) If $\Psi \wedge \mu \nvdash \bot$, then $\Psi \circ \mu \leftrightarrow \Psi \wedge \mu$

(R*3) If $\mu \nvdash \bot$, then $\Psi \circ \mu \nvdash \bot$

(R*4) If $\Psi_1 = \Psi_2$ and $\mu_1 \leftrightarrow \mu_2$, then $\Psi_1 \circ \mu_1 \equiv \Psi_2 \circ \mu_2$

(R*5) $(\Psi \circ \mu) \wedge \varphi \vdash \Psi \circ (\mu \wedge \varphi)$

(R*6) If $(\Psi \circ \mu) \wedge \varphi \nvdash \bot$, then $\Psi \circ (\mu \wedge \varphi) \vdash (\Psi \circ \mu) \wedge \varphi$

This is nearly the Katsuno and Mendelzon formulation of AGM postulates [Katsuno and Mendelzon, 1991]; the only differences are that we work with epistemic states instead of belief bases and that postulate (R*4) is weaker than its AGM counterpart. See [Darwiche and Pearl, 1997] for a full motivation of this definition.

A representation theorem, stating how revisions can be characterized in terms of pre-orders on interpretations, holds. In order to give such semantical representation, the concept of faithful assignment on epistemic states is defined.

Definition 1 *A function that maps each* epistemic state Ψ *to a pre-order* $\leq_\Psi$ *on interpretations is called a* faithful assignment over epistemic states *if and only if:*

1. *If* $I \models \Psi$ *and* $J \models \Psi$, *then* $I \simeq_\Psi J$
2. *If* $I \models \Psi$ *and* $J \not\models \Psi$, *then* $I <_\Psi J$
3. *If* $\Psi_1 = \Psi_2$, *then* $\leq_{\Psi_1} = \leq_{\Psi_2}$

Now the reformulation of the Katsuno and Mendelzon [Katsuno and Mendelzon, 1991] representation theorem in terms of epistemic states is:

Proposition 1 *A revision operator* $\circ$ *satisfies postulates (R*1-R*6) if and only if there exists a faithful assignment (over epistemic states) that maps each epistemic state* Ψ *to a total pre-order* $\leq_\Psi$ *such that:*

$$Mod(\Psi \circ \mu) = \min(Mod(\mu), \leq_\Psi)$$

Notice that this theorem gives information only on the objective part of the resulting epistemic state, but does not allow to know what is the pre-order associated with $\Psi \circ \mu$, *i.e.* we can not identify the new epistemic state, but only its associated belief base $Mod(\Psi \circ \mu)$. Making the parallel with the classical Katsuno and Mendelzon representation theorem (*cf* definition 2 and [Katsuno and Mendelzon, 1991]), that allows to define exactly what is the belief base $Mod(\Psi \circ \mu)$[3], the last theorem is only a weak representation theorem.

2.3 Darwiche and Pearl Postulates

A strong limitation of AGM revision postulates is that they impose very weak constraints on the iteration of the revision process. Darwiche and Pearl [Darwiche

[3]Recall that classical AGM operators are functions that map a belief base and a formula to a belief base, which is (completely) defined by the theorem, whereas proposition 1 concerns operators that are functions which map an epistemic state and a formula to an epistemic state, that is not completely defined by the theorem.

and Pearl, 1994; Darwiche and Pearl, 1997] proposed postulates for iterated revision. The aim of these postulates is to keep as much as possible of conditional beliefs (a conditional belief can be expressed as "if μ would be the case, then φ must be true") of the old belief base. Those conditional beliefs are encoded in the total pre-orders on interpretations. So, besides postulates (R*1-R*6), a revision operator has to satisfy:

(C1) If $\varphi \vdash \mu$, then $(\Psi \circ \mu) \circ \varphi \equiv \Psi \circ \varphi$

(C2) If $\varphi \vdash \neg\mu$, then $(\Psi \circ \mu) \circ \varphi \equiv \Psi \circ \varphi$

(C3) If $\Psi \circ \varphi \vdash \mu$, then $(\Psi \circ \mu) \circ \varphi \vdash \mu$

(C4) If $\Psi \circ \varphi \nvdash \neg\mu$, then $(\Psi \circ \mu) \circ \varphi \nvdash \neg\mu$

These postulates can be explained as follows: (C1) states that if two pieces of information arrive and if the second implies the first, the second alone would give the same belief base. (C2) says that when two contradictory pieces of information arrive, the second alone would give the same belief base. (C3) states that an information should be retained after revising by a second information such that, when revising the current belief base by it, the first one holds. (C4) says that no piece of information can contribute to its own denial.

3 Revision Operators with Memory

A "classical" AGM revision operator is equivalent to a faithful assignment over belief bases as stated in the following theorem [Katsuno and Mendelzon, 1991].

Definition 2 *A function that maps each* belief base φ *to a pre-order* $\leq_\varphi$ *on interpretations is called a* faithful assignment over belief bases *if and only if:*

1. *If* $I \models \varphi$ *and* $J \models \varphi$, *then* $I \simeq_\varphi J$
2. *If* $I \models \varphi$ *and* $J \not\models \varphi$, *then* $I <_\varphi J$
3. *If* $\varphi_1 \leftrightarrow \varphi_2$, *then* $\leq_{\varphi_1} = \leq_{\varphi_2}$

It is important to note that in what follows, we have two distinct kinds of faithful assignments: one over belief bases and one over epistemic states.

Proposition 2 *A revision operator* $\circ$ *satisfies classical AGM postulates (R1-R6)*[4] *if and only if there exists a faithful assignment (over belief bases) that maps each belief base* φ *to a total pre-order* $\leq_\varphi$ *such that:*

$$Mod(\varphi \circ \mu) = \min(Mod(\mu), \leq_\varphi)$$

[4]It is the same set of postulates than (R*1-R*6) but expressed for belief bases instead of epistemic states (*cf* [Katsuno and Mendelzon, 1991]).

So one can define a revision operator directly by defining the corresponding faithful assignment over belief bases. It is the case for most distance-based revision operators such as Dalal operator [Dalal, 1988; Katsuno and Mendelzon, 1991].

More precisely we say that a revision operator $\circ$ is defined from a distance d iff the following conditions hold:

- d is a distance, that is d is a function $d : \mathcal{W} \times \mathcal{W} \mapsto I\!N$ which satisfies: $d(I,J) = d(J,I)$ and $d(I,J) = 0$ iff $I = J$.
- The distance between an interpretation I and a belief base φ is defined as:
$$d(I,\varphi) = \min\{d(I,J) : J \models \varphi\}$$
- This distance induces a faithful assignment:
$$I \leq_\varphi J \text{ iff } d(I,\varphi) \leq d(J,\varphi)$$
- And the revision operator is defined by
$$Mod(\varphi \circ \mu) = \min(Mod(\mu), \leq_\varphi)$$

One can check that the assignment obtained like this is a faithful assignment and thus that all operators defined in this way satisfy AGM postulates. It can also be easily checked that operators defined in this way do not satisfy many of the iterated revision postulates.

Now we will give a construction that allows, from a given faithful assignment (*i.e.* from a given classical AGM revision operator), to define another revision operator that satisfies AGM postulates but also most of the iterated revision postulates.

First, let us notice that an epistemic state can be represented by a total pre-order on interpretations as suggested by theorem 1 and by several related works (*cf e.g.* [Darwiche and Pearl, 1997; Benferhat *et al.*, 2000]). So, with this particular representation (identifying the epistemic state Ψ with a pre-order $\leq_\Psi$), the belief base $Bel(\Psi)$ is simply the formula whose models are minimal for the pre-order, that is $Bel(\Psi) = \min(\mathcal{W}, \leq_\Psi)$. And the other interpretations are ordered according to their relative plausibility for the agent. For example, $I \leq_\Psi J$ means that the agent that is in the epistemic state Ψ considers I as more plausible than J. It is this preferential information that can be used to encompass the iterated revision behaviour, by considering revision operators as functions that map a pre-order (epistemic state) and a formula (new information) into a new pre-order (epistemic state). This idea is the mainstay in most of iterated revision works [Williams, 1994; Darwiche and Pearl, 1997; Nayak, 1994].

So, using this representation by means of pre-orders on interpretations and theorem 1 we will define a familly of revision operators as follows:

Definition 3 *Suppose that we have a function that maps each belief base φ to a pre-order $\leq_\varphi$. Then, we define the epistemic state (the pre-order) $\Psi \circ \varphi$ resulting of the revision of Ψ by the new information φ as:*

$$I \leq_{\Psi\circ\varphi} J \text{ iff } I <_\varphi J \text{ or } I \simeq_\varphi J \text{ and } I \leq_\Psi J$$

Then one can check that:

Proposition 3 *If the function that maps each belief base φ to a total pre-order $\leq_\varphi$ is a faithful assignment over belief bases, then the revision operator on epistemic states defined in definition 3 satisfies postulates (R*1-R*6). We will call such operators* revision operators with memory.

So, with definition 3, one can start from any epistemic state (total pre-order over interpretations) and carry on iterated revisions. A particular epistemic state we can mention is the "empty" epistemic state, where the agent has no belief and no preferential information, that is, such that $\forall I, J \in \mathcal{W}\ I \simeq J$. We will denote by Ξ this epistemic state. So, the objective part of this epistemic state is $Bel(\Xi) = \top$. It can be considered as the epistemic state generalisation of $\top$ for the belief base framework, since they are both neutral elements for the corresponding operators: $\Psi \circ \Xi = \Psi$ (as $\varphi \circ \top = \varphi$ in the belief base framework). One can consider that all agents start with this epistemic sate (we will consider this in the examples).

Concerning iteration postulates stated by Darwiche and Pearl [Darwiche and Pearl, 1997]:

Proposition 4 *Revision operators with memory satisfy postulates (C1),(C3) and (C4).*

It can be also easily checked that (C2) is satisfied by a unique revision operator with memory, since it demands (in the presence of the other revision postulates), that the pre-order associated to a belief base by the faithful assignment on belief base used in definition 3 is a two-level pre-order with the models of the belief base at the lowest level and the counter-models at the higher one. This operator will be presented in the next section.

So most of our revision operators with memory do not satisfy (C2). But we do not consider this as a drawback. We rather think that it is (C2) that is not fully satisfactory. See [Konieczny and Pino Pérez, 2001] for more explanations on this point.

For a more complete logical characterization of this family of operators see [Konieczny and Pino Pérez, 2000].

3.1 Basic memory operator

Let us define the assignment that maps each belief base to a pre-order in the following way:

Definition 4 *Let φ be a belief base, the pre-order $\leq^b_\varphi$ associated to φ is defined as :*

$$I \leq^b_\varphi J \text{ if and only if } I \models \varphi \text{ or } (I \not\models \varphi \text{ and } J \not\models \varphi)$$

So we have what we shall call a basic order, which is a two-level order (at most), with the models of φ at the lowest level and the other worlds at the highest level.

Definition 5 *The* basic memory operator *is the memory operator obtained from this assignment (i.e. the operator obtained by definitions 4 and 3).*

It is worthy to note that if one uses this faithful assigment (definition 4) to define a classical AGM operator (proposition 2), one obtains the *full meet revision operator* which is not quite a good operator. But, even with this basic order on belief bases in the revision with memory framework, one can build very complex epistemic states. This is due to revision memory.

The assignment of Definition 4 is a faithful assignment on belief bases; with propositions 3 and 4, it is easy to show that:

Proposition 5 *The only revision operator with memory that satisfies (R*1-R*6) and (C1-C4) is the basic memory revision operator.*

This operator has been already studied in the literature under different particular representations: in [Nayak, 1994] with epistemic entrenchments, in [Benferhat *et al.*, 1999; Papini, 2001] with polynomials and syntactic belief bases. Finally, we can note that Liberatore has shown [Liberatore, 1997] that several problems are computationally simpler for the basic memory operator than for the other iterated belief revision proposals (including Boutilier's natural revision [Boutilier, 1993], Lehmann's ranking revision [Lehmann, 1995] and Williams' transmutations [Williams, 1994]).

3.2 Dalal memory operator

We use in this section the Hamming distance between interpretations[5]. The Dalal distance between an interpretation I and a belief base φ is defined as $d(I,\varphi) = \min_{J \models \varphi}(d(I,J))$.

Let's define the assignment that maps each belief base to a pre-order in the following way:

Definition 6 *Let φ be a belief base, the pre-order $\leq_\varphi^d$ associated to φ is defined as :*

$$I \leq_\varphi^d J \text{ if and only if } d(I,\varphi) \leq d(J,\varphi)$$

So we have a pre-order with the models of φ at the lowest level and the other worlds in the higher levels, according to their Dalal distance.

Definition 7 *The* Dalal memory operator *is the memory operator obtained from this assignment (i.e. the operator obtained by definitions 6 and 3).*

We can show through a simple example that this operator differs from the classical Dalal revision operator [Dalal, 1988; Katsuno and Mendelzon, 1991]. Let a and b be two propositional letters and consider for example the sequence $\Psi = \Xi \circ a \circ b \circ \neg(a \wedge b)$. The classical Dalal operator gives $Bel(\Psi) = (a \wedge \neg b) \vee (\neg a \wedge b)$, whereas Dalal memory operator gives $Bel(\Psi) = (\neg a \wedge b)$. This behaviour seems more natural since at the next to last step we learned that b was true, and it is normal to keep some credit for this evidence in the following step. It is in this way, that our operators use revision with "memory".

4 Dynamical Revision Operators with Memory

For revision operators with memory, the revision policy is fixed once the operator is fixed. For example for the Dalal memory operator, the way to associate a pre-order to each new evidence is completely determined at the beginning of the process by the Dalal distance.

So, whereas the aim of revision operators with memory is to give a strong preference to the new evidence, one can object that the faithful assignment used to associate a pre-order to the new evidence does not change and so, that an old information is used in each revision step.

The solution to cope with this objection is to find a way to change the faithful assignment during the course of revisions.

Such a solution will be given in this section.

So, first, let's sum up the way revision operators with memory work :

- The definition of a particular operator lies in the chosen faithful assignment over belief bases. Let's call f such an assignment. So, for each formula φ, f associates a total pre-order $f(\varphi)$ (also noted $\leq_{f(\varphi)}$) satisfying the conditions of definition 2.
- Each time a new evidence φ comes, the operator associates to it its corresponding pre-order $f(\varphi)$.
- The new epistemic state is the result of incorporating the pre-order in the old epistemic state, giving preference to the new evidence (*i.e.* to the pre-order) by using a lexicographical order : $\leq_{\Phi \circ \varphi} = \leq_{lex(f(\varphi),\Phi)}$[6].

So, what we want now is to be able to change f during the agent's life. That is, to change the revision policy of the agent, so that when a new evidence comes, it is not always associated to the same pre-order.

The idea is to start from an *a priori* faithful assignment over belief bases such as for revision by memory operators, but then to modify it at each revision step. To be able to do that, we have to use a slightly more general definition of epistemic states. The (representation of) epistemic states we use for revision operators with memory are pre-orders on interpretations $\leq_\Phi$, from which we can extract the corresponding belief base $Bel(\Phi) = min(\mathcal{W}, \leq_\Phi)$.

For dynamical revision operators with memory, the representation of an epistemic state we use is a couple $\Phi = (\varphi, f)$, where φ is the current belief base and f is the current faithful assignment (So, with this representation, we can extract the pre-order corresponding to the belief base : $f(\varphi)$, and straightforwardly $Bel(\Phi) = \varphi$).

As for classical revision operators with memory, to define a particular dynamical revision with memory operator, one needs an initial, *a priori* faithful assignment over belief bases (*i.e.* a classical AGM revision operator), that will encode the initial revision policy of the agent.

So let's define dynamical revision operators with memory:

[5] The Hamming distance between two interpretations is the number of propositional letters on which the two interpretations differ.

[6] Where $I \leq_{lex(\leq_1,\leq_2)} J$ means $I <_1 J$ or ($I \simeq_1 J$ and $I \leq_2 J$).

Definition 8 *Let $\Phi = (\varphi, f)$ be an epistemic state and let μ be a formula denoting a new evidence. We define the new epistemic state $\Phi \circ \mu$, resulting of the* dynamical revision with memory *of Φ by μ, as $\Phi \circ \mu = (\varphi', f')$, where φ' is a formula whose models are $\min(Mod(\mu), f(\varphi))$, and f' is a function (faithful assignment over belief bases) that is identical to f for each belief base ψ except when $\psi \leftrightarrow \varphi'$. In this case $f'(\psi)$ is defined as :*

$$I \leq_{f'(\psi)} J \text{ iff } I <_{f(\mu)} J \text{ or } I \simeq_{f(\mu)} J \text{ and } I \leq_{f(\varphi)} J$$

So, pointwise, the dynamical operators work exactly the same way as memory operators. The difference is that they also change the given faithful assignment over belief bases at each step. One could believe that the difference between the two families of operators is not huge, since the corresponding pre-orders (faithful assignment) used change only for one value at each step. But as we will see next the dynamical revision operators with memory satisfy the following postulate that the revision operators with memory do not always satisfy (*cf* example 1):

(C5) If $Bel(\Phi) \leftrightarrow \mu$ then $\Phi \circ \mu = \Phi$.

This axiom says that the current epistemic state does not change in all cases where the new piece of information coincides with the observable part of this epistemic state. Note that this axiom is almost trivial in the classical AGM framework[7]. But in the framework of complex epistemic states it is not the case. In fact, as we already mentioned, the revision operators with memory do not always satisfy (C5) as can be seen in the following example.

Example 1 *We are reasoning about two ships (coded by two variables). Suppose we start from the Dalal classical AGM revision operator. Let Φ be the epistemic state with observable part being the following formula:* "only one of the two ships is working" *($Bel(\Phi) = (l \wedge \neg r) \vee (\neg l \wedge r)$). Let μ be the formula expressing that* "the ship on the left is not working" *($\mu = \neg l$). The epistemic state after the revision $\Phi \circ \mu$ using Dalal memory operator has observable part the formula expressing* "only the ship on the right is working". *The other (conditional) information of this epistemic state can be described by the conditionals* "if the ship on the right is not working then the two ships are bad" *and* "if the ship on the left is working then only the ship on the left is working". *Now if we revise this current epistemic state by the fact that* "only the ship on the right is working" *($\varphi = \neg l \wedge r$), which is indeed the observable part of the current epistemic state, we obtain a different epistemic state in which for instance we have the conditional* "if the ship on the left is working then the two ships are working". *On the contrary, the current epistemic state does not change after revision by this new information when the operator is the dynamical Dalal revision with memory operator.*

We illustrate this example below. In order to do that consider a language $\mathcal{L}$ with only two propositional letters l and r. We will denote interpretations simply by the truth assignment, i.e 10 *denotes the interpretation mapping l to true and r to false. Two interpretations are equivalent, with respect to the pre-order, if they appear at the same level. An interpretation I is better than another interpretation J ($I < J$) if it appears at a lower level . $\circ_{MD}$ denotes the Dalal revision with memory operator and $\circ_{DMD}$ the dynamical Dalal with memory operator.*
Let's see the pre-order associated to some belief bases by the faithful assignment over belief bases given by the Dalal distance:

$$\leq_{\Phi}^{D} = \begin{matrix} 11\ 00 \\ 01\ 10 \end{matrix} \qquad \leq_{\mu}^{D} = \begin{matrix} 10\ 11 \\ 00\ 01 \end{matrix} \qquad \leq_{\varphi}^{D} = \begin{matrix} 10 \\ 11\ 00 \\ 01 \end{matrix}$$

And the epistemic states reached by the operators are:

$$\leq_{\Phi} = \begin{matrix} 11\ 00 \\ 01\ 10 \end{matrix} \qquad \leq_{\Phi \circ_{MD} \mu} = \begin{matrix} 11 \\ 10 \\ 00 \\ 01 \end{matrix} \qquad \leq_{\Phi \circ_{MD} \mu \circ_{MD} \varphi} = \begin{matrix} 10 \\ 11 \\ 00 \\ 01 \end{matrix}$$

$$\leq_{\Phi} = \begin{matrix} 11\ 00 \\ 01\ 10 \end{matrix} \qquad \leq_{\Phi \circ_{DMD} \mu} = \begin{matrix} 11 \\ 10 \\ 00 \\ 01 \end{matrix} \qquad \leq_{\Phi \circ_{DMD} \mu \circ_{DMD} \varphi} = \begin{matrix} 11 \\ 10 \\ 00 \\ 01 \end{matrix}$$

Note that the idea here is that, when the agent receives a new evidence that she has met before, the repetition of this evidence suggests that the old beliefs of the agent were correct, and so she holds on to the last pre-order that corresponds to this evidence.

In fact, if one considers the definition of iterated revision operators according to Darwiche and Pearl (cf section 2.2), it amounts to say that we change the revision operators at each step, since the corresponding faithful assignment over epistemic states changes at each step. So, in a sense, dynamical revision operators with memory are definable by a family of revision operators with memory (that corresponds to the set

[7] In that framework, it is a consequence of the other axioms.

of all faithful assignments over belief bases reached by the course of revisions).

Concerning the logical properties of this family of operators, it is easy to check the following:

Proposition 6 *A dynamical revision operator with memory satisfies (R*1)-(R*6). It satisfies (C1), (C3), (C4) and (C5) but it never satisfies (C2).*

Finally, as another example, let's see the behaviour of the full meet revision operator $\circ_B$, the basic memory operator $\circ_{MB}$ and the dynamical basic memory operator $\circ_{DMB}$ (they are all built from the same faithful assignment over belief bases) on the same situations.

Example 2 *Consider a language $\mathcal{L}$ with only two propositional letters a and b (considered in that order for the valuations). Let's see the pre-order associated to some belief bases by the faithful assignment over belief bases given by the Basic distance:*

$$\leq_a^B = \begin{matrix} 00\ 01 \\ 10\ 11 \end{matrix} \qquad \leq_b^B = \begin{matrix} 00\ 10 \\ 01\ 11 \end{matrix}$$

$$\leq_{a\wedge b}^B = \begin{matrix} 00\ 01\ 10 \\ 11 \end{matrix} \qquad \leq_{\neg a}^B = \begin{matrix} 10\ 11 \\ 00\ 01 \end{matrix}$$

And the epistemic states reached by the operators are:

$$\leq_{a\circ_{MB}b} = \leq_{a\circ_{DMB}b} = \begin{matrix} 00 \\ 10 \\ 01 \\ 11 \end{matrix}$$

$$\leq_{a\circ_{MB}b\circ_{MB}\neg a} = \leq_{a\circ_{DMB}b\circ_{DMB}\neg a} = \begin{matrix} 10 \\ 11 \\ 00 \\ 01 \end{matrix}$$

$$\leq_{a\circ_{MB}b\circ_{MB}\neg a\circ_{MB}a\wedge b} = \begin{matrix} 10 \\ 00 \\ 01 \\ 11 \end{matrix}$$

$$\leq_{a\circ_{DMB}b\circ_{DMB}\neg a\circ_{DMB}a\wedge b} = \begin{matrix} 00 \\ 10 \\ 01 \\ 11 \end{matrix}$$

So we have that:

$$\begin{aligned} a \circ_B b \circ_B \neg a \circ_B a \wedge b \circ_B \neg b &\equiv \neg b \\ a \circ_{MB} b \circ_{MB} \neg a \circ_{MB} a \wedge b \circ_{MB} \neg b &\equiv \neg a \wedge \neg b \\ a \circ_{DMB} b \circ_{DMB} \neg a \circ_{DMB} a \wedge b \circ_{DMB} \neg b &\equiv a \wedge \neg b \end{aligned}$$

As noted previously, the full meet revision operator $\circ_B$ does not have a very good behaviour : each time the new evidence contradicts the current beliefs, the new beliefs are only the logical consequences of the new evidence. So, it absolutely does not consider the previous revisions. With the revision operator with memory $\circ_{MB}$, the agent is able to build complex epistemic states (pre-orders), that lead to a satisfactory behaviour for iterated revision. With this operator, the two evidences $\neg a$ and $\neg b$ recently learned lead to this belief base. With the dynamical revision operator with memory $\circ_{DMB}$, the evidence learned at the next to last step $(a \wedge b)$ recalls the agent the last time she had this belief (after $a \circ_{DMB} b$), and this modifies her epistemic state.

5 Conclusion

It is worthy to note that the two families of operators defined, revision with memory and dynamical revision with memory, are revision operators in the sense of Darwiche and Pearl, that is, they map an epistemic state and a formula (new evidence) to an epistemic state. The input is a formula and the pre-processing step, that associates a pre-order to that formula is part of the revision operator and not a part of the input. So, this work differs in that respect to [Benferhat *et al.*, 2000], which considers revision of epistemic states by epistemic states. This work return to the old AGM tradition in which one revises a complex epistemic state (a theory) by a simple fact (a formula).

An interesting future work is to try to characterize logically the dynamical revision with memory operators. A very first step in this sense is the (C5) axiom.

Acknowledgements

Thanks to Professor Olga Porras for her careful reading and numerous suggestions to improve the English version of this paper.

References

[Alchourrón *et al.*, 1985] C. E. Alchourrón, P. Gärdenfors, and D. Makinson. On the logic of theory change: Partial meet contraction and revision functions. *Journal of Symbolic Logic*, 50:510–530, 1985.

[Benferhat *et al.*, 1999] S. Benferhat, D. Dubois, and O. Papini. A sequential reversible belief revision method based on polynomials. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI'99)*, 1999.

[Benferhat *et al.*, 2000] S. Benferhat, S. Konieczny, O. Papini, and R. Pino Pérez. Iterated revision by epistemic states: axioms, semantics and syntax. In *Proceedings of the fourteenth European Conference*

on Artificial Intelligence (ECAI'00), pages 13–17, 2000.

[Boutilier, 1993] C. Boutilier. Revision sequences and nested conditionals. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence (IJCAI'93)*, 1993.

[Boutilier, 1996] C. Boutilier. Iterated revision and minimal change of conditional beliefs. *Journal of Philosophical Logic*, 25(3):262–305, 1996.

[Dalal, 1988] M. Dalal. Investigations into a theory of knowledge base revision: preliminary report. In *Proceedings of the National Conference on Artificial Intelligence (AAAI'88)*, pages 475–479, 1988.

[Darwiche and Pearl, 1994] A. Darwiche and J. Pearl. On the logic of iterated belief revision. In Morgan Kaufmann, editor, *Theoretical Aspects of Reasoning about Knowledge: Proceedings of the 1994 Conference (TARK'94)*, pages 5–23, 1994.

[Darwiche and Pearl, 1997] A. Darwiche and J. Pearl. On the logic of iterated belief revision. *Artificial Intelligence*, 89:1–29, 1997.

[Gärdenfors, 1988] P. Gärdenfors. *Knowledge in flux.* MIT Press, 1988.

[Katsuno and Mendelzon, 1991] H. Katsuno and A. O. Mendelzon. Propositional knowledge base revision and minimal change. *Artificial Intelligence*, 52:263–294, 1991.

[Konieczny and Pino Pérez, 2000] S. Konieczny and R. Pino Pérez. A framework for iterated revision. *Journal of Applied Non-Classical Logics*, 10(3-4):339–367, 2000.

[Konieczny and Pino Pérez, 2001] S. Konieczny and R. Pino Pérez. Some operators for iterated revision. In *Proceedings of the Sixth European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU'01)*, Lecture Notes in Artificial Intelligence 2143, pages 498–509, 2001.

[Lehmann, 1995] D. Lehmann. Belief revision, revised. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI'95)*, pages 1534–1540, 1995.

[Liberatore, 1997] P. Liberatore. The complexity of iterated belief revision. In *Proceedings of the Sixth International Conference on Database Theory (ICDT'97)*, pages 276–290, 1997.

[Nayak *et al.*, 1996] A. C. Nayak, N. Y. Foo, M. Pagnucco, and A. Sattar. Changing conditional beliefs unconditionally. In *Proceedings of the Sixth Conference of Theoretical Aspects of Rationality and Knowledge (TARK'96)*, pages 119–135. Morgan Kaufmann, 1996.

[Nayak, 1994] A. Nayak. Iterated belief change based on epistemic entrenchment. *Erkenntnis*, 41:353–390, 1994.

[Papini, 2001] O. Papini. *Frontiers of Belief Revision*, chapter Iterated revision operators stemming from the history of an agent's observations, pages 281–303. Kluwer, 2001.

[Spohn, 1987] W. Spohn. Ordinal conditional functions: a dynamic theory of epistemic states. In W. L. Harper and B. Skyrms, editors, *Causation in Decision, Belief Change, and Statistics*, volume 2, pages 105–134. 1987.

[Williams, 1994] M. A. Williams. Transmutations of knowledge systems. In *Proceedings of the Fourth International Conference on the Principles of Knowledge Representation and Reasoning (KR'94)*, pages 619–629, 1994.

[Williams, 1995] M. A. Williams. Iterated theory base change: a computational model. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI'95)*, pages 1541 – 1550, 1995.

Answer Set Programming and Abductive Reasoning

Unifying Abductive Logic Programming and Stable Logic Programming

Marc Denecker
Département d'Informatique
Université Libre de Bruxelles
Boulevard du Thriomphe CP212
Brussels, Belgium

Abstract

Abductive Logic Programming (ALP) and Answer Set Programming (ASP) were developed independently and are defined in terms of quite different foundations and concepts. Yet, the existence of formally proven relationships and the often striking congruence between application programs in both formalisms suggest that, at a deeper level, both formalisms might be tighly related. In this talk, I will take an epistemological viewpoint to investigate and uncover this relationship.

Epistemological foundations of both formalisms can been defined in terms of two nonmonotonic reasoning principles, in particular autoepistemic/default reasoning on the one hand and iterated inductive definitions on the other hand. Both principles are fundamentally different, even contrary nonmonotonic principles, one being a *minimising knowledge* principle while the second being a *maximising knowledge* principle. In the talk, I will argue that Answer Set Programming with two negations is an instance of the first principle whereas Abductive Logic Programming and Stable Logic Programming are instances of the second principle.

Stable logic programming is thus presented as a fundamentally different logic than Answer Set Programming with two negations. This contradicts with the standard view in which Stable Logic Programming is viewed as a subformalism of ASP. I will back up my position with formal results showing that stable logic programming and ASP attribute fundamentally different modalities to the negation as failure operator and the rule operator.

On the other hand, this work unifies Stable Logic Programming and Abductive Logic Programming as syntactic variants of the same underlying logic. The impact on methodology will be discussed. At the computational level, Abductive Logic Programming and Stable Logic Programming provide different computational approaches for essentially the same type of reasoning problem.

During his invited talk at NMR2000, Victor Marek complained about the lack of relationship and the confused position of logic programming in the broader area of mathematical and philosophical logic. In this talk, a big step is made in clarifying the picture. At the same time, the talk addresses a number of fundamental logical ambiguities and long standing open problems about the meaning of language constructs such as negation as failure and rule operator in logic programming extensions.

Abduction, ASP and Open Logic Programs

P. A. Bonatti
Dipartimento di Tecnologie dell'Informazione
Università di Milano
bonatti@dti.unimi.it

Abstract

Open logic programs and open entailment have been recently proposed as an abstract framework for the verification of incomplete specifications based upon normal logic programs and the stable model semantics. There are obvious analogies between open predicates and abducible predicates. Their extension is not specified in the program. However, despite superficial similarities, there are features of open programs that have no immediate counterpart in the framework of abduction and viceversa. Similarly, open programs cannot be immediately simulated with answer set programming (ASP).

In this paper we start a thorough investigation of the relationships between open inference, abduction and ASP. We shall prove that open programs generalize the other two frameworks. Similarities and differences between the three frameworks will be analyzed formally. The generalized framework suggests interesting extensions of abduction under the generalized stable model semantics. In some cases, we will be able to reduce open inference to abduction and ASP, thereby characterizing the computational complexity of credulous and skeptical open inference for finite, function-free open programs. At the same time, the aforementioned reduction opens the way to new applications of abduction and ASP.

1 Introduction

Open logic programs and open entailment [Bonatti 2001] have been recently proposed as an abstract framework for the verification of incomplete specifications based upon normal logic programs and the stable model semantics.

An important example of incomplete specifications is given by compound security policies [Bonatti et al. 2000], where details such as the set of users and the formulation of certain subpolicies are tipically unknown at verification time. In this setting, it is interesting to verify whether the policy will necessarily satisfy privacy laws, all parties' requirements, etc.

Logic-based agents are a second important example. In IMPACT [Subrahmanian et al. 2000], agent programs must satisfy a property called *conflict freedom*. This property depends on a variable state that extends the agent program. The state is unknown at verification time, and can be regarded as a runtime extension of the agent program. Conflict freedom should hold for all possible such extensions.

Incomplete specifications are modelled by identifying a set of predicates, called *open predicates*, that are not completely defined in the program. Different forms of entailment capture what could possibly be true and what must necessarily be true, across a space of possible complete specifications of the open predicates.

There are obvious analogies between open predicates and abducible predicates. Their extension is not specified in the program. In the framework of abduction, a suitable definition of abducible predicates must be found as part of the reasoning task, while in the framework of open programs one may also quantify over all possible definitions of the open predicates (skeptical open inference). Moreover, abduction looks for simple definitions of the abducible predicates, consisting of ground facts, while the framework of open programs admits arbitrary definitions. It should be pointed out that arbitrary rules can be abduced by introducing new abducibles, but the point is whether the result-

ing abduction framework must necessarily be infinite when the set of abducible rules is.

Thus, despite superficial similarities, there are features of open programs that cannot be immediately reproduced in the framework of abduction and viceversa. Similarly, answer set programming (ASP) cannot trivially simulate open programs, because of the complex structure of some forms of open inference (called *mixed* inference), and because of the simultaneous treatment of diverse Herbrand domains, featured by open program semantics.

In this paper we start a thorough investigation of the relationships between open inference, abduction and ASP. We shall consider the abduction frameworks originated by [Eshghi and Kowalski 1989], refined in [Kakas and Mancarella 1990], and further studied – from a procedural standpoint – in [Satoh and Iwayama 1992]. The semantics is based upon *generalized stable models*, due to [Kakas and Mancarella 1990]. The main contributions of the paper are the following.

- We shall prove that open programs generalize both ASP and the generalized stable model semantics. A related benefit is that the applicability range of abduction and ASP is potentially extended to the intended applications of open programs (under some restrictions). As a second benefit, the relationships between the three frameworks open the way to a cross-fertilization of the proof procedures for abduction, ASP and open inference. Moreover, the generalized framework suggests some interesting extensions:
  - The generalized stable model semantics does not allow to abduce the existence of new individuals. Open logic programs provide such extension, previously supported only under 3-valued completion semantics [Denecker and De Schreye 1998] and stratified negation [Shanahan 1989]. Furthermore, open logic programs can express upper bounds on domain cardinality. None of the existing approaches handles such constraints.
  - The existing abductive procedures for the generalized stable model semantics do not support nonground inference. Open inference is a first step toward nonground abduction under the generalized stable model semantics. Moreover, open inference suggests new computation strategies, e.g., based on lemma generation.
  - The *mixed* inferences introduced in the open framework suggest a new form of abduction, that stands to standard abduction as *diagnosis as entailment* stands to *diagnosis as satisfaction*.
- In some cases, we will be able to reduce open inference to ASP. As a result, we characterize the complexity of two forms of open inference for finite and function-free open programs. Moreover, we show how ASP can be used to compute open inference in this special case. A complete embedding will be left as an open issue.
- The (partial) embedding of open programs into ASP makes it possible to use ASP engines such as Smodels [Niemelä and Simons 1997] for abductive reasoning under the open version of the generalized stable model semantics.

The basic definitions concerning abduction frameworks and open programs will be recalled in Section 2. Then we will study the relationships between open programs and abduction frameworks in Section 3. In Section 4, credulous and skeptical open inferences are embedded into standard credulous and skeptical inferences in ASP. The main results are summarized and discussed in Section 5.

2 Preliminaries

We assume the reader to be familiar with normal logic programs and the stable model semantics [Gelfond and Lifschitz 1988]. We say that a normal logic program is *consistent* if it has at least one stable model.

An *abduction framework* is a pair $\langle T, A \rangle$, where T is a normal logic program and A is a set of *abducible predicates*. Let $\mathsf{Abducibles}(T, A)$ be the set of all ground atoms $p(t_1, \ldots, t_n)$ such that $p \in A$ and t_i belongs to the Herbrand domain of T $(1 \leq i \leq n)$.

Definition 2.1 A *generalized stable model* of an abduction framework $\langle T, A \rangle$ is a stable model of $T \cup E$, for some $E \subseteq \mathsf{Abducibles}(T, A)$. ∎

An *open program* is a triple $\langle P, F, O \rangle$ where P is a normal logic program, F is a set of function and constant symbols not occurring in P, and O is a set of predicate symbols. A *completion*[1] of $\langle P, F, O \rangle$ is a normal logic program P' such that

[1] Note that this notion of completion has nothing to do with the standard notion of completion derived from Clark's work. We have not been able to identify an alternative term to express correctly the idea of complete predicate specification.

1. $P' \supseteq P$,
2. the constant and function symbols of P' occur in P or F,
3. if $r \in P' \setminus P$, then $head(r) \in O$.

The set of all the completions of $\langle P, F, O \rangle$ is denoted by $\mathsf{Comp}(P, F, O)$. There exist four kinds of *open inference*:

1. (Credulous open inference) $\langle P, F, O \rangle \models^c \Psi$ iff for some $P' \in \mathsf{Comp}(P, F, O)$, P' credulously entails Ψ.
2. (Skeptical open inference) $\langle P, F, O \rangle \models^s \Psi$ iff for all $P' \in \mathsf{Comp}(P, F, O)$, P' skeptically entails Ψ.
3. (Mixed open inference I) $\langle P, F, O \rangle \models^{cs} \Psi$ iff for some consistent $P' \in \mathsf{Comp}(P, F, O)$, P' skeptically entails Ψ.
4. (Mixed open inference II) $\langle P, F, O \rangle \models^{sc} \Psi$ iff for each consistent $P' \in \mathsf{Comp}(P, F, O)$, P' credulously entails Ψ.

The four open entailments are pairwise dual and form a diamond-shaped lattice [Bonatti 2001].

Proposition 2.2 *(Duality) For all open programs Ω and all sentences Ψ,*

1. $\Omega \models^c \Psi$ *iff* $\Omega \not\models^s \neg\Psi$;
2. $\Omega \models^{sc} \Psi$ *iff* $\Omega \not\models^{cs} \neg\Psi$.

Proposition 2.3 *(Entailment lattice) Suppose there exists a consistent $P' \in \mathsf{Comp}(\Omega)$. Then, for all sentences Ψ,*

1. $\Omega \models^s \Psi$ *implies* $\Omega \models^{cs} \Psi$ *and* $\Omega \models^{sc} \Psi$;
2. $\Omega \models^{cs} \Psi$ *implies* $\Omega \models^c \Psi$;
3. $\Omega \models^{sc} \Psi$ *implies* $\Omega \models^c \Psi$.

3 Open programs and abduction

Abducible predicates are similar to open predicates, as their extension is not specified in the program. However, open predicates can be given arbitrarily complex definitions (including all sorts of cycles, possibly involving non-open predicates), while abducible predicates are always defined by sets of ground facts (cf. the clause $E \subseteq \mathsf{Abducibles}(T, A)$ in Definition 2.1).

Therefore, in order to relate the two approaches, we need a representation lemma showing that a similar restriction can be posed on the completions of open predicates, without loss of generality.

Lemma 3.1 *If M is a stable model of $P' \in \mathsf{Comp}(P, F, O)$, then M is a stable model of some $P'' \in \mathsf{Comp}(P, F, O)$ such that $P'' \setminus P$ is a set of ground facts.*

Proof.(Sketch) Let $P'' = P \cup \{p(\vec{t}) \in M \mid p \in O\}$. It can be verified that M is also a stable model of P''. Moreover, $P'' \setminus P$ is a set of ground facts by definition. ∎

A rule $r = H \leftarrow B$ could also be abduced by introducing a new abducible n_r (that plays the role of r's name), and inserting $H \leftarrow B, n_r$ in the program (cf. [Poole 1988]). Clearly, if arbitrary rules are to be abduced (as it happens in open programs), then this method leads necessarily to an infinite abduction framework (more precisely, both the program and the set of abducibles are infinite). Lemma 3.1 improves rule abduction by showing that in fact one needs not add any new symbols and rules, provided that no constraint is posed on the space of abducible rules.

Now the basic correspondence between abduction and open programs follows easily.

Theorem 3.2 *M is a generalized stable model of an abductive framework $\langle T, A \rangle$ iff M is a stable model of some $P \in \mathsf{Comp}(T, \emptyset, A)$.*

The relationships between abductive frameworks and open programs suggest two natural extensions of abductive reasoning under the generalized stable model semantics. First note that the set F of possible function symbols for the completions is empty in Theorem 3.2. By allowing a nonempty F, the generalized stable model semantics can be given the ability of assuming the existence of new individuals and functions (as in [Denecker and De Schreye 1998] and [Shanahan 1989]). Typically, in the framework of reasoning about action and change, this feature is applied to explain a sequence of events with unknown actions. Abductive frameworks can be extended accordingly, by a straightforward generalization of $\mathsf{Abducibles}(\cdot, \cdot)$.

Definition 3.3 For each abduction framework $\langle T, A \rangle$, let $\mathsf{Abducibles}^o(T, A)$ be a set of ground atoms $p(t_1, \ldots, t_n)$ such that $p \in A$ and t_i is a term built from the function and constant symbols of T, plus a denumerable set Sk of (skolem) constants not occurring in T. $\mathsf{Abducibles}^o(T, A)$ will be called a set

of *open abducibles* of the abduction framework $\langle T, A \rangle$ (w.r.t. the set of skolem constants Sk). ∎

Clearly, the choice of Sk is irrelevant, as long as it does not intersect the vocabulary of T.

Definition 3.4 An *open generalized stable model* of an abduction framework $\langle T, A \rangle$ is a stable model of $T \cup E$, for some $E \subseteq \mathsf{Abducibles}^o(T, A)$. ∎

By analogy with the previous theorem we have:

Proposition 3.5 *M is an open generalized stable model of an abductive framework $\langle T, A \rangle$ iff M is a stable model of some $P \in \mathsf{Comp}(T, Sk, A)$, where Sk is the set of skolem constants adopted in* $\mathsf{Abducibles}^o(T, A)$.

The second extension concerns inference modalities. Mixed inference of type I suggests a new form of abduction that can be understood by analogy with the two main forms of diagnosis, based on satisfaction and entailment, respectively. More precisely, diagnostic reasoning can be carried out either by looking for a model of the domain theory T where the observations Q are *satisfied*, or by finding a set of explanations E such that $T \cup E$ *entails* Q. The two kinds of diagnosis can be regarded as the analogues of standard generalized stable model computation (which is equivalent to credulous open reasoning, by Theorem 3.2) and mixed open inference of type I, respectively. The latter can be reformulated for abductive frameworks as follows.

Definition 3.6 A sentence Q is a *generalized skeptical consequence* of an abductive framework $\langle T, A \rangle$ if there exists $E \subseteq \mathsf{Abducibles}(T, A)$ such that Q holds in *all* the stable models of $T \cup E$.

Similarly, Q is an *open generalized skeptical consequence* of an abductive framework $\langle T, A \rangle$ if there exists $E \subseteq \mathsf{Abducibles}^o(T, A)$ such that Q holds in *all* the stable models of $T \cup E$. ∎

Open programs improve standard abduction in one more respect: open predicates can be *partially specified* (cf. [Bonatti 2001], Example 14). The same effect can be obtained by adding one more abducible predicate for each partially defined predicate (cf. [Kakas et al. 1992]). A direct approach (such as the open framework) that introduces no auxiliary symbols may be considered more elegant.

4 Open programs and ASP

Open programs obviously generalize answer set programming. If $F = O = \emptyset$, then $\mathsf{Comp}(P, F, O) = \{P\}$, and the four entailment relations collapse to the standard credulous and skeptical entailment of the stable model semantics.

Proposition 4.1 *Let P be a normal logic program. A sentence Q is a credulous (resp. skeptical) consequence of P iff $\langle P, \emptyset, \emptyset \rangle \models^c Q$ (resp. $\langle P, \emptyset, \emptyset \rangle \models^s Q$). Moreover, if P is consistent, then $\langle P, \emptyset, \emptyset \rangle \models^c Q$ is equivalent to $\langle P, \emptyset, \emptyset \rangle \models^{sc} Q$, and $\langle P, \emptyset, \emptyset \rangle \models^s Q$ is equivalent to $\langle P, \emptyset, \emptyset \rangle \models^{cs} Q$.*

In the rest of this section, we focus on the opposite embedding.

Given Lemma 3.1, embedding open programs into ASP may seem trivial. Apparently, the possible extensions of open predicates could all be generated by suitable cyclic definitions, by analogy with the embedding of generalized stable model semantics into (standard) stable model semantics [Satoh and Iwayama 1991]. However, handling the different Herbrand domains of the completions is not so trivial.

Example 4.2 Let $P = \{p(a),\ q \leftarrow \neg p(X)\}$ and $F = \{b\}$. Suppose that $p \notin O$ and $q \notin O$. The stable models of the completions $P' \in \mathsf{Comp}(P, F, O)$ satisfy q iff b occurs in P'. As a consequence, neither q nor $\neg q$ are skeptical open consequences of $\langle P, F, O \rangle$. Now consider two naive attempts at embedding $\langle P, F, O \rangle$ into ASP:

$$P_1 = P \cup \{p(a) \leftarrow \neg \bar{p}(a)\} \cup \{\bar{p}(a) \leftarrow \neg p(a)\}$$
$$P_2 = P_1 \cup \{p(b) \leftarrow \neg \bar{p}(b)\} \cup \{\bar{p}(b) \leftarrow \neg p(b)\}$$

(where $\bar{p}$ is a new predicate symbol). Note that $\neg q$ is a skeptical consequence of P_1 while q is a skeptical consequence of P_2, so none of the two programs is sound with respect to open inference. ∎

A faithful (partial) embedding of open programs into ASP can be obtained by modeling the Herbrand domains explicitly (a device used also for modeling quantification in Kripke structures where different worlds have different domains).

Definition 4.3 Let $\langle P, F, O \rangle$ be an open program. Define the corresponding normal logic program $\Pi(P, F, O)$ as follows. For each predicate symbol $p \in O$, introduce a new distinct predicate symbol $\bar{p}$. Moreover, let U, S and $\bar{S}$ be three new predicate symbols distinct from the symbols $\bar{p}$. $\Pi(P, F, O)$ consists of the following rules, for all rules $H \leftarrow Body$ in P, for all n-ary function symbols f occurring in P or F, and

for all $p \in O$:

$$H \leftarrow Body, U(x_1), \ldots, U(x_n) \qquad \text{where the } x_i\text{s are the variables of } H \leftarrow Body$$
$$U(f(x_1, \ldots, x_n)) \leftarrow U(x_1), \ldots, U(x_n), S(f)$$

$$S(f) \qquad \text{if } f \notin F$$
$$S(f) \leftarrow \neg\bar{S}(f) \qquad \text{if } f \in F$$
$$\bar{S}(f) \leftarrow \neg S(f) \qquad \text{if } f \in F$$

$$p(x_1, \ldots, x_n) \leftarrow \neg\bar{p}(x_1, \ldots, x_n), U(x_1), \ldots, U(x_n)$$
$$\bar{p}(x_1, \ldots, x_n) \leftarrow \neg p(x_1, \ldots, x_n), U(x_1), \ldots, U(x_n)$$

∎

Intuitively, S and $\bar{S}$ select a vocabulary from P and F, U captures the corresponding ground terms (i.e., the Herbrand domain of some completion), and the last rules generate an arbitrary set of ground open atoms.

Remark 4.4 Strictly speaking, in the above definition the atoms $S(f)$ and $\bar{S}(f)$ should be replaced by $S(\hat{f})$ and $\bar{S}(\hat{f})$, where $\hat{f}$ is a *name* for f. For simplicity, here we use a Prolog-like, relaxed syntax.

Example 4.5 Let P and F be as in Example 4.2, let $O = \{r\}$. Then $\Pi(P, F, O)$ consists of the following rules:

$$p(a)$$
$$q \leftarrow \neg p(X), U(X)$$
$$U(a) \leftarrow S(a)$$
$$U(b) \leftarrow S(b)$$
$$S(a)$$
$$S(b) \leftarrow \neg\bar{S}(b)$$
$$\bar{S}(b) \leftarrow \neg S(b)$$
$$r(X) \leftarrow \neg\bar{r}(X), U(X)$$
$$\bar{r}(X) \leftarrow \neg r(X), U(X).$$

∎

The following theorem proves that the above embedding is correct. Let $M|_{P,F,O} = \{\, p(t_1, \ldots, t_n) \in M \mid p \text{ occurs in } \langle P, F, O \rangle \text{ and } U(t_i) \in M \ (1 \leq i \leq n) \,\}$.

Theorem 4.6 *M is a stable model of $\Pi(P, F, O)$ iff there exist $P' \in \mathsf{Comp}(P, F, O)$ and a stable model M' of P' such that $M|_{P,F,O} = M'$.*

Corollary 4.7 *For all sentences Q with no occurrences of the new predicates $\bar{p}$ ($p \in O$), U, S and $\bar{S}$,*

1. *$\langle P, F, O \rangle \models^c Q$ iff Q is a credulous consequence of $\Pi(P, F, O)$.*
2. *$\langle P, F, O \rangle \models^s Q$ iff Q is a skeptical consequence of $\Pi(P, F, O)$.*

Note that $\Pi(P, F, O)$ may contain function symbols (then its consequences may be undecidable). If either F or O are infinite then $\Pi(P, F, O)$ is infinite, otherwise $\Pi(P, F, O)$ is finite and it can be computed in polynomial time (w.r.t. $|P \cup F \cup O|$). The following corollary immediately follows.

Corollary 4.8 *Suppose that P, F and O are finite and contain only 0-ary function symbols.*

1. *If P is ground, then credulous and skeptical open inferences are NP-complete and coNP-complete, respectively.*
2. *If P is not ground, then credulous and skeptical open inferences are NEXPTIME-complete and coNEXPTIME-complete, respectively.*

Corollary 4.7 and Corollary 4.8 cannot be easily extended to mixed open inference. The above translation technique (that basically derives from Lemma 3.1) captures faithfully the credulous consequences of the completions, but apparently it does not scale to their skeptical consequences. Currently, we do not know whether mixed inference can be polynomially embedded into ASP.

The results of the previous section can be combined with the above translation to obtain an embedding of abduction into ASP.

Corollary 4.9 *For all interpretations M,*

1. *M is a generalized stable model of an abduction framework $\langle T, A \rangle$ iff $M = M'|_{T,\emptyset,A}$, for some stable model M' of $\Pi(T, \emptyset, A)$.*
2. *M is an open generalized stable model of an abduction framework $\langle T, A \rangle$ iff $M = M'|_{T,Sk,A}$, for some stable model M' of $\Pi(T, Sk, A)$ (where Sk is the set of skolem constants adopted in $\mathsf{Abducibles}^o(T, A)$).*

The translation $\Pi(T, \emptyset, A)$ is slightly more redundant than the translation introduced in [Satoh and Iwayama 1991], but a trivial unfolding process yields an equivalent embedding.

Example 4.10 Consider the abduction framework $\langle P, O \rangle$, where P and O are specified as in Example 4.5. $\Pi(P, \emptyset, O)$ is

$$p(a)$$
$$q \leftarrow \neg p(X), U(X)$$
$$U(a) \leftarrow S(a)$$
$$S(a)$$
$$r(X) \leftarrow \neg\bar{r}(X), U(X)$$
$$\bar{r}(X) \leftarrow \neg r(X), U(X).$$

Unfolding yields the (expected) simplified program

$$
\begin{array}{l}
p(a) \\
q \leftarrow \neg p(a) \\
r(a) \leftarrow \neg \bar{r}(a) \\
\bar{r}(a) \leftarrow \neg r(a) \,.
\end{array}
$$

∎

The translation $\Pi(T, Sk, A)$ extends the approach by [Satoh and Iwayama 1991] with the ability of abducing new individuals. Unfortunately, $\Pi(T, Sk, A)$ is infinite if Sk is infinite. Therefore, in the absence of any upper bound on the domain's cardinality, the existing credulous engines for ASP can only be applied to a finite approximation $\Pi(T, Sk', A)$ of the original problem, where Sk' is a finite subset of Sk. Clearly, this approximation yields all and only the abductions that postulate the existence of at most $|Sk'|$ individuals, besides those explicitly mentioned in the domain knowledge.

Example 4.11 Let

$$
\begin{array}{rcl}
T & = & \{\, p(a),\ q \leftarrow r(X), \neg p(X) \,\} \\
A & = & \{\, r \,\}\,.
\end{array}
$$

No generalized stable model of $\langle T, A \rangle$ satisfies q, whereas if $Sk = \{s_0, s_1, \ldots, s_i, \ldots\}$, then $\langle T, A \rangle$ has an open generalized stable model

$$\{p(a), q, \} \cup \{r(s_i) \mid i \in I\}$$

for each set of natural numbers I. Only one individual needs to be abduced in order to explain q. Accordingly, the program $\Pi(T, \{s_0\}, A)$ consisting of the rules

$$
\begin{array}{l}
p(a) \\
q \leftarrow r(X), \neg p(X), U(X) \\
U(a) \leftarrow S(a) \\
U(s_0) \leftarrow S(s_0) \\
S(a) \\
S(s_0) \leftarrow \neg \bar{S}(s_0) \\
\bar{S}(s_0) \leftarrow \neg S(s_0) \\
r(X) \leftarrow \neg \bar{r}(X), U(X) \\
\bar{r}(X) \leftarrow \neg r(X), U(X)
\end{array}
$$

has a stable model $\{p(a), q, r(s_0)\}$, that "explains" q with $\{r(s_0)\}$. In this case, the bound on the open domain does not cause any significant loss of information, in the sense that all the alternative minimal explanations are isomorphic to $\{r(s_0)\}$. ∎

5 Summary and Conclusions

Open programs constitute a simple unifying framework that generalizes both abduction frameworks (under the generalized stable model semantics) and answer set programming, as shown by Theorem 3.2 and Proposition 4.1. In particular, any abduction framework $\langle T, A \rangle$ corresponds to the open program $\langle T, \emptyset, A \rangle$, while each normal logic program P is captured by the degenerate open program $\langle P, \emptyset, \emptyset \rangle$.

The reduction of the generalized stable model semantics to open inference highlights similarities and differences between the two frameworks. In particular, it shows that the only real difference lies in the treatment of the domains, that are fixed in abduction frameworks, and variable in the open framework (cf. the discussion after Theorem 3.2). On the contrary, the different kinds of predicate specifications considered by the two frameworks (ground facts vs. arbitrary definitions) have no influence, as they select the same set of models (Lemma 3.1). This result implies also that in the absence of constraints on the space of abducible rules, no new abducible predicates and rules need to be introduced in order to abduce complex sentences (as opposed to the standard formulation [Poole 1988]).

We have applied the open framework to give the generalized stable model semantics the ability of abducing new individuals (Definition 3.4). Then we extended the embedding of abduction into ASP accordingly, using the more general result for open programs (Corollary 4.9). As a result, ASP engines such as Smodels can be applied to abductive reasoning over *bounded* open domains (cf. the discussion after Corollary 4.9). Because of the bound on the domain size, ASP can only approximate abduction with unbounded open domains. We are currently generalizing *finitary programs* [Bonatti 2001b] to achieve semidecidable, *exact* abduction with unbounded open domains.

Note, however, that the bound on domain size is not always a restriction. In some cases, the bound may be part of the domain knowledge. The existing calculi that support open domains do not express nor reason about such restrictions [Denecker and De Schreye 1998, Shanahan 1989].

The open framework suggests a new form of abduction analogous to *diagnosis-as-entailment*, called *generalized skeptical consequences* (Definition 3.6).

Our results improve the understanding of open inference. From the embedding into ASP we derived the complexity of skeptical and credulous open inference in the finite case, and in the absence of proper functions (Corollary 4.8). In the other cases, the translation may be infinite or contain function symbols, and the only known automated deduction method is the skeptical open resolution calculus introduced in [Bonatti 2001].

This calculus derives sentences that hold no matter how the open program is completed. It may be in-

teresting to investigate the use of such skeptical inferences to pre-compute lemmata, in order to speed-up the computation of explanations.

Moreover, the skeptical open calculus is nonground, and may return nonground answers. This may be a starting point for nonground abduction under the open generalized stable model semantics.

On the other hand, Theorem 3.2 and Corollary 4.7 open the way to the application of abduction procedures and ASP engines to open inference problems, at least under some cardinality and arity restrictions. In particular, abduction and ASP might be applied to static verification problems for various kinds of incomplete specifications.

Currently it is not clear whether the mixed forms of open inference can be embedded into ASP and abduction frameworks. We conjecture that no polynomial reduction based on normal logic programs exists. It should be verified whether the calculus for mixed inference illustrated in [Bonatti 2001] can be immediately applied to the generalized skeptical consequences introduced in Definition 3.6.

Acknowledgements

The author is grateful to the anonymous referees for their comments, that led to significant improvements in the paper.

The work reported in this paper was partially supported by the European Community within the Fifth (EC) Framework Programme under contract IST-1999-11791 - FASTER project.

References

[Bonatti 2001] P. A. Bonatti. Reasoning with open programs. *Proc. of LPNMR'01*, pp. 147-159, LNAI 2173, Springer Verlag, 2001.

[Bonatti 2001b] P. A. Bonatti. Reasoning with infinite stable models. *Proc. of IJCAI*, Morgan Kaufmann, 2001.

[Bonatti et al. 2000] P.A. Bonatti, S. De Capitani Di Vimercate, P. Samarati. An algebra for composing access control policies. *ACM Transactions on Information and System Security, 5(1), 2002*, to appear. Preliminary version in Proc. of the 7th ACM Conference on Computer and Communication Security, CCS'2000, Atene, 2000.

[Denecker and De Schreye 1998] D. Denecker, D. De Schreye. SLDNFA: An abductive procedure for abductive logic programs. *The Journal of Logic Programming*, 34(2):111-167, 1998.

[Eshghi and Kowalski 1989] K. Eshghi, R.A. Kowalski. Abduction compared with negation as failure. In *Proc. of the 6th Int.l Conf. on Logic Programming*, MIT Press, 1989.

[Gelfond and Lifschitz 1988] M. Gelfond, V. Lifschitz. The stable model semantics for logic programming. In *Proc. of the 5th ICLP*, pp.1070-1080, MIT Press, 1988.

[Kakas et al. 1992] A.C. Kakas, R.A. Kowalski, F. Toni. Abductive Logic Programming. *Journal of Logic and Computation*, 2(6):719-770, 1992.

[Kakas and Mancarella 1990] A.C. Kakas, P. Mancarella. Generalized Stable Models: a semantics for abduction. *Proc. of ECAI'90*, pp.385-391, 1990.

[Niemelä and Simons 1997] I. Niemelä, P. Simons. Smodels - an implementation of the stable model and well-founded semantics for normal LP. In J. Dix, U. Furbach, A. Nerode (eds.), *Logic Programming and Nonmonotonic Reasoning: 4th international conference, LPNMR'97*, LNAI 1265, Springer Verlag, Berlin, 1997.

[Poole 1988] D. Poole. A logical framework for default reasoning. *Artificial Intelligence*, 36(1):27-47, 1988.

[Satoh and Iwayama 1992] K. Satoh, N. Iwayama. A query evaluation method for abductive logic programming. In *Proc. of the Joint Int.l Conf. and Symposium on Logic Programming* (JICSLP'92), 671-685, MIT Press, 1992.

[Satoh and Iwayama 1991] K. Satoh, N. Iwayama. Computing abduction by using the TMS. In *Proc. of the Int.l Conf. on Logic Programming* (JICSLP'92), 504-ff, MIT Press, 1991.

[Shanahan 1989] M. Shanahan. Prediction is deduction but explanation is abduction. In *Proc. of IJCAI'89*, 1055-ff, 1989.

[Subrahmanian et al. 2000] V.S. Subrahmanian, P.A. Bonatti, J. Dix, T. Eiter, S. Kraus, F. Ozcan, R. Ross. *Heterogeneous Active Agents.* MIT Press, 2000.

Speculative Computation and Abduction for an Autonomous Agent

Ken Satoh
National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, 101-8430, Japan

Abstract

In this paper, we propose an agent architecture for a combination of speculative computation and abduction. *Speculative computation* is a tentative computation when complete information for performing computation is not obtained. We use a default value to complement such incomplete information. Unlike usual default reasoning, the real value for the information can be obtained during the computation and the computation can be revised on the fly. In the previous work, we applied this technique to handling distributed problem solving under incomplete communication environments in the context of multi-agent systems and proposed correct procedures in abductive logic programming in terms of perfect model semantics. In the previous work, however, we regard assumptions as defaults and therefore, we could not perform a hypothetical reasoning which is the original usage of abduction. In this paper, we extend our framework so that speculative computation and abduction can be both performed. As a result, our procedure becomes an extension of the abductive procedure developed by Kakas and Mancarella augmented by dynamic belief revision mechanism about outside world.

1 Introduction

In this paper, we consider a method of combining a dynamically revisable computation and abductive reasoning. Consider the following example.

- A robot would like to go from a room to another room through intermediate rooms by making a plan before it takes an action.
- There is a door between rooms; and if the door is closed, the robot cannot get in the room of the destination.
- There is a lion which destroys a robot if they are in the same room.
- There is a sensor to detect which door between the rooms is open or not, and an image analysis module[1] to recognize a lion in a room.
- The information from the sensor and the recognizer will be obtained during the planning.
- We suppose that a robot has default values about the information sent from the sensor and the recognizer before starting planning.

The robot sends a request to the sensor and the recognizer for the necessary information to make a plan. Simultaneously, the robot can also make a tentative plan based on his default assumptions. This may be successful even if no information is returned from the sensor and the recognizer since the default assumptions are normally true under the usual circumstance. Suppose that the tentative planning takes a time and some information is returned from the sensor and the recognizer during planning. If the information is consistent with the default assumptions, then it does not cause any problem and the robot can continue the planning. If the robot waited for the returned information, the robot would not start the planning before this point. Therefore, making a tentative plan beforehand is effective in this case. However, suppose that the obtained information is contradictory with the default assumption. Then, it is no use to continue making a tentative

[1] We call the image analysis module a recognizer in this paper.

plan which does not reflect the outside world. In this case, the robot must revise the computation dynamically so that the obtained information is adapted. In this situation making a tentative plan is not effective. Thus, we call this tentative computation a *speculative computation*. Note that this kind of dynamic revision can occur many times as new information obtained from sensors since the world is dynamically changed. Therefore, we need a mechanism of iterative dynamic revisions as well.

We applied the idea of speculative computation in order to handle distributed problem solving under incomplete communication environments in the context of multi-agent systems and proposed correct procedures in abductive logic programming in terms of perfect model semantics (Satoh00; Satoh02). The usage for abductive logic programming in the work is default reasoning and so it is different from ordinary abduction treated in (Kakas90). The ordinary abduction is useful not only for default reasoning but also other applications (Kakas98) such as the above example of planning. In this paper, we extend our framework so that the ordinary abduction can be performed for a more powerful reasoning mechanism. We distinguish the assumption returned from the outside worlds and the abducibles used inside the agent's reasoning. As a result, our method is an extension of the procedure of Kakas and Mancarella (Kakas90) so that dynamic revision of computation is possible.

2 Framework of Speculative Computation in Master-Slave System

A *speculative framework for a master-slave system* SF_{MS} is the following tuple $\langle \Sigma, \mathcal{E}, \Delta, \mathcal{A}, \mathcal{P}, \mathcal{I} \rangle$ where

- Σ is a finite set of constants. We call an element in Σ a *slave agent identifier.*
- $\mathcal{E}$ is a set of predicates called *external predicates.* When Q is a literal with an external predicate and S is a slave agent identifier, we call $Q@S$ an *askable literal.* We define $\sim (Q@S)$ as $(\sim Q)@S$.
- Δ is a set of ground askable literals satisfying the following condition:

  Δ does not contain both of $p(t_1, ..., t_n)@S$ and $\sim p(t_1, ..., t_n)@S$ simultaneously[2] .

  We call Δ a *default answer set.*
- $\mathcal{A}$ is a set of predicates called *abducible predicates.* When Q is a literal with an *abducible predicate*, Q is called *abducible.*
- $\mathcal{P}$ is a set of rules of the form:

  $$H \leftarrow B_1, B_2, ..., B_n.$$

  where
  - H is a positive ordinary literal[3] .
  - each of $B_1, ..., B_n$ is an ordinary literal or an askable literal or an abducible.

  Let R be a rule of the form $H \leftarrow B_1, B_2, ..., B_n$. We call H a head denoted as $head(R)$ and $B_1, ..., B_n$ a body denoted as $body(R)$. $head(R)$ is non-empty, but if there is no literal in the body, $body(R) = \emptyset$.
- $\mathcal{I}$ is a set of integrity constraints of the form:

  $$\bot \leftarrow B_1, B_2, ..., B_n.$$

  where
  - $\bot$ is a special symbol for contradiction.
  - each of $B_1, ..., B_n$ is is an ordinary literal or an askable literal or an abducible, but at least one of them is an askable literal or an abducible.

□

Intuitively, an askable literal has two kinds of meaning.

1. An askable literal $Q@S$ in a rule in $\mathcal{P}$ represents a question a master agent asks to a slave agent S.
2. An askable literal in Δ represents the default truth value; if $p(t_1, ..., t_n)@S \in \Delta$, $p(t_1, ..., t_n)$ is normally true for a question to an agent S, and if $\sim p(t_1, ..., t_n)@S \in \Delta$, $p(t_1, ..., t_n)$ is normally false for a question to an agent S.

Example 1 *The example of robot planning in the Introduction is represented as the following speculative framework* $SF_{MS} = \langle \Sigma, \mathcal{E}, \Delta, \mathcal{A}, \mathcal{P}, \mathcal{I} \rangle$. *Below,* $open(a, b)$ *means that a door from the room* a *to the room* b *is open.* $lion_in(a)$ *means that a lion is in the room* a. *We have default information that every door*

[2] Note that we can have different default values for different agents. For example, $p(t_1, ..., t_n)@S_1$ and $\sim p(t_1, ..., t_n)@S_2$ are possible for S_1 and S_2. Moreover, there might be a case that neither $p(t_1, ..., t_n)@S$ nor $\sim p(t_1, ..., t_n)@S$ is in Δ.

[3] An ordinary literal is a literal which is neither an askable literal nor an abducible.

is open and a lion is usually at the room 3. The set of rules is used to create a plan to go from a room to another room. However, there is an integrity constraint saying that there is no lion at the room of the destination to avoid being destroyed.

- $\Sigma = \{sensor, recognizer\}$
- $\mathcal{E} = \{open, lion_in\}$
- $\Delta = \{open(1,2)@sensor, open(1,3)@sensor,$
  $open(2,4)@sensor, open(3,4)@sensor,$
  $\sim lion_in(2)@recognizer,$
  $lion_in(3)@recognizer,$
  $\sim lion_in(4)@recognizer\}$
- $\mathcal{A} = \{go_to_room^{\dagger}\}$[4]
- $\mathcal{P}$ *is the following set of rules*[5] :

  $plan(F,F) \leftarrow .$

  $plan(X,F) \leftarrow$

  $door(X,Y), open(X,Y)@sensor,$

  $go_to_room^{\dagger}(X,Y), plan(Y,F).$

  $door(1,2) \leftarrow .$

  $door(1,3) \leftarrow .$

  $door(2,4) \leftarrow .$

  $door(3,4) \leftarrow .$
- $\mathcal{I}$ *is the following set of the integrity constraint:*

  $\bot \leftarrow$

  $go_to_room^{\dagger}(X,Y),$

  $lion_in(Y)@recognizer.$

  □

In the above representation, every door is assumed to be open and a lion is assumed to be in the room 3 by default.

3 An Implementation of Speculative Computation

In this section, we show an implementation of speculative computation. The execution of speculative framework is based on two phases, a *process reduction phase* and a *fact arrival phase*. The process reduction phase is a normal execution of a program in a master agent and the fact arrival phase is an interruption phase when an answer arrives from a slave agent.

Intuitively, processes are created when a choice point of computation is encountered like case splitting. A process terminates successfully if all the computation is done and the used defaults have not been contradictory with the current belief state.

In the process reduction phase, we reduce an active process into a new process. Reduction for an ordinary literal or an abducible corresponds with treatment of an ordinary literal or an abducible in *abductive derivation* of (Kakas90). Moreover, we use an extended literal in our procedure which corresponds with *consistency derivation* of (Kakas90) so that consistency derivation can be interleaved with abductive derivation whereas such interleaving is not implemented in (Kakas90).

The major difference between our procedure and that of (Kakas90) is treatment of askable literals. We assume that an agent has an belief set which initially defines a default of the outside information and is changed according to the information coming from outside world. When a process needs to reduce an askable literal, the literal is asked to the designated agent. Then, the literal is assumed and if it is in the current belief set, we can continue the process, otherwise we must suspend the process.

When an answer confirming a default comes from a slave agent, we just continue the current process. This is an effect of speculative computation since at this point, we have already executed a process without waiting. On the other hand, when an answer denying the current default comes from a slave agent, we must update the current belief state about the outside world and the current process must be suspended if the process uses the default which is contradictory with the new belief.

3.1 Preliminary Definitions

We define the following concepts for a proof procedure.

Definition 1 *An* extended literal *is either a literal or an expression of the form* $fail(\{L_1, ..., L_n\})$ *where* L_i *is a literal.* □

$fail(\{L_1, ..., L_n\})$ is used to show that there is no proof for one of L_i and corresponds with consistency derivation in (Kakas90).

Definition 2 *A* process *is the following tuple* $\langle GS, OD, IA, ANS \rangle$ *which consists of*

[4] We put † for an abducible predicate to distinguish from non-abducible predicate.

[5] A string beginning with an upper case letter represents a variable and a string beginning with a lower case letter represents a constant.

- *GS: A set of extended literals to be proved called* a goal set.
- *OD: A set of askable literals called* outside defaults.
- *IA: A set of negative literals or abducibles called* inside assumptions.
- *ANS: A set of instantiations of variables in the initial query.* □

Each process represents an alternative way of computation. The intuitive meaning of the above objects is as follows:

- GS expresses the current status of an alternative of computation.
- OD is a set of assumed information about outside world during a process.
- IA is a set of abducibles which has been assumed during a process.
- ANS gives an answer for variables in the initial query.

We use the following three sets for process reduction.

Definition 3

- A process set PS *is a set of processes.*
- *A set of* already asked queries AAQ *is a set of askable literals.*
- A current belief state CBS *is a set of askable literals.* □

PS is a set of processes which express all the alternative computations considered so far. AAQ is used to avoid from asking redundant questions to slave agents, and CBS is the current belief state which expresses the agent's belief of the current status of the outside world.

We define an *active* process and a *suspended* process as follows.

Definition 4

Let $\langle GS, OD, IA, ANS\rangle$ *be a process and* CBS *be a current belief state. A process is* active w.r.t. CBS *if* $OD \subseteq CBS$ *and a process is* suspended w.r.t. CBS *otherwise.* □

The intuitive meaning of active process is a process whose set of assumed information about outside world is consistent with the current belief set. It is because the current belief set is *very likely to reflect the outside world* and so the process using a subset of the current belief set as assumption is supposed to make a plan being consistent with the outside world.

3.2 Process Reduction Phase

Now, we give the precise description of the process reduction phase and the fact arrival phase. In the following description, we specify changed PS, AAQ, CBS as $NewPS$, $NewAAQ$, $NewCBS$; otherwise each PS, AAQ, CBS is unchanged.

Initial Step: Let GS be an initial goal set. We give $\langle GS, \emptyset, \emptyset, ANS\rangle$ to the proof procedure where ANS is a set of variables in GS. That is, $PS = \{\langle GS, \emptyset, \emptyset, ANS\rangle\}$. And let $AAQ = \emptyset$ and $CBS = \Delta$

Iteration Step: Do the following.

Case 1 If there is an active process $\langle \emptyset, OD, IA, ANS\rangle$ w.r.t. CBS in PS, then terminate the process by returning outside defaults OD, inside assumptions IA, and instantiation for variables ANS.

Case 2 If there is no active process then terminate the process by reporting a failure of the goal.

Case 3 Select an active process $\langle GS, OD, IA, ANS\rangle$ w.r.t. CBS from PS and select an extended literal L in GS. Let $PS' = PS - \{\langle GS, OD, IA, ANS\rangle\}$ and $GS' = GS - \{L\}$.

For the selected extended literal L, do the following.

Case 3.1 If L is a positive ordinary literal,
$NewPS = PS'$
$\cup \{\langle(\{body(R)\} \cup GS')\theta, OD, IA, ANS\theta\rangle |$
$\exists R \in \mathcal{P}$ and $\exists$most general unifier(mgu) θ
s.t. $head(R)\theta = L\theta\}$

Case 3.2 If L is a ground negative ordinary literal or a ground abducible then do the following:

Case 3.2.1 If $L \in IA$ then
$NewPS = PS' \cup \{\langle GS', OD, IA, ANS\rangle\}$

Case 3.2.2 If $\overline{L}$[6] $\in IA$ then
$NewPS = PS'$

Case 3.2.3 If $L \notin IA$ then
$NewPS = PS'$
$\cup\{\langle NewGS, OD, IA \cup \{L\}, ANS\rangle\}$

[6] $\overline{L}$ is a complement of a literal L.

where $NewGS =$
$\{fail(BS) | BS \in resolvent(L, \mathcal{P} \cup \mathcal{I})\} \cup GS'$
and $resolvent(L, T)$ is defined as follows:
- if L is a ground negative ordinary literal,
$resolvent(L, T) =$
$\{\{L_1\theta, ..., L_k\theta\} | H \leftarrow L_1, ..., L_k \in T$ s.t.
$\overline{L} = H\theta$ by a ground substitution $\theta\}$.
- if L is a ground abducible,
$resolvent(L, T) =$
$\{\{L_1\theta, ..., L_{i-1}\theta, L_{i+1}\theta, ..., L_k\theta\} |$
$\bot \leftarrow L_1, ..., L_k \in T$ s.t.
$L = L_i\theta$ by a ground substitution $\theta\}$.

Case 3.3 If L is $fail(BS)$, then
- if $BS = \emptyset$, $NewPS = PS'$.
- if $BS \neq \emptyset$ then do the following:
  1. Select B from BS
  and let $BS' = BS - \{B\}$.
  2. **Case 3.3.1**
  if B is a positive ordinary literal,
  $NewPS = PS'$
  $\cup\{\langle NewGS \cup GS', OD, IA, ANS\rangle\}$
  where
  $NewGS =$
  $\{fail((\{body(R)\} \cup BS')\theta) | \exists R \in \mathcal{P}$
  and $\exists$mgu θ s.t. $head(R)\theta = B\theta\}$
  Case 3.3.2
  if B is a ground negative ordinary literal or a ground askable literal or an abducible,
  $NewPS = PS'$
  $\cup\{\langle\{fail(BS')\} \cup GS', OD, IA, ANS\rangle\}$
  $\cup\{\langle\{\overline{B}\} \cup GS', OD, IA, ANS\rangle\}$

Case 3.4 If L is a ground askable literal, $Q@S$, then do the following:
1. if $L \notin AAQ$ and $\overline{L} \notin AAQ$, then send the question Q to the slave agent S and $NewAAQ = AAQ \cup \{L\}$.
2. if $\overline{L} \in OD$ then $NewPS = PS'$
else $NewPS = PS'$
$\cup\{\langle GS', OD \cup \{L\}, IA, ANS\rangle\}$.

□

Note that we can select any goal of any active process for a goal reduction and we guarantee the correctness of this procedure by the theorem below. This means that we can put additional strategies for an efficient goal reduction.

3.3 Fact Arrival Phase

Suppose an answer Q is returned from a slave agent S. Let $L = Q@S$. After one step of process reduction is finished, do the following:

- If $\overline{L} \in CBS$, then
$NewCBS = CBS - \{\overline{L}\} \cup \{L\}$
- else if $L \notin CBS$, then
$NewCBS = CBS \cup \{L\}$

Note that there might be some askable literals which are not included in the initial belief set. In this case, processes using such askable literals and ones using their complements are both suspended until the answers for the askable literals are returned and therefore no speculative computation is performed. This mechanism is used when we do not know the default value of the askable literal, or we need the definite answer to proceed the execution.

3.4 Correctness of the Proof Procedure

For the correctness of the proof procedure, we define the following. We use generalized stable model semantics (Kakas90) for the semantics of the above program.

Definition 5 *Let T be a set of rules and integrity constraints. We denote the set of ground rules obtained by replacing all the variables in every rule or every integrity constraint of T by every ground term in the language as Π_T.* □

Definition 6 *Let T be a set of rules and integrity constraints. Let M be a set of ground atoms and Π_T^M be the following program.*

$\Pi_T^M = \{H \leftarrow B_1, ..., B_l |$
$"H \leftarrow B_1, ..., B_l, \sim A_1, ..., \sim A_h." \in \Pi_P$
and $A_i \notin M$ for each $i = 1, ..., h.$}

Let $min(\Pi_T^M)$ be the least model of Π_T^M. A stable model *for a logic program T is M iff $M = min(\Pi_T^M)$ and $\bot \notin M$* □

Definition 7 *(Kakas90) Let T be a set of rules and integrity constraints, and Θ be a set of ground abducibles.*

A generalized stable model $M(\Theta)$ *is a stable model of $T \cup \mathcal{F}(\Theta)$ where*
$\mathcal{F}(\Theta) =$
$\{H \leftarrow | H$ *is a positive literal and* $H \in \Theta\}$.
□

The following theorem shows correctness of the above procedure. The theorem intuitively means that for any evaluation strategy of processes, when we receive an answer with a set of outside defaults and a set of inside assumptions from the proof procedure, the answer is correct w.r.t. a generalized stable model w.r.t. inside

assumptions and a program which is obtained from the original program and the current belief of the world.

Theorem 1 *Let $SF_{MS} = \langle \Sigma, \mathcal{E}, \Delta, \mathcal{A}, \mathcal{P}, \mathcal{I} \rangle$ be a speculative framework where $\mathcal{P}$ is a call-consistent logic program whose set of integrity constraints is satisfiable. Let GS be an initial goal set. Suppose that GS is reduced to $\emptyset$ when OD is outside defaults, IA is inside assumptions, ANS is set of instantiation of variables in GS and CBS is the current belief set. Let GS' be a goal obtained from GS by replacing all the variables in GS by ANS. Then, there is a generalized stable model $M(\Theta)$ for $\mathcal{P} \cup \mathcal{I} \cup \mathcal{F}(CBS)$, such that $M(\Theta) \models GS'$ and $OD \subseteq CBS$ and $IA \subseteq \Theta$.*

Sketch of Proof Since $\mathcal{P}$ is a call-consistent logic program, so is $\mathcal{P} \cup \mathcal{I} \cup \mathcal{F}(CBS)$. Then, we can construct an abductive derivation (Kakas90) from process reduction steps from $\langle GS, \emptyset, \emptyset, ANS' \rangle$ to $\langle \emptyset, OD, IA, ANS \rangle$ where ANS' is a set of variables in GS. This derivation is correct for generalized stable model semantics for a call-consistent logic program according to the result of (Kakas90). This means $M(\Theta) \models GS\theta$ and $OD \subseteq CBS$ and $IA \subseteq \Theta$. □

4 Example

We use the program in Example 1 and take the following strategy for process reduction.

1. When we reduce a positive literal, new processes are created along with the rule order in the program which are unifiable with the positive literal.
2. We always select a newly created or a newly resumed process and a left-most literal.

The following is an execution trace for $plan(1,4)$. We show PS, AAQ, CBS only when a change occurs. We abbreviate *plan* as *p*, *door* as *d*, *open* as *o*, *go_to_room* as $g^\dagger$, *lion_in* as *l*, *sensor* as *s* and *recognizer* as *r*. We underlined a selected literal in the selected active process.

1. $PS = \{\langle \{\underline{p(1,4)}\}, \emptyset, \emptyset \rangle\}$[7],
   $AAQ = \emptyset$,
   $CBS =$
   $\{o(1,2)@s, o(1,3)@s, o(2,4)@s, o(3,4)@s,$
   $\sim l(2)@r, l(3)@r, \sim l(4)@r\}$
2. By **Case 3.1**,
   $PS = \{$
   $\langle \{\underline{d(1,Y)}, o(1,Y)@s, g^\dagger(1,Y), p(Y,4)\}, \emptyset, \emptyset \rangle\}$
3. By **Case 3.1**,
   $PS = \{$
   $\langle \{o(1,2)@s, g^\dagger(1,2), p(2,4)\}, \emptyset, \emptyset \rangle,$
   $\langle \{o(1,3)@s, g^\dagger(1,3), p(3,4)\}, \emptyset, \emptyset \rangle\}$
4. By **Case 3.4**,
   $o(1,2)$ **is asked to the agent** s.
   $PS = \{$
   $\langle \{\underline{g^\dagger(1,2)}, p(2,4)\}, \{o(1,2)@s\}, \emptyset \rangle, P_1$[8] $\}$
   $AAQ = \{o(1,2)@s\}$
5. By **Case 3.2.3**,
   $PS = \{$
   $\langle \{\underline{fail(\{l(2)@r\})}, p(2,4)\}, \{o(1,2)@s\}, \{g^\dagger(1,2)\} \rangle,$
   $P_1\}$
6. By **Case 3.3.2**,
   $PS = \{$
   $\langle \{\underline{fail(\emptyset)}, p(2,4)\}, \{o(1,2)@s\}, \{g^\dagger(1,2)\} \rangle,$
   $\langle \{\sim l(2)@r, p(2,4)\}, \{o(1,2)@s\}, \{g^\dagger(1,2)\} \rangle,$
   $P_1\}$
7. By **Case 3.3**,
   $PS = \{$
   $\langle \{\underline{\sim l(2)@r}, p(2,4)\}, \{o(1,2)@s\}, \{g^\dagger(1,2)\} \rangle,$
   $P_1\}$
8. By **Case 3.4**,
   $\sim l(2)$ **is asked to the agent** r.
   $PS = \{$
   $\langle \{\underline{p(2,4)}\}, \{o(1,2)@s, \sim l(2)@r\}, \{g^\dagger(1,2)\} \rangle,$
   $P_1\}$
   $AAQ = \{o(1,2)@s, \sim l(2)@r\}$
9. $o(1,2)$ **is returned from** s**!!**
   Nothing changes.
10. By **Case 3.1**,
    $PS = \{$
    $\langle \{\underline{d(2,Y)}, o(2,Y)@s, g^\dagger(2,Y), p(Y,4)\}, \{o(1,2)@s, \sim l(2)@r\}, \{g^\dagger(1,2)\} \rangle,$
    $P_1\}$
11. By **Case 3.1**,
    $PS = \{$
    $\langle \{o(2,4)@s, g^\dagger(2,4), p(4,4)\}, \{o(1,2)@s, \sim l(2)@r\}, \{g^\dagger(1,2)\} \rangle$
    $P_1\}$
12. $l(2)$ **is returned from** r**!!**

[7] We omit ANS part in the process since there is no variable in the initial query.

[8] From now on P_1 is an abbreviation of $\langle \{o(1,3)@s, g^\dagger(1,3), p(3,4)\}, \emptyset, \emptyset \rangle$.

By **Fact Arrival Phase**,
$CBS =$
$\{o(1,2)@s, o(1,3)@s, o(2,4)@s, o(3,4)@s,$
$l(2)@r, l(3)@r, \sim l(4)@r\}$
$PS = \{$
$\langle\{\underline{o(1,3)@s}, g^{\dagger}(1,3), p(3,4)\}, \emptyset, \emptyset\rangle, P_2$[9] $\}$

13. By **Case 3.4**,
$o(1,3)$ **is asked to the agent** s.
$PS = \{$
$\langle\{\underline{g^{\dagger}(1,3)}, p(3,4)\}, \{o(1,3)@s\}, \emptyset\rangle, P_2\}$
$AAQ = \{o(1,2)@s, \sim l(2)@r, o(1,3)@s\}$

14. By **Case 3.2.3**,
$PS = \{$
$\langle\{\underline{fail(\{l(3)@r\})}, p(3,4)\},$
$\{o(1,3)@s\}, \{g^{\dagger}(1,3)\}\rangle,$
$P_2\}$

15. By **Case 3.3.2**,
$PS = \{$
$\{\underline{fail(\emptyset)}, \{o(1,3)@s\}, \{g^{\dagger}(1,3)\}\rangle,$
$\langle\{\sim l(3)@r, p(3,4)\},$
$\{o(1,3)@s\}, \{g^{\dagger}(1,3)\}\rangle,$
$P_2\}$
$AAQ =$
$\{o(1,2)@s, \sim l(2)@r, o(1,3)@s, \sim l(3)@r\}$

16. By **Case 3.3**,
$PS = \{$
$\langle\{\underline{\sim l(3)@r}, p(3,4)\},$
$\{o(1,3)@s\}, \{g^{\dagger}(1,3)\}\rangle,$
$P_2\}$
$AAQ =$
$\{o(1,2)@s, \sim l(2)@r, o(1,3)@s, \sim l(3)@r\}$

17. By **Case 3.4**,
$\sim l(3)$ **is asked to the agent** r.
$PS = \{$
$\langle\{p(3,4)\},$
$\{o(1,3)@s, \sim l(3)@r\}, \{g^{\dagger}(1,3)\}\rangle,$
$P_2\}$

18. $\sim l(2)$ **is returned from** r**!!**
By **Fact Arrival Phase**,
$CBS =$
$\{o(1,2)@s, o(1,3)@s, o(2,4)@s, o(3,4)@s,$
$\sim l(2)@r, l(3)@r, \sim l(4)@r\}$
$PS = \{$
$\langle\{\underline{o(2,4)@s}, g^{\dagger}(2,4), p(4,4)\},$
$\{o(1,2)@s, \sim l(2)@r\}, \{g^{\dagger}(1,2)\}\rangle$
P_3[10] $\}$

19. By **Case 3.4**,
$o(2,4)$ **is asked to the agent** s.
$PS = \{$
$\langle\{\underline{g^{\dagger}(2,4)}, p(4,4)\},$
$\{o(1,2)@s, \sim l(2)@r, o(2,4)@s\}, \{g^{\dagger}(1,2)\}\rangle$
$P_3\}$
$AAQ =$
$\{o(1,2)@s, \sim l(2)@r, o(1,3)@s,$
$\sim l(3)@r, o(2,4)@s\}$

20. By **Case 3.2.3**,
$PS = \{$
$\langle\{\underline{fail(\{l(4)@r\})}, p(4,4)\},$
$\{o(1,2)@s, \sim l(2)@r, o(2,4)@s\},$
$\{g^{\dagger}(1,2), g^{\dagger}(2,4)\}\rangle$
$P_3\}$

21. By **Case 3.3.2**,
$PS = \{$
$\langle\{\underline{fail(\emptyset)},$
$\{o(1,2)@s, \sim l(2)@r, o(2,4)@s\},$
$\{g^{\dagger}(1,2), g^{\dagger}(2,4)\}\rangle$
$\langle\{\sim l(4)@r, p(4,4)\},$
$\{o(1,2)@s, \sim l(2)@r, o(2,4)@s\},$
$\{g^{\dagger}(1,2), g^{\dagger}(2,4)\}\rangle$
$P_3\}$

22. By **Case 3.3**,
$PS = \{$
$\langle\{\underline{\sim l(4)@r}, p(4,4)\},$
$\{o(1,2)@s, \sim l(2)@r, o(2,4)@s\},$
$\{g^{\dagger}(1,2), g^{\dagger}(2,4)\}\rangle$
$P_3\}$

23. By **Case 3.4**,
$\sim l(4)$ **is asked to the agent** r.
$PS = \{$
$\langle\{\underline{p(4,4)}\},$
$\{o(1,2)@s, \sim l(2)@r, o(2,4)@s, \sim l(4)@r\},$
$\{g^{\dagger}(1,2), g^{\dagger}(2,4)\}\rangle$
$P_3\}$
$AAQ =$
$\{o(1,2)@s, \sim l(2)@r, o(1,3)@s,$
$\sim l(3)@r, o(2,4)@s, \sim l(4)@r\}$

24. By **Case 3.1**,
$PS = \{$
$\langle\emptyset,$
$\{o(1,2)@s, \sim l(2)@r, o(2,4)@s, \sim l(4)@r\},$
$\{g^{\dagger}(1,2), g^{\dagger}(2,4)\}\rangle$
$P_3\}$

25. $\{o(1,2)@s, \sim l(2)@r, o(2,4)@s, \sim l(4)@r\}$ and $\{g^{\dagger}(1,2), g^{\dagger}(2,4)\}$ are returned as outside defaults

[9] From now on P_2 is an abbreviation of
$\langle\{o(2,4)@s, g^{\dagger}(2,4), p(4,4)\},$
$\{o(1,2)@s, \sim l(2)@r\}, \{g^{\dagger}(1,2)\}\rangle$.

[10] From now on, P_3 is an abbreviation of
$\langle\{p(3,4)\}, \{o(1,3)@s, \sim l(3)@r\}, \{g^{\dagger}(1,3)\}\rangle$.

and inside assumptions.

At Step 4, a default $o(1,2)@s$ is assumed and computation of this process continues and at the same time, the real truth value of $o(1,2)$ is asked to the sensor s. If we had to wait for an answer from the agent s, we would have to suspend this process. This is an effect of speculative computation.

At Step 5, we perform an ordinary abduction. That is, we assume $g^{\dagger}(1,2)$ and check if any integrity constraint does not lead to contradiction. In this case, we check if $l(2)@r$ is not derived.

At Step 9, suppose that the answer is returned from s. Since this answer confirms the default, nothing changes and we can continue the reduction. This is the effect of speculative computation.

At Step 12, suppose the answer $l(2)$ is returned from r. Since this answer denies the default, we have to change the CBS and the current reduced process is suspended because the outside default is contradictory with the CBS.

At Step 17, the current reduced process is suspended because the askable literal is contradictory with the CBS.

At Step 18, suppose that the new answer $\sim l(2)$ is returned from r. Since this answer denies the default, we have to change the CBS and the process which becomes consistent with CBS is activated.

5 Related Research

In computer science, there are several research on speculative computation such as optimistic transaction in database, three-phase transition in fault-tolerant system, efficient execution mechanism for functional programming (Burton85) and parallel logic programming (Gregory93). This work is inspired by some of these works and is an abstract form of the above technique in multi-agent systems.

There are works on a formalization of an agent in terms of logic programming (Kowalski99; Hayashi99; Shanahan98). (Kowalski99) gives an agent cycle which reconciles reactive reasoning and proactive reasoning. However, they do not consider a dynamic belief revision about environment during proactive reasoning. (Shanahan98) proposes an important formalism of combining sensory data, planning and acting so that they can be interleaved. In his formalism, he considers not only a dynamic belief revision, but also a formalization of action which is not considered in our framework. The difference between his approach and our approach is its implementation. In his approach, every time new information leads to contradiction, the reasoning is restarted whereas in our framework, we return to the choice point where new information contradicts with the assumption. Moreover, he does not consider a speculative computation in his framework. The most closely related work with ours is (Hayashi99). He proposes a procedure of agent in planning where sensory data can be influenced even during reasoning process. In his procedure, the previous planning fragment can be reused. However, he does not consider speculative nature of the technique nor an abductive mechanism. Moreover, his procedure seems very complex because he considers deletion and addition of rules whereas our procedure is relatively simple since our procedure is a natural extension of the procedure of (Kakas90) with treatment of derivation using the current belief state.

After the original work of speculative computation (Satoh00), related research with speculative computation has been published (Sakama00; Inoue01). (Sakama00) gives a local semantics of an agent by translating a program into a new one attaching time stamp with each predicate. The revision of each agent is formalized as overriding the truth value of the predicate with the previous time stamp by the truth value of the predicate with the new time stamp. The idea of revision in the belief state is inspired by this revision mechanism and in fact the belief set in our procedure records the newest truth value of the predicate. (Inoue01) gives a bottom-up evaluation of speculative computation by calculating every answer w.r.t. possible replies. However, the work has a difficulty in controlling when queries are sent to other agents for the setting considered here.

This defect of controlling of queries also applies to answer set programming since answer set programming also computes a stable model in a bottom-up manner. Moreover, there is a problem of computation for abduction using answer set programming. Suppose that we have n abducibles. If we take a translation of abducibles using a technique of (Satoh91), then, at the worst case scenario, we have to check 2^n combinations of truth values of abducibles. Therefore, to apply answer set programming to abduction, more sophisticated control method of evaluation is necessary.

6 Conclusion

The following are the contributions of this paper.

- We combine speculative computation and abductive reasoning. We believe this combination gives

a foundation for more powerful reasoning systems.

- We propose a sound implementation of the combination for a master-slave multi-agent system in terms of generalized stable model semantics.

This is an initial work on dynamic belief revision and many issues remain for future research.

- Extensions to other kinds of multi-agent systems such as a system where every agent can communicate each other.
- Introducing other kinds of action than reasoning and sensing.
- Controlling effective manipulation of alternative processes. This is related with decision theoretic search since we need to estimate probabilities and utilities to assess exploration of alternative processes.
- Reducing redundancies of computation in order to reuse the same computation in alternative processes.
- Applying our framework to real applications such as robotics.

Acknowledgments

This research is partly supported by Grant-in-Aid for Scientific Research, "Construction of Multi-agent Systems with Speculative Computation", by The Ministry of Education, Culture, Sports, Science and Technology in Japan. I would be very grateful to Robert Kowalski and Murray Shanahan for instructive comments on this paper.

References

Burton, F. W., "Speculative Computation, Parallelism, and Functional Programming", *IEEE Transactions on Computers*, vol c-34 pp. 1190 – 1193 (1985).

Gregory, S., "Experiment with Speculative Parallelism in Parlog", Proceedings of ILPS'93, pp. 370 – 387 (1993).

Hayashi, H., "Replanning in Robotics by Dynamic SLDNF", *Proceedings of IJCAI-99 WS on Scheduling and Planning Meet – Real-Time Monitoring in a Dynamic and Uncertain World* (1999).

Inoue, K., Kawaguchi, S., and Haneda, H., "Controlling Speculative Computation in Multi-Agent Environments", *Proc. of ICLP2001 Workshop on Computational Logic in Multi-Agent Systems (CLIMA-01)*, pp. 9 – 18 (2001).

Kakas, A. C., Kowalski, R., Toni, F., "The Role of Abduction in Logic Programming", Handbook of Logic in Artificial Intelligence and Logic Programming 5, pp. 235 – 324 (1998).

Kakas, A. C., Mancarella, P., "On the Relation between Truth Maintenance and Abduction", Proceedings of PRICAI-90, pp. 438 – 433 (1990).

Kowalski, R. A., Sadri, F., "From Logic Programming to Multi-Agent Systems", To appear in Annals of Mathematics and Artificial Intelligence (1999).

Sakama, C., Inoue, K., Iwanuma, K., and Satoh, K., "A Defeasible Reasoning System in Multi-Agent Environments", *Proc. of CL2000 Workshop on Computational Logic in Multi-Agent Systems (CLIMA-00)*, pp.1 – 6 (2000).

Satoh, K., Inoue, K., Iwanuma, K., and Sakama, C., "Speculative Computation by Abduction under Incomplete Communication Environments", *Proc. of ICMAS2000*, pp. 263 – 270 (2000).

Satoh, K., Iwayama, N., "Computing Abduction by Using the TMS", *Proc. ICLP91*, pp. 505 – 518 (1991).

Satoh, K., Yamamoto, K., "Speculative Computation with Multi-Agent Belief Revision", *Proc. of AAMAS2002*, to appear (2000).

Shanahan, M. P., "Reinventing Shakey", *Working Notes of AAAI Fall Symposium on Cognitive Robotics* (1998), also in *Logic-Based Artificial Intelligence*, J. Minker ed., Kluwer Academic Press, pp. 233 – 253 (2000).

Pruning Operators for Answer Set Programming Systems

Francesco Calimeri
University of Calabria
87030 Rende (CS), Italy
calimeri@mat.unical.it

Wolfgang Faber
TU Wien
A-1040 Wien, Austria
faber@kr.tuwien.ac.at

Nicola Leone
University of Calabria
87030 Rende (CS), Italy
leone@unical.it

Gerald Pfeifer
TU Wien
A-1040 Wien, Austria
pfeifer@dbai.tuwien.ac.at

Abstract

Answer Set Programming (ASP) is a novel paradigm in Logic Programming, which allows for solving problems in a simple and highly declarative way. The language of ASP (function-free disjunctive logic programming) is very expressive and supports the representation of problems of high computational complexity (specifically, all problems in the complexity class $\Sigma_2^P = NP^{NP}$). Importantly, the ASP encoding of a large variety of problems is often very concise, simple, and elegant.

In this paper, we explain the computational process performed by ASP systems, with a focus on search space pruning, which is crucial for efficiency. We analyze the properties of two main pruning operators, namely (*Fitting's operator* and *Well-founded operator*), discuss their peculiarities and differences with respect to efficiency and effectiveness. We design an intelligent strategy for combining the two operators, which exploits the advantages of both. We implement our approach in the ASP system **DLV**, and perform some experiments. The experiments show interesting results, and evidence how the choice of the pruning operator affects the performance of ASP systems.

1 INTRODUCTION

The high expressiveness of answer set programming (which allows for representing all problems in the complexity class $\Sigma_2^P = NP^{NP}$ [Eiter et al., 1997]) comes at the price of a high computational cost in the worst case, which makes the implementation of efficient ASP systems a difficult task. Still some efforts have been made in this direction. After some pioneering work on stable model computation [Bell et al., 1994, Subrahmanian et al., 1995], a number of modern ASP systems are now available. The two most widespread ASP systems are **DLV** [Faber et al., 1999a, Faber et al., 2001] and Smodels [Niemelä, 1999, Simons, 2000]; but also other systems support ASP to some extent, including NoMoRe [Anger et al., 2001], QUIP [Egly et al., 2000], CCALC [McCain and Turner, 1998], XSB [Rao et al., 1997], and DCS [East and Truszczyński, 2000]. Nevertheless, much work has to be done to make ASP systems fully satisfactory for modern knowledge-based applications. The design of new optimization techniques and smart algorithms for the computation of ASP programs is of utmost importance. The present paper goes in this direction, focusing on search space pruning, an extremely critical problem for the efficiency of ASP systems. The main contributions of the paper are as follows:

- We describe the main steps of the computational process performed by ASP systems with a focus on search space pruning, which is crucial for efficiency. We analyze the properties of the disjunctive extensions of two well-known pruning operators for ASP, *Fitting*'s operator and the *Well-founded* operator. We discuss their strengths and weaknesses w.r.t. efficiency and effectiveness.

- We design an intelligent strategy for combining these two pruning operators, which exploits the advantages of both, starting from several known results established in different works on ASP and focusing on: modularity properties [Lifschitz and Turner, 1994, Eiter et al., 1997, Leone et al., 1997], head-cycle free programs [Ben-Eliyahu and Dechter, 1994], acyclic programs [Fages, 1994], disjunctive unfounded sets and complexity [Leone et al., 1997],

combining them in a smart way.

- We implement our approach in the ASP system **DLV**, taking care of efficiency issues and respecting the known complexity bounds. Indeed, the fixpoints of Fitting's operator are computed in linear time, as are the Greatest Unfounded Sets (which contribute the negative inferences in the Well-founded operator).
- We report experimental results on a number of benchmark problems to assess the validity of our approach. The experiments show that the choice of the pruning operator has a strong impact on the performance of ASP systems, and specifically that our techniques considerably improve the efficiency of the ASP system **DLV**.

To our knowledge, this is the first paper that focuses on pruning the search space for *disjunctive* ASP programs. A number of related works have studied pruning operators in the domain of non-disjunctive ASP programs. The use of the well-founded operator in the computation of non-disjunctive programs has been first proposed in [Leone et al., 1993] while its concrete implementation in a system was first done in [Subrahmanian et al., 1995]. The well-founded operator corresponds to the *upper closure* operator of the Smodels system [Niemelä, 1999], which has been implemented very efficiently in Smodels, employing a novel optimization technique to localize the computation [Simons, 2000].

2 ASP PROGRAMS AND NOTATION

A *(disjunctive) rule* r is a formula

$$a_1 \vee \cdots \vee a_n \text{ :- } b_1, \cdots, b_k, \texttt{ not } b_{k+1}, \cdots, \texttt{ not } b_m.$$

where $a_1, \cdots, a_n, b_1, \cdots, b_m$ are classical literals (atoms possibly preceded by the classical negation symbol $\neg$), and $n \geq 0$, $m \geq k \geq 0$. Given a rule r, let $H(r) = \{a_1, ..., a_n\}$ denote the set of head literals, $B^+(r) = \{b_1, ..., b_k\}$ and $B^-(r) = \{\texttt{not } b_{k+1}, ..., \texttt{not } b_m\}$ the set of positive and negative body literals, resp., and $B(r) = B^+(r) \cup B^-(r)$.

A rule r with $B^-(r) = \emptyset$ is called *positive*; a rule with $H(r) = \emptyset$ is referred to as *integrity constraint*. If the body is empty we usually omit the :- sign. Comparison operators (like $=, <, >, <>$) are built-in predicates and may appear in rule bodies.

An *(ASP) program* $\mathcal{P}$ is a finite set of rules; $\mathcal{P}$ is a *positive* program if all rules in $\mathcal{P}$ are positive (i.e., not-free). An object (atom, rule, etc.) containing no variables is called *ground* or *propositional*.

Given a literal l, let $\texttt{not}.l = a$ if $l = \texttt{not } a$, otherwise $\texttt{not}.l = \texttt{not } l$, and given a set L of literals, $\texttt{not}.L = \{\texttt{not}.l \mid l \in L\}$.

Herbrand Universe, *Herbrand Base*, and the *ground instantiation Ground($\mathcal{P}$)* of $\mathcal{P}$ are defined as usual. The semantics of an ASP program is given by its (consistent) answer sets - an extension of the concept of stable models to the realm of ASP programs proposed by Gelfond and Lifschitz in [Gelfond and Lifschitz, 1991], for a nice introduction we refer to [Lifschitz, 1996].

Example 2.1 *For the program* $\mathcal{P} = \{a \vee \neg\ b \vee c.,\ \text{:- } a.,\ \neg\ b \text{ :- } c.,\ c \text{ :- } \neg\ b.\}$, $I_1 = \{a, \neg\ b\}$ *and* $I_2 = \{c\}$ *are interpretations, but* $M = \{\neg\ b, c\}$ *is the only answer set.*

3 ANSWER SETS COMPUTATION

In this section, we describe the main steps of the computational process performed by ASP systems. We will describe the computational engine of the **DLV** system [Faber et al., 1999a, Faber et al., 2001] which will be used for the experiments, but also other ASP systems like Smodels [Niemelä and Simons, 1996, Simons, 2000] employ a very similar procedure.

In general, an answer set program $\mathcal{P}$ contains variables. The computational step of an ASP system eliminates these variables, generating a ground instantiation *ground($\mathcal{P}$)* of $\mathcal{P}$ which is a (usually much smaller) subset of all syntactically constructible instances of the rules of $\mathcal{P}$ having precisely the same answer sets as $\mathcal{P}$ [Faber et al., 1999a]. In this phase, also true negation is eliminated by a straightforward rewriting pass (so in the following "atoms" might be rewritten classical literals). The hard part of the computation is then performed on this simplified ground program by the Model Generator, which is sketched in Figure 1. For brevity, $\mathcal{P}$ refers to the simplified ground program in the sequel.

Roughly, the Model Generator produces some "candidate" answer sets. Each candidate I is then verified by the function *IsAnswerSet(I)*, which checks whether I is a minimal model of the program $\mathcal{P}^I$ obtained by applying the GL-transformation w.r.t. I.

The interpretations handled by the Model Generator are partial interpretations. A *partial* interpretation I is a consistent set of atoms, possibly preceded by the default negation symbol (not). A ground literal L is true w.r.t. I if $L \in I$; it is false w.r.t. I if $\texttt{not}.L \in I$; L

```
Function ModelGenerator(var I: Interpretation): bool;
var  inconsistency: bool;
begin
     DetCons(I,inconsistency);
     if inconsistency then return false;
     if"no atom is undefined in I" then
       return IsAnswerSet(I);
     Select an undefined atom A using a heuristic;
     if ModelGenerator(I ∪ {A}) then return true;
     else return ModelGenerator(I ∪ {not A});
end;
```

Figure 1: Computation of Answer Sets

is undefined otherwise. Initially, the ModelGenerator function is invoked with I set to the empty interpretation (all atoms are undefined at this stage). If the program $\mathcal{P}$ has an answer set, then the function returns true and sets I to the computed answer set; otherwise it returns false. The Model Generator is similar to the Davis-Putnam procedure in SAT solvers. It first calls a function DetCons, which extends I with those literals that can be deterministically inferred. This is similar to unit propagation as employed by SAT solvers, but exploits the peculiarities of ASP for making further inferences (e.g., it uses the knowledge that every answer set is a minimal model).

If DetCons does not detect any inconsistency, an atom A is selected according to a heuristic criterion and ModelGenerator is recursively called on both $I \cup \{A\}$ and $I \cup \{\texttt{not } A\}$. The atom A corresponds to a branching variable of a SAT solver.

For the performance of an ASP system the implementation of DetCons is crucial in two ways: It has to perform its task as quickly as possible, while pruning the search space as much as possible.

4 PRUNING OPERATORS

In this section we review two operators that are useful to implement DetCons. As already mentioned, DetCons has to expand a given interpretation as much as possible to reduce the search space, while ensuring that such an expansion never causes any answer set to be missed. In other words, if an interpretation I is contained in an answer set M, that answer set will also contain the expansion of I computed by DetCons. We can state this "safety" property formally for a generic operator $\Gamma_{\mathcal{P}}$: for each interpretation I and each answer set M of a given program $\mathcal{P}$, $I \subseteq M$ iff $\Gamma_{\mathcal{P}}(I) \subseteq M$.

The two operators in question are the *Fitting* ($\Phi_{\mathcal{P}}$) operator and the *Well-founded* ($\mathcal{W}_{\mathcal{P}}$) operator. Both have the property described above, and extend the two corresponding operators defined for disjunction-free programs [Fitting, 1985, van Gelder et al., 1991] to the class of disjunctive logic programs. Both $\Phi_{\mathcal{P}}$ and $\mathcal{W}_{\mathcal{P}}$ consist of two parts: The part drawing positive inferences (which is an extension of the immediate consequence operator $T_{\mathcal{P}}$, defined for three-valued interpretations of normal logic programs [van Gelder et al., 1991], to disjunctive programs) is the same for both operators; they only differ in the way they perform negative inferences.

Definition 4.1 Let $\mathcal{P}$ be a program, and I a (partial) interpretation.

$$T_{\mathcal{P}}(I) = \{a \mid \exists r \in \mathcal{P} \text{ s.t. } a \in H(r) : H(r) - \{a\} \subseteq \texttt{not}.I \wedge B(r) \subseteq I\}$$

Example 4.1 *Consider the program* $\mathcal{P}_1 = \{a \vee b.,\ c \text{ :- not } a.,\ d \text{ :- } e.,\ e \text{ :- } d.,\ k \text{ :- not } e.\}$. *Suppose* $I = \{\texttt{not } a\}$, *then* $T_{\mathcal{P}}(I) = \{b, c\}$.

4.1 FITTING ($\Phi_{\mathcal{P}}$) OPERATOR

$\Phi_{\mathcal{P}}$ extends the *Fitting* operator [Fitting, 1985] to the disjunctive case. Intuitively, $\Phi_{\mathcal{P}}$ derives falsity of those atoms for which there is no rule left that could be used to derive them as true.

Definition 4.2 Let $\mathcal{P}$ be a program, and I a (partial) interpretation. Define $\gamma_{\mathcal{P}}$, $\Phi_{\mathcal{P}}$ and $\{F_n\}$ as follows.

$$\gamma_{\mathcal{P}}(I) = \{a \mid \forall r \in \mathcal{P} \text{ s.t. } a \in H(r) : (H(r) - \{a\}) \subseteq I \vee B(r) \cap \texttt{not}.I \neq \emptyset\}$$
$$\Phi_{\mathcal{P}}(I) = T_{\mathcal{P}}(I) \cup \texttt{not}.\gamma_{\mathcal{P}}(I)$$
$$F_0 = I,\quad F_k = F_{k-1} \cup \Phi_{\mathcal{P}}(F_{k-1}) \quad \textit{for } k > 0$$

$F_0, \ldots, F_n$ is a growing sequence whose *n-th* term is the *n-fold* application of $\Phi_{\mathcal{P}}$ to I. The least fixpoint $\Phi^{\infty}_{\mathcal{P}}(I)$ of $\Phi_{\mathcal{P}}$ containing I, is defined as the limit to which $\{F_n\}_{n \in \mathcal{N}}$ converges.

Example 4.2 *Consider the program* $\mathcal{P}_1$ *of Example 4.1, and the interpretation* $I = \{a\}$. *Here* $\gamma_{\mathcal{P}}(I) = \{b, c\}$. *It is easy to see that* $\Phi_{\mathcal{P}}(I) = \emptyset \cup \texttt{not}.\{b, c\} = \{\texttt{not } b, \texttt{not } c\}$. *We thus obtain* $F_0 = \{a\}, F_1 = \{a, \texttt{not } b, \texttt{not } c\}, F_2 = F_1 = \Phi^{\infty}_{\mathcal{P}}(I)$.

Proposition 4.1 *For every answer set* M *of a given program* $\mathcal{P}$, *an interpretation* $I \subseteq M$ *iff* $\Phi^{\infty}_{\mathcal{P}}(I) \subseteq M$.

Importantly, for any interpretation I, $\Phi^{\infty}_{\mathcal{P}}(I)$ is *linear-time computable*. The $\Phi_{\mathcal{P}}$ operator seems to be a good choice as a pruning operator, as it is "*safe*", has the capability to perform negative inferences, and its fixpoint $\Phi^{\infty}_{\mathcal{P}}(I)$ is efficiently computable.

Unfortunately, $\Phi_{\mathcal{P}}$ fails to derive all possible negative consequences. For instance, in Example 4.2 it fails to derive d and e as false w.r.t. I while the only rules having these atoms in the head will never have a true body. The *Well-founded* operator presented in the following section is "stronger" in this respect.

4.2 WELL-FOUNDED ($\mathcal{W}_{\mathcal{P}}$) OPERATOR

The $\mathcal{W}_{\mathcal{P}}$ operator defined in [Leone et al., 1997] extends the operator defined in [van Gelder et al., 1991] (whose least fixpoint is the well-founded model) to the disjunctive case. It is defined by an extension of the notion of *unfounded sets* to disjunctive logic programs.

Definition 4.3 Let I be a (partial) interpretation for a program $\mathcal{P}$. A set $X \subseteq B_{\mathcal{P}}$ of ground atoms is an *unfounded set* for $\mathcal{P}$ w.r.t. I if, for each $a \in X$ and for each rule $r \in \mathcal{P}$ such that $a \in H(r)$, at least one of the following conditions holds: (i) $B(r) \cap \mathtt{not}.I \neq \emptyset$, (ii) $B^+(r) \cap X \neq \emptyset$, (iii) $(H(r) - X) \cap I \neq \emptyset$.

While for non-disjunctive programs the union of unfounded sets is again an unfounded set for all interpretations, this does not hold in general for disjunctive programs (see [Leone et al., 1997]). For instance, for $\mathcal{P} = \{a \vee b\}$ and $I = \{a, b\}$, both $\{a\}$ and $\{b\}$ are unfounded sets w.r.t. I; but their union $\{a, b\}$ is not. Let $\mathbf{I}_{\mathcal{P}}$ denote the set of all interpretations of $\mathcal{P}$ for which the union of all unfounded sets for $\mathcal{P}$ w.r.t. I is an unfounded set for $\mathcal{P}$ w.r.t. I as well. Given $I \in \mathbf{I}_{\mathcal{P}}$, let $GUS_{\mathcal{P}}(I)$ (the *greatest unfounded set* of $\mathcal{P}$ w.r.t. I) denote the union of all unfounded sets for $\mathcal{P}$ w.r.t. I. We can now introduce the *Well-founded* operator.

Definition 4.4 Let $\mathcal{P}$ be a program, and $I \in \mathbf{I}_{\mathcal{P}}$ a (partial) interpretation. Define $\mathcal{W}_{\mathcal{P}}$ and $\{W_n\}$ as follows:

$$\mathcal{W}_{\mathcal{P}}(I) = \mathcal{T}_{\mathcal{P}}(I) \cup \mathtt{not}.GUS_{\mathcal{P}}(I).$$
$$W_0 = I, \quad W_k = W_{k-1} \cup \mathcal{W}_{\mathcal{P}}(W_{k-1}) \quad \textit{for } k > 0$$

$W_0, \ldots, W_n$ is a growing sequence whose *n-th* term is the *n-fold* application of $\mathcal{W}_{\mathcal{P}}$ to I. The least fixpoint $\mathcal{W}_{\mathcal{P}}^{\infty}(I)$ of $\mathcal{W}_{\mathcal{P}}$ containing I is the limit to which $\{W_n\}_{n \in \mathcal{N}}$ converges. Note that $\mathcal{W}_{\mathcal{P}}^{\infty}(I) \in \mathbf{I}_{\mathcal{P}}$ if $I \in \mathbf{I}_{\mathcal{P}}$.

Example 4.3 *Considering the program $\mathcal{P}_1$ of Example 4.1 and the interpretation $I = \{a\}$, we get $GUS_{\mathcal{P}}(I) = \{b, c, d, e\}$. b is added because of (iii), and c because of (i) in Definition 4.3. Then d and e appear in the head of only a single rule each and for both (ii) of Definition 4.3 holds. We obtain $\mathcal{W}_{\mathcal{P}}(I) = \{\mathtt{not}\ b, \mathtt{not}\ c, \mathtt{not}\ d, \mathtt{not}\ e\}$ and $W_0 = \{a\}, W_1 = \{a, \mathtt{not}\ b, \mathtt{not}\ c, \mathtt{not}\ d, \mathtt{not}\ e\}, W_2 = \{a, \mathtt{not}\ b, \mathtt{not}\ c, \mathtt{not}\ d, \mathtt{not}\ e, k\}, W_3 = W_2 = \mathcal{W}_{\mathcal{P}}^{\infty}(I)$.*

Proposition 4.2 [Leone et al., 1997] Let I be an interpretation for a program $\mathcal{P}$, and let M be an answer set for $\mathcal{P}$. If $I \subseteq M$, then (a) I belongs to the domain $\mathbf{I}_{\mathcal{P}}$ of $\mathcal{W}_{\mathcal{P}}$, and (b) $\mathcal{W}_{\mathcal{P}}(I) \subseteq M$.

The drawback of $\mathcal{W}_{\mathcal{P}}$ in the disjunctive case is that it is applicable only for interpretations in the class $\mathbf{I}_{\mathcal{P}}$. Deciding whether $I \in \mathbf{I}_{\mathcal{P}}$ is co-NP-hard [Scarcello, 1997, Leone et al., 1997], and cannot be efficiently tested unless P = NP.

5 PRUNING OPERATORS ON SYNTACTICALLY RESTRICTED CLASSES OF PROGRAMS

In this section, we explore several interesting properties of the operators on some restricted (syntactic) classes of programs. To this end, we introduce *dependency graphs* which are based on the dependencies of head predicates on the positive body predicates of a rule.

With every program $\mathcal{P}$, we associate a directed graph $DG_{\mathcal{P}} = (\mathcal{N}, E)$, called the *dependency graph* of $\mathcal{P}$, where (i) each predicate of $\mathcal{P}$ is a node in $\mathcal{N}$, and (ii) there is an arc in E directed from node a to node b if there is a rule r in $\mathcal{P}$ such that two predicates a and b of literals appear in $B^+(r)$ and $H(r)$, respectively.

A *component* is a maximal strongly connected subset of nodes. Since there is a one-to-one correspondence between nodes in $DG_{\mathcal{P}}$ and predicates in $\mathcal{P}$, each component of $DG_{\mathcal{P}}$ is related to a subset of predicates (which we call a "component" of $\mathcal{P}$ as well). Those rules in $\mathcal{P}$ which have a head predicate in C, form a *subprogram* of $\mathcal{P}$ restricted to C[1]. We denote this subprogram by $\mathcal{P}_C$. By $Comp(\mathcal{P})$ we denote the set of all "components", again referring to both predicates and related subprograms.

Given a program $\mathcal{P}$ and its dependency graph $DG_{\mathcal{P}}$, we say that:

- a component C is *cyclic* if the related subprogram $\mathcal{P}_C$ of $\mathcal{P}$ contains at least one recursive rule (i.e., a rule r such that a head predicate and a positive body predicate of r are in C); C is *acyclic* if it is not cyclic.
- a component C is *head-cycle-free* (HCF) iff the related subprogram $\mathcal{P}_C$ of $\mathcal{P}$ contains no rule r such that two predicates occurring in the head of r belong to C.

[1]Note that a disjunctive rule may occur in the subprograms of two different components.

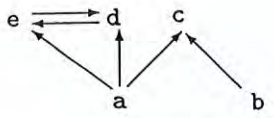


Figure 2: $DG_{\mathcal{P}_1}$ *graph*

$DG_{\mathcal{P}}$ and $\mathcal{P}$ are *cyclic* if there is *at least one cyclic* component, otherwise they are acyclic. They are HCF if *all* components are HCF.

Example 5.1 *Consider the following program* $\mathcal{P}_1$*:*

$$\{\ a \vee b.,\ \ c \text{ :- } a.\ ,\ \ c \text{ :- } b.\ ,\ \ d \vee e \text{ :- } a.\ ,\ \ d \text{ :- } e.,\ \ e \text{ :- } d, \texttt{not } b.\ \}$$

The dependency graph $DG_{\mathcal{P}_1}$ *of* $\mathcal{P}_1$ *is depicted in Figure 2. There are four components:* $\{a\}$, $\{b\}$, $\{c\}$, $\{d,e\}$. *All of them are acyclic except for the last which is also the only non-*HCF *component, as the head of* $d \vee e$:- a. *contains two predicates belonging to the same cycle. The whole graph, and thus the program, is cyclic but not* HCF.

We next present two well-known theorems about the operators $\Phi^{\infty}_{\mathcal{P}}(I)$ and $\mathcal{W}^{\infty}_{\mathcal{P}}(I)$, the importance of which will become clear in the next section.

Theorem 5.1 Given a program $\mathcal{P}$ and an interpretation I, if $\mathcal{P}$ is *acyclic* then $\mathcal{W}_{\mathcal{P}}(I) = \Phi_{\mathcal{P}}(I)$.

Intuitively, since $\mathcal{P}$ is acyclic, Condition (ii) of Definition 4.3 is irrelevant, and the unfoundedness conditions coincide with the conditions of applicability of the $\gamma_{\mathcal{P}}$ operator of Definition 4.2.

The next theorem follows from the results in [Leone et al., 1997].

Theorem 5.2 Let $\mathcal{P}$ be a HCF ground program and I be an interpretation, then

- I belongs to $\mathbf{I}_{\mathcal{P}}$ (i.e., I is in the domain of $\mathcal{W}_{\mathcal{P}}$),
- $\mathcal{W}^{\infty}_{\mathcal{P}}(I)$ is computable in *quadratic time* in the size of $\mathcal{P}$.

6 EFFICIENT COMBINATION OF PRUNING OPERATORS

We now show how to combine the $\Phi_{\mathcal{P}}$ and $\mathcal{W}_{\mathcal{P}}$ operators, resulting in an efficient implementation of DetCons.

6.1 A PONDERED CHOICE

From the previous sections, given a program $\mathcal{P}$ and an interpretation I, we know that:

- the computation of $\Phi^{\infty}_{\mathcal{P}}(I)$ is always very efficient (*linear time computable*);
- $\mathcal{W}_{\mathcal{P}}$ is "stronger" than $\Phi_{\mathcal{P}}$ (i.e., $\Phi_{\mathcal{P}}(I) \subseteq \mathcal{W}_{\mathcal{P}}(I)$ for any interpretation I);
- the computation of $\mathcal{W}_{\mathcal{P}}$ is intractable in the general case (since deciding whether an interpretation belongs to its domain is co-NP-hard);
- the computation of $\mathcal{W}^{\infty}_{\mathcal{P}}(I)$ is tractable (*quadratic*) when $\mathcal{P}$ belongs to the restricted class of *head-cycle-free* programs;
- $\Phi_{\mathcal{P}}$ is equivalent to $\mathcal{W}_{\mathcal{P}}$ when $\mathcal{P}$ belongs to the restricted class of *acyclic* programs.

Based on these observations, we have designed an approach which exploits the positive aspects of both operators, including the efficiency of *Fitting*'s operator wherever we are sure that it is equivalent to the *Well-founded* operator or that the computation of the latter it intractable. On the other hand, our approach takes advantage of the (potentially) stronger pruning of the *Well-founded* operator where feasible. To that end we employ some modularity properties of Disjunctive Logic Programs.

Proposition 6.1 [Leone et al., 1997] Let $\Gamma_{\mathcal{P}} \in \{\Phi_{\mathcal{P}}, \mathcal{W}_{\mathcal{P}}\}$, and I be an interpretation for a program $\mathcal{P}$. Then, $\Gamma_{\mathcal{P}}(I) = \bigcup_{C \in Comp(\mathcal{P})} \Gamma_{\mathcal{P}_C}(I)$, where $Comp(\mathcal{P})$ is the set of all components of $\mathcal{P}$, and $\mathcal{P}_C$ denotes the subprogram of component C.

Our implementation of DetCons is sketched in Figure 3. Proposition 6.1 allows us to realize a combined use of the two operators (Fitting and Well-founded) by choosing the most suitable, depending on the syntactical structure of each component. In particular, we apply $\mathcal{W}_{\mathcal{P}}$ on *cyclic and HCF* components, where it is stronger than $\Phi_{\mathcal{P}}$ but efficiently computable. On the other hand, we apply $\Phi_{\mathcal{P}}$ on *acyclic* components, where it is equivalent to $\mathcal{W}_{\mathcal{P}}$ and more efficiently computable, and on *cyclic and not-HCF* components, where $\mathcal{W}_{\mathcal{P}}$ is known to be intractable.

Thus, if the input program is acyclic, we always apply the linear operator $\Phi_{\mathcal{P}}$ without any loss in pruning strength; but if the program is cyclic, we limit the application of $\mathcal{W}_{\mathcal{P}}$ to those components (*cyclic and HCF*) where it can result in a strong pruning of the search space.

```
Procedure DetCons(var I: Interpretation,
                  var inconsistency: bool);
begin
  inconsistency := false;
  for each component C ∈ Comp(P) do
  begin
    switch classOf(C)
      case acyclic:
        ComputeFittingFixpoint(I, inconsistency);
      case cyclic-notHCF:
        ComputeFittingFixpoint(I, inconsistency);
      case cyclic-HCF:
        ComputeWellFoundedFixpoint(I, inconsistency);
    end;
    if inconsistency then break;
  end;
end;
```

Figure 3: Procedure DetCons

6.2 IMPLEMENTATION ISSUES

We conclude this section with some implementation remarks. Due to space constraints, DetCons has been presented in a simplified manner and details concerning the standard propagation routines and further propagation rules have been omitted (see [Faber et al., 1999b, Faber et al., 2001] for details).

At the beginning of the ASP computation, we classify the components of the program w.r.t. acyclicity and head-cycle freeness, since the *DetCons* procedure needs this information. Such a classification is done very efficiently in linear time. We first build the dependency graph $DG_{\mathcal{P}}$ of $\mathcal{P}$ (in linear time); then, we compute the strongly connected components of $DG_{\mathcal{P}}$ applying the linear-time Tarjan algorithm [Tarjan, 1972], and we finally scan the components checking whether they are acyclic or HCF.

To implement the *Well-founded* operator, we have designed an algorithm computing $GUS_{\mathcal{P},C}(I)$ for for an interpretation I and a cyclic HCF component C of a program $\mathcal{P}$.

Recall Definition 4.3 where three conditions account for cases in which a set of atoms cannot be derived. Conditions (i) and (iii) basically correspond to rule satisfaction (w.r.t. I and $I - X$, respectively), while condition (ii) is used to detect positive cycles without foundation. The basic idea is to compute $C - GUS_{\mathcal{P},C}(I)$ by incrementally deriving atoms in C which are "founded", i.e., which do not belong to $GUS_{\mathcal{P},C}(I)$. That is, we build a finite sequence $Y_0, \ldots, Y_n$, where $Y_0 = \emptyset$ and $Y_n = C - GUS_{\mathcal{P},C}(I)$. To this end, we look for rules which do not satisfy any of the three conditions of Definition 4.3 (the conditions are checked w.r.t. X set to Y_i and interpretation I). Once one such a rule r is found, we derive that $H(r) \cap C$ (which is a single atom, since the C is HCF) is "founded", that is, it does not belong to $GUS_{\mathcal{P},C}(I)$, and can be added to Y_{i+1}. The foundedness of an atom may imply the foundedness of further atoms; we proceed until a fixpoint is reached.

At the end of the computation, all atoms in $C - Y_n$ are known to be unfounded, and we set them to false in I. This can result in inconsistency if $I \cap C - Y_n \neq \emptyset$, i.e., if an unfounded atom was set to true in I.

Frequently, all atoms in $GUS_{\mathcal{P},C}(I)$ happen to be already false w.r.t. I, and its computation is completely useless. We would like to identify cases where this condition is recognized without actually computing $GUS_{\mathcal{P},C}(I)$. To this end, at each step of DetCons (Figure 3), we propagate the deterministic consequences over all components by means of the *Fitting* operator [2] and then execute the GUS-computation only on some "selected" components instead of "all" HCF and cyclic components.

In particular, we only call the GUS-computation for components where some atom may have become unfounded by the most recent propagation step. In order to do that, we store some further information during the *Fitting* propagation.

Basically, an atom A can become unfounded if it has lost a "potential support", as some rule r containing A in the head has become satisfied during the last propagation (either the body of r has become false or a head atom of r, different from A, has become true).[3] If no atom of a component C has lost any "potential support", then $GUS_{\mathcal{P},C}(I)$ is unaltered, and its computation is superfluous. To automatically recognize such superfluous computations, when a rule becomes satisfied, we push the component of each each atom that loses its support in a queue. And we eventually launch the GUS-computation only for the components stored in this queue, i.e. only for those cyclic and HCF components in which at least one head atom lost a "potentially supporting rule". This way, we avoid a lot of useless GUS computations.

It is worthwhile noting that we have designed and implemented *linear-time* algorithms for computing both the *Greatest Unfounded Set* (as described above) and the *Fitting* operator. These algorithms use propagation queues and suitable counters like the *Dowling and Gallier* [Dowling and Gallier, 1984, Minoux, 1988] algorithm; we omit the description for space limitations,

[2] In a "localized" way, using a technique à la *Dowling and Gallier*.

[3] A (disjunctive) rule can support only one atom in its head.

full details can be found in [Calimeri, 2001].

7 COMPARISONS AND BENCHMARKS

In order to evaluate our intuitions, we have implemented two new pruning operators, based on the conclusions drawn in the previous section in the ASP system **DLV** [Faber et al., 1999b, Faber et al., 2001]. and experimentally compared the new pruning operators against the original pruning operator employed by **DLV**. Next, we describe the compared methods, the benchmark problems and data, and we finally discuss the results of the experiments.

7.1 OVERVIEW OF THE COMPARED METHODS

We have compared the following three methods in the ASP system **DLV**.

Old. The method originally employed by **DLV**. It always uses the generalized Fitting operator $\Phi_{\mathcal{P}}$ introduced in Section 4.1, which is efficiently computable (a fixpoint is reached in linear time), but does not prune the search space as much as $\mathcal{W}_{\mathcal{P}}$.

ifPoss. Based on the generalized Well-founded operator $\mathcal{W}_{\mathcal{P}}$ introduced in Section 4.2, and exploiting observations from Section 6, this method avoids the use of $\mathcal{W}_{\mathcal{P}}$ on those components where its computation is very expensive (i.e., deciding its applicability is intractable). It employs $\mathcal{W}_{\mathcal{P}}$ on all head-cycle free components, while it resorts to the generalized Fitting operator $\Phi_{\mathcal{P}}$ on the remaining (i.e., non-HCF) components.

ifNeed. This is the method described in Figure 3. It fully implements the theoretical results from Section 6, using both $\Phi_{\mathcal{P}}$ and $\mathcal{W}_{\mathcal{P}}$ where appropriate, and is a refinement of method **ifPoss**. $\mathcal{W}_{\mathcal{P}}$ is only used on cyclic head-cycle free components, whereas $\Phi_{\mathcal{P}}$ is applied for all acyclic components.

7.2 BENCHMARK PROBLEMS

To properly evaluate the pruning techniques described in the previous sections, we chose a couple of benchmark problems: Hamiltonian path, Blocksworld Planning, and Sokoban.

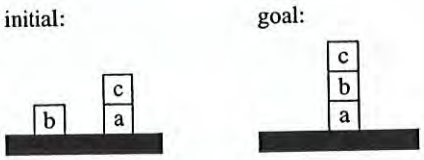
Hamiltonian Path (HAMPATH) is a classical NP-complete problem from graph theory: Given an undirected graph $G = (V, E)$, where V is the set of vertices of G and E is the set of edges, and a node $a \in V$ of this graph, does there exist a path of G starting at a and passing through each node in V exactly once?

Suppose that the graph G is specified by two predicates $node(X)$ and $arc(X, Y)$, and the starting node is specified by the predicate *start* which contains only a single tuple. Then, the following program solves the problem HAMPATH:

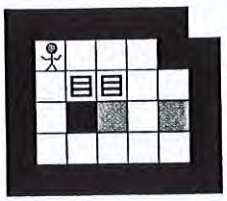
$reached(X)$:- $start(X)$.
$reached(X)$:- $inPath(Y, X)$.
:- $node(X)$, not $reached(X)$.
$inPath(X, Y) \vee outPath(X, Y)$:- $reached(X), arc(X, Y)$.
:- $inPath(X, Y), inPath(X, Y1), Y <> Y1$.
:- $inPath(X, Y), inPath(X1, Y), X <> X1$.

Blocksworld (BW) is a classic problem from the planning domain, and one of the oldest problems in AI: Given a table and a number of blocks in a (known) initial state and a desired goal state, try to reach that goal state by moving one block at a time such that each block is either on top of another block or the table at any given time step.

Due to space restrictions we refer to [Erdem, 1999, Faber et al., 1999a] for a complete encoding. Figure 4 shows a simple instance that can be solved in three time steps: First we move block c to the table, then block b on top of a, and finally c on top of b.



(a) BW Instance



(b) SOKO Instance

Figure 4: BW and SOKO Instances

Sokoban (SOKO) is a game puzzle which has been developed by the Japanese company "Thinking Rab-

bit, Inc." in 1982. "Sokoban" means "warehouse-keeper" in Japanese. Each puzzle consists of a room layout (a number of square fields representing walls or parts of the floor, some of which are marked as storage space) and a starting situation (one sokoban and a number of boxes, all of which must reside on some floor location). The goal is to move all boxes onto storage locations. To this end, the sokoban can walk on floor locations (unless occupied by some box), and push single boxes onto unoccupied floor locations. Figure 4 shows a typical configuration involving two boxes, where grey fields are storage fields and black fields are walls.

We have written a **DLV** program which finds solutions with a given number of push actions (where one push action can move a box over a number of fields) for a given puzzle together with a script which iteratively runs that **DLV** program with increasing numbers of push actions (starting at one) until some solution is found. In this way solutions with a minimal number of push actions are found.

The puzzle in Figure 4 is solvable with 6 push actions, so the script uses **DLV** to prove that no solutions with 1–5 push actions exist, and then to compute a solution with 6 push actions.

7.3 BENCHMARK DATA

Random graph instances for HAMPATH were generated using a tool[4] by Patrik Simons which has been used to compare Smodels against SAT solvers [Simons, 2000]. For each problem size n we generated *ten* instances, always assuming node 0 as the starting node, and for each instance we stopped after the first solution had been found.

The blocksworld problems P1 to P4 have been employed in [Erdem, 1999] to compare ASP systems, and can be solved in 4, 6, 8 and 9 steps, respectively. We augmented these by problems P5 and P6 which require 11 and 12 steps, respectively. For each of these problems, we generated 8 random permutations of the input. In addition, we also tried to solve each of these problems with one step less than required, which fails to produce any plan but shows the minimality of the regular solutions. These instances are labeled P1-1, P2-1 and so forth in Figure 5.

A vast amount of Sokoban puzzles is available on the Internet in a simple ANSI text format. The examples we used for benchmarks are results of efforts to automatically generate hard puzzles. One set has been created by Yoshio Murase[5], the other set is by Jacques Duthen[6]. The puzzle in Figure 4 is number 2 of Duthen's instances.

[4] `http://tcs.hut.fi/Software/smodels/misc/hamilton.tar.gz` .

7.4 EXPERIMENTAL RESULTS

All experiments were performed on a Pentium III/733 machine with 256MB of main memory running SuSE GNU/Linux (kernel 2.2.14).

For each invocation of **DLV** we allowed a maximum run-time of 600 seconds. For SOKO we have several invocations per problem instance, so the total reported time may be more than 600 seconds. Average running times for HAMPATH and BW are displayed in Figure 5.

	Old	ifPoss	ifNeed
10	0.02	0.02	0.01
20	0.04	0.04	0.04
30	0.19	0.08	0.07
40	0.11	0.12	0.12
50	-	0.18	0.18
60	-	0.71	0.64
70	-	0.34	0.33
80	-	0.43	0.41
90	-	0.61	0.56
100	-	0.68	0.66
110	-	0.82	0.78
120	-	1.21	1.19

(a) Hamiltonian Path

	Old	ifPoss	ifNeed
P1 -1	0.02	0.02	0.02
P2 -1	0.04	0.04	0.04
P3 -1	1.58	1.66	1.57
P4 -1	1.26	1.33	1.30
P5 -1	9.48	9.84	9.47
P6 -1	146.92	147.13	146.90
P1	0.02	0.02	0.02
P2	0.06	0.07	0.06
P3	4.11	5.17	4.15
P4	10.56	10.18	10.66
P5	174.21	187.64	174.17
P6	166.12	170.13	166.08

(b) Blocksworld

Figure 5: Average Running Times

On HAMPATH, both **ifPoss** and **ifNeed** perform similarly to **Old** for small problem instances, but scale tremendously better and are able to efficiently deal with graphs of 120 nodes, whereas **Old** cannot solve problems with more than 40 nodes. On average, **ifNeed** slightly outperforms **ifPoss**. These benchmark programs have highly cyclic HCF dependency graphs.

[5] `http://www.ne.jp/asahi/ai/yoshio/sokoban/auto52/auto52.htm` .

[6] `http://hem.passagen.se/awl/ksokoban/sokogen-990602.skm` .

Thus, **ifNeed** and **ifPoss** can exploit the pruning power of the well-founded operator, significantly outperforming **Old** which employs only the Fitting operator. On the other hand, the dependency graphs of these programs usually have one big component containing nearly all atoms. Therefore, there is nearly no difference between **ifNeed** and **ifPoss**, as the former cannot avoid many calls to the well-founded operator.

For BW, **Old** and **ifNeed** are nearly equivalent, and both outperform **ifPoss** by a few percent, though all three approaches seem to scale similarly. We explain this as follows: These programs have only few cyclic HCF components while most components are acyclic. Moreover, these few cyclic components are also very small, and the well-founded operator does not bring a relevant gain in terms of pruning compared to the Fitting operator (so **Old** and **ifNeed** show essentially the same behavior). **ifPoss** pays a computational overhead w.r.t. the other methods, because it is needlessly invoked several times on acyclic components.

SOKO, finally, shows that both **ifPoss** and **ifNeed** perform significantly better than **Old**, which fails to solve more than 50% of all problems instances and usually takes one or two orders of magnitude longer to solve the remaining ones:

	Yoshio Murase		Jacques Duthen	
	solved	unsolved	solved	unsolved
Old	12	40	38	40
ifPoss	41	11	68	10
ifNeed	41	11	69	9

For the Yoshio Murase set, **ifNeed** yields an average speedup of 18.45% over **ifPoss**, the maximum speedup being 60.67% (instance 9). On the Jacques Duthen set, **ifNeed** yields an average speedup of 15% over **ifPoss**, the maximum speedup being 67% (instance 62). A full account of the Sokoban test results is given in a longer version which has been published as a Technical Report [Calimeri et al., 2001].

In summary, our benchmarks show that both **ifPoss** and **ifNeed** are strictly preferable to **Old**, and that of these two, **ifNeed** shows a measurable speedup on a wide range of examples. Therefore the latest **DLV** release employs **ifNeed** by default.

For future work, we plan to focus on (automatically) improving the choice of pruning operators and tuning the actual implementations of these operators. Another perspective for future work would be to evaluate the applicability and impact of even "stronger" operators, e.g. the $\mathcal{F}_S$ operator introduced in [Baral, 1992].

Yet another issue to be considered is the fact that dependency graphs in **DLV** are currently static. They could be made dynamic in the sense that only currently undefined atoms are considered in the graphs. This could have the advantage that non-HCF components could become HCF and cyclic components could become acyclic during the computation, allowing for more efficient algorithms on these components. But it is unclear how the computational cost of maintaining these graphs relates to possible savings.

Acknowledgments

This work was supported by the Austrian Science Fund (FWF) under grants P14781-INF and Z29-INF, and by the European Commission under project ICONS, project no. IST-2001-32429.

References

[Anger et al., 2001] Anger, C., Konczak, K., and Linke, T. (2001). NoMoRe: A System for Non-Monotonic Reasoning. *LPNMR'01*, number 2173 in Lecture Notes in AI (LNAI), pp. 406–410.

[Baral, 1992] Baral, C. (1992). Generalized Negation As Failure and Semantics of Normal Disjunctive Logic Programs. *LPAR'92*, pp. 309–319.

[Bell et al., 1994] Bell, C., Nerode, A., Ng, R. T., and Subrahmanian, V. (1994). Mixed Integer Programming Methods for Computing Nonmonotonic Deductive Databases. *JACM*, 41:1178–1215.

[Ben-Eliyahu and Dechter, 1994] Ben-Eliyahu, R. and Dechter, R. (1994). Propositional Semantics for Disjunctive Logic Programs. *Annals of Mathematics and Artificial Intelligence*, 12:53–87.

[Calimeri, 2001] Calimeri, F. (2001). Progettazione e sviluppo di tecniche di ottimizzazione per sistemi di basi di conoscenza. Master's thesis, D.E.I.S., Università degli Studi della Calabria, Rende (CS), Italy.

[Calimeri et al., 2001] Calimeri, F., Faber, W., Leone, N., and Pfeifer, G. (2001). Pruning Operators for Answer Set Programming Systems. Technical Report INFSYS RR-1843-01-07, Institut für Informationssysteme, Technische Universität Wien, Austria.

[Dowling and Gallier, 1984] Dowling, W. F. and Gallier, J. H. (1984). Linear-time algorithms for testing the satisfability of propositional horn formulae. *Journal of Logic Programming*, 3:267–284.

[East and Truszczyński, 2000] East, D. and Truszczyński, M. (2000). dcs: An implementation of DATALOG with Constraints. *NMR'2000*, Breckenridge, Colorado, USA.

[Egly et al., 2000] Egly, U., Eiter, T., Tompits, H., and Woltran, S. (2000). Solving Advanced Reasoning Tasks using Quantified Boolean Formulas. In *AAAI'00*, pp. 417–422. AAAI Press / MIT Press.

[Eiter et al., 1997] Eiter, T., Gottlob, G., and Mannila, H. (1997). Disjunctive Datalog. *ACM Transactions on Database Systems*, 22(3):364–418.

[Erdem, 1999] Erdem, E. (1999). Applications of Logic Programming to Planning: Computational Experiments. Unpublished draft. `http://www.cs.utexas.edu/users/esra/papers.html`.

[Faber et al., 1999a] Faber, W., Leone, N., Mateis, C., and Pfeifer, G. (1999a). Using Database Optimization Techniques for Nonmonotonic Reasoning. *DDLP'99*, pp. 135–139. Prolog Association of Japan.

[Faber et al., 1999b] Faber, W., Leone, N., and Pfeifer, G. (1999b). Pushing Goal Derivation in DLP Computations. *LPNMR'99*, number 1730 in Lecture Notes in AI (LNAI), pp. 177–191.

[Faber et al., 2001] Faber, W., Leone, N., and Pfeifer, G. (2001). Experimenting with Heuristics for Answer Set Programming. In *IJCAI 2001*, pp. 635–640, Seattle, WA, USA. Morgan Kaufmann Publishers.

[Fages, 1994] Fages, F. (1994). Consistency of clark's completion and existence of stable models. *Journal of Methods of Logic in Computer Science*, 1(1):51–60.

[Fitting, 1985] Fitting, M. (1985). A Kripke-Kleene semantics for logic programs. *Journal of Logic Programming*, 2(4):295–312.

[Gelfond and Lifschitz, 1991] Gelfond, M. and Lifschitz, V. (1991). Classical Negation in Logic Programs and Disjunctive Databases. *New Generation Computing*, 9:365–385.

[Leone et al., 1993] Leone, N., Romeo, M., Rullo, P., and Saccà, D. (1993). Effective implementation of negation in database logic query languages. In *LOGIDATA+: Deductive Database with Complex Objects*, LNCS 701, pp. 159–175.

[Leone et al., 1997] Leone, N., Rullo, P., and Scarcello, F. (1997). Disjunctive stable models: Unfounded sets, fixpoint semantics and computation. *Information and Computation*, 135(2):69–112.

[Lifschitz, 1996] Lifschitz, V. (1996). Foundations of Logic Programming. *Principles of Knowledge Representation*, pp. 69–127. CSLI Publications, Stanford.

[Lifschitz and Turner, 1994] Lifschitz, V. and Turner, H. (1994). Splitting a Logic Program. *ICLP'94*, pp. 23–37. MIT Press.

[McCain and Turner, 1998] McCain, N. and Turner, H. (1998). Satisfiability Planning with Causal Theories. *KR'98*, pp. 212–223. Morgan Kaufmann Publishers.

[Minoux, 1988] Minoux, M. (1988). LTUR: a simplified linear-time unit resolution algorithm for Horn formulae and computer implementation. *Information Processing Letters*, 29:1–12.

[Niemelä, 1999] Niemelä, I. (1999). Logic programming with stable model semantics as constraint programming paradigm. *Annals of Mathematics and Artificial Intelligence*, 25(3–4):241–273.

[Niemelä and Simons, 1996] Niemelä, I. and Simons, P. (1996). Efficient Implementation of the Well-founded and Stable Model Semantics. *ICLP'96*, pp. 289–303, Bonn, Germany. MIT Press.

[Rao et al., 1997] Rao, P., Sagonas, K. F., Swift, T., Warren, D. S., and Freire, J. (1997). XSB: A System for Efficiently Computing Well-Founded Semantics. *LPNMR'97*, number 1265 in Lecture Notes in AI (LNAI), pp. 2–17, Dagstuhl, Germany.

[Scarcello, 1997] Scarcello, F. (1997). *Caratterizzazione e Calcolo di Modelli Stabili in Programmazione Logica Disgiuntiva.* PhD thesis, Università della Calabria, Rende (CS), Italy.

[Simons, 2000] Simons, P. (2000). *Extending and Implementing the Stable Model Semantics.* PhD thesis, Helsinki University of Technology, Finland.

[Subrahmanian et al., 1995] Subrahmanian, V., Nau, D., and Vago, C. (1995). WFS + Branch and Bound = Stable Models. *IEEE TKDE*, 7(3):362–377.

[Tarjan, 1972] Tarjan, R. (1972). Depth-first search and linear graph algorithm. *SIAM Journal of Computing*, 1(2).

[van Gelder et al., 1991] van Gelder, A., Ross, K., and Schlipf, J. (1991). The Well-Founded Semantics for General Logic Programs. *JACM*, 38(3):620–650.

More on noMoRe

Thomas Linke **Christian Anger** **Kathrin Konczak**

Universität Potsdam, Institut für Informatik
August-Bebel-Strasse 89, D- 14482 Potsdam
{linke,canger,konczak}@cs.uni-potsdam.de

Abstract

We focus on the efficient computation of answer sets for normal logic programs. We concentrate on a recently proposed rule-based method (implemented in the noMoRe system) for computing answer sets. We show how noMoRe and its underlying method can be improved tremendously by improving its deterministic consequences. With these improvements noMoRe is able to deal with problem classes it could not handle so far.

1 Introduction

Answer set programming (ASP) is a programming paradigm, which allows for solving problems in a compact and highly declarative way. The basic idea is to specify a given problem in a declarative language, e.g. normal logic programs[1], such that the different answer sets given by answer sets semantics [8] correspond to the different solutions of the initial problem [10]. As an example, consider the independent set problem, which is to determine if there exists a maximal (wrt set inclusion) independent subsets of nodes for a given graph. A subset $S \subseteq V$ of nodes of a graph $G=(V,E)$ is called *independent* if there are no edges between nodes in S. Let

$$P=\left\{\begin{array}{l} in(a) \leftarrow not\ in(d), not\ in(b) \\ in(b) \leftarrow not\ in(a), not\ in(c) \\ in(c) \leftarrow not\ in(b), not\ in(d) \\ in(d) \leftarrow not\ in(c), not\ in(a) \end{array}\right\} \quad (1)$$

be a logic program and let us call the rules r_a, r_b, r_c and r_d, respectively. Then program (1) encodes the independent set problem for graph $G = (\{a,b,c,d\}, \{(a,b),(b,c),(c,d),(d,a)\})$. Program (1) has two answer sets $X_1 = \{in(a), in(c)\}$ and $X_2 = \{in(b), in(d)\}$ corresponding to the two independent sets of graph G.

Currently there are reasonably efficient implementations (e.g. `smodels` [15] and `dlv` [5]) available as well as interesting applications of answer set programming. (e.g. [3, 13, 14]). Since computation of answer sets is NP-complete for normal logic programs (and Σ_2^P-complete for disjunctive logic programs), most algorithms rely on methods similar to the Davis-Putnam algorithm for SAT. That is, they contain a non-deterministic part (making choices) and a part computing deterministic consequences for these choices. Whereas in [7] different heuristics are investigated in order to make the "right" choices, in this paper we improve the deterministic consequences of the recently proposed rule-based noMoRe system [1]. For all ASP systems relying on Davis-Putnam-like algorithms (that is, for all systems mentioned so far), non-deterministic choices and deterministic consequences determine the behavior of the resulting algorithm. In particular, we redefine propagation of so-called a-colorings as introduced in [11] such that we are able to include *backward propagation*. Furthermore, we introduce a technique called *jumping* to ensure complete backward propagation and give experimental results showing the influence of the presented techniques.

2 Background

We deal with normal logic programs which contain the symbol *not* used for *negation as failure*. A *normal logic program* is a set of rules of the form $p \leftarrow q_1, \ldots, q_n, not\ s_1, \ldots, not\ s_k$ where p, q_i $(0 \leq i \leq n)$ and s_j $(0 \leq j \leq k)$ are propositional atoms. A rule is a *fact* if $n=k=0$, it is called *basic* if $k=0$ and *quasi-fact* if $n=0$. For a rule r like above we define $head(r)=p$ and $body(r) = \{q_1, \ldots, q_n, not\ s_1, \ldots, not\ s_k\}$. Fur-

[1] The language of normal logic programs is not the only one suitable for ASP. Others are disjunctive logic programs, propositional logic or DATALOG with constraints [4].

thermore, let $body^+(r)=\{q_1,\ldots,q_n\}$ denote the set of positive body atoms and $body^-(r)=\{s_1,\ldots,s_k\}$ the set of negative body atoms. We denote the set of all facts of a program P by $Facts(P)$ and the set of all atoms of P by $Atoms(P)$.

Let r be a rule. r^+ then denotes the rule $head(r) \leftarrow body^+(r)$, obtained from r by deleting all negative body atoms in the body of r. For a logic program P let $P^+ = \{r^+ \mid r \in P\}$. A set of atoms X is *closed under* a basic program P iff for any $r \in P$, $head(r) \in X$ whenever $body(r) \subseteq X$. The smallest set of atoms which is closed under a basic program P is denoted by $\mathrm{Cn}(P)$. The *reduct*, P^X, of a program P *relative to* a set X of atoms is defined by $P^X = \{r^+ \mid r \in P \text{ and } body^-(r) \cap X = \emptyset\}$. We say that a set X of atoms is an *answer set* of a program P iff $\mathrm{Cn}(P^X)=X$.

A set of rules P is *grounded* iff there exists an enumeration $\langle r_i\rangle_{i\in I}$ of P such that for all $i \in I$ we have that $body^+(r_i) \subseteq head(\{r_1,\ldots,r_{i-1}\})$. Observe that there exists a unique maximal grounded set $P' \subseteq P$ for each program P.[2] For a set of rules P and a set of atoms X we define the set of generating rules of P wrt X as $GR(P,X) = \{r \in P \mid body^+(r) \subseteq X, body^-(r) \cap X = \emptyset\}$. Then X is an answer set of P iff we have $X=\mathrm{Cn}(GR(P,X)^+)$. This characterizes answer sets in terms of generating rules. Observe, that in general $GR(P,X)^+ \neq P^X$ (take $P=\{a \leftarrow, b \leftarrow c\}$ and $X=\{a\}$).

We need some graph theoretical terminology . A *directed graph* (or *digraph*) G is a pair $G=(V,A)$ such that V is a finite, non-empty set (vertices) and $A \subseteq V \times V$ is a set (arcs). For a digraph $G=(V,A)$ and a vertex $v \in V$, we define the set of all *predecessors* of v as $\mathrm{Pred}(v)=\{u \mid (u,v)\in A\}$. Analogously, the set of all *successors* of v is defined as $\mathrm{Succ}(v)=\{u \mid (v,u)\in A\}$. A *path from v to v'* in $G=(V,A)$ is a finite subset $P_{vv'} \subseteq V$ such that $P_{vv'}=\{v_1,\ldots,v_n\}$, $v=v_1$, $v'=v_n$ and $(v_i,v_{i+1}) \in A$ for each $1 \leq i < n$. Let $G=(V,A)$ and $G'=(V',A')$ be digraphs. Then G' is a *subgraph* of G if $V' \subseteq V$ and $A' \subseteq A$. G' is an *induced subgraph* of G if G' is a subgraph of G s.t. for each $v,v' \in V'$ we have that $(v,v') \in A'$ iff $(v,v') \in A$.

In order to represent more information in a directed graph, we need a special kind of arc labeling. $G=(V,A^0 \cup A^1)$ is a directed graph whose arcs $A^0 \cup A^1$ are labeled zero (A^0) and one (A^1). We call arcs in A^0 and A^1 *0-arcs* and *1-arcs*, respectively. For G we distinguish 0-predecessors (0-successors) from 1-predecessors (1-successors) denoted by $\mathrm{Pred0}(v)$ ($\mathrm{Succ0}(v)$) and $\mathrm{Pred1}(v)$ ($\mathrm{Succ1}(v)$) for $v \in V$, respectively. A path $P_{vv'}$ in G is called *0-path* if $Arcs(P_{vv'}) \subseteq A^0$.

[2] Here we mean sets P' which are maximal wrt set inclusion and grounded.

Block Graphs for Normal Logic Programs

Next we summarize the central definitions of block graphs for logic programs and a-colorings of block graphs (cf. [11, 12]).

Definition 1 *Let P be a logic program and let P' be the maximal grounded subset of P. The* block graph $\Gamma_P=(V_P, A^0_P \cup A^1_P)$ *of P is a directed graph with vertices $V_P=P$ and two different kinds of arcs defined as follows*

$$A^0_P=\{(r',r) \mid r',r\in P' \text{ and } head(r')\in body^+(r)\}$$
$$A^1_P=\{(r',r) \mid r',r\in P' \text{ and } head(r')\in body^-(r)\}.$$

This definition captures the conditions under which a rule r' blocks another rule r (i.e. $(r',r) \in A^1$). We introduce an 1-arc (r',r) in Γ_P if $r' = (q \leftarrow \ldots)$ and $r = (\ldots \leftarrow \ldots, not\ q, \ldots)$. We also gather all groundedness information in Γ_P, because we only introduce a 0-arc (r',r) (between rules $r' = (q \leftarrow \ldots)$ and $r=(\ldots \leftarrow q,\ldots)$) if r and r' are in the maximal grounded subset of P.[3] Figure 1 shows the block graph of program (1). Observe, that operations $head(r)$, $body^+(r)$ and $body^-(r)$ (for $r \in P$) are operations on the block graph, since the nodes of Γ_P are the rules of logic program P.

In order to define so-called *application colorings* or *a-colorings* for block graphs we need the following definition.

Definition 2 *Let P be a logic program and let $\Gamma_P = (V_P, A^0_P \cup A^1_P)$ the corresponding block graph. Furthermore, let $x \in Atoms(P)$ and let $G_x=(V_x,A_x)$ be a subgraph of Γ_P. Then G_x is a* grounded 0-graph for x in Γ_P *iff the following conditions hold:*

1. *G_x is an acyclic subgraph of Γ_P s.t. $A_x \subseteq A^0_P$*
2. *G_x contains a target node r_x s.t. $x=head(r_x)$ and from every other node there exists a 0-path to the target node*
3. *for each node $r \in V_x$ we have $body^+(r)=\emptyset$ or for each $q' \in body^+(r)$ there exists a node $r' \in V_x$ s.t. $q'=head(r')$ and $(r',r) \in A_x$.*

[3] Observe, that for program $P=\{p \leftarrow q, q \leftarrow p\}$ the maximal grounded subset of rules is empty and therefore Γ_P contains no 0-arcs.

Observe, that the nodes of a grounded 0-graph are grounded according to definition (see Section 2). Furthermore, the different grounded 0-graphs for atom x in Γ_P correspond to the different classical "proofs" for x in P^+, ignoring the default negations of all rules.

Definition 3 *Let P be a logic program, let $\mathcal{C} : P \mapsto \{\ominus, \oplus\}$ be a total mapping*[4]. *We call r* grounded wrt Γ_P and $\mathcal{C}$ *iff for each $q \in body^+(r)$ there exist a grounded 0-graph $G_q = (V_q, A_q))$ for q in Γ_P s.t. $\mathcal{C}(V_q) = \oplus$.*

A rule r is called blocked wrt Γ_P and $\mathcal{C}$ *if there exists some $r' \in Pred1(r)$ s.t. $\mathcal{C}(r') = \oplus$*[5].

Now we are ready to define a-colorings.

Definition 4 *Let P be a logic program, let Γ_P be the corresponding block graph and let $\mathcal{C} : P \mapsto \{\ominus, \oplus\}$ be a total mapping. Then $\mathcal{C}$ is an* a-coloring *of Γ_P iff the following condition hold for each $r \in P$*

AP *$\mathcal{C}(r) = \oplus$ iff r is grounded and r is not blocked wrt Γ_P and $\mathcal{C}$.*

Let $\mathcal{C}$ be an a-coloring of some block graph Γ_P. Rules are then intuitively applied wrt some answer set of P if and only if they are colored $\oplus$, that is, condition **AP** captures the intuition of applying a rule wrt to some answer set. Similarly, the negation of condition **AP** (r is **not** grounded **or** r is blocked) captures the intuition when a rule is not applicable wrt to some answer set.

Observe, that there are programs for which the corresponding block graph has no a-coloring and thus no answer set. Let r_p be rule $p \leftarrow not\ p$. Then program $P = \{r_p\}$ has block graph $(P, \emptyset, \{(r_p, r_p)\})$, that is, Γ_P consists of a single 1-loop. By Definition 4 there is no a-coloring of Γ_P. If we color r_p with $\oplus$ we get a direct contradiction to **AP**, since then r_p is blocked. On the other hand, if we color r_p with $\ominus$ then r_p is trivially grounded and not blocked. Therefore r_p has to be colored $\oplus$ which again is a contradiction.

The main result in [11] states that Program P has an answer set X iff Γ_P has an a-coloring $\mathcal{C}$ s.t.

$$GR(P, X) = \{r \in P \mid \mathcal{C}(r) = \oplus\}.$$

This result constitutes a rule-based method to compute answer sets by computing a-colorings. In Figure 1 we have depicted the two a-colorings of the block graph of program (1) left and right from '/', respectively.

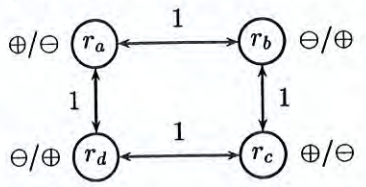


Figure 1: Block graph and a-colorings of program (1).

3 Propagation

In the **nonmo**notonic **R**easoning system **noMoRe** [1] the approach described in the last section is implemented. Let us assume that each program is grounded. In order to describe the deterministic part of the implementation and its improvements, we need some central properties of nodes. All those properties are defined wrt partial a-colorings. We call a partial mapping $\mathcal{C} : P \mapsto \{\ominus, \oplus\}$ a *partial a-coloring* if $\mathcal{C}$ is an a-coloring of the induced subgraph of Γ_P with nodes $Dom(\mathcal{C})$[6].

Definition 5 *Let P be a logic program and let $\mathcal{C}$ be a partial a-coloring of Γ_P. For each node $r \in P$ we define the following properties wrt Γ_P and $\mathcal{C}$:*

1. p-grounded(r) *iff* $\forall q \in body^+(r) :$ $\exists r' \in Pred0(r) : q = head(r')$ *and* $\mathcal{C}(r') = \oplus$
2. p-notgrounded(r) *iff* $\exists q \in body^+(r) :$ $\forall r' \in Pred0(r) : q \neq head(r')$ *or* $\mathcal{C}(r') = \ominus$
3. p-blocked(r) *iff* $\exists r' \in Pred1(r) : \mathcal{C}(r') = \oplus$
4. p-notblocked(r) *iff* $\forall r' \in Pred1(r) : \mathcal{C}(r') = \ominus$.

Notice the difference between total and partial a-colorings. For example, if p-notgrounded(r) holds for r wrt some total a-coloring $\mathcal{C}$ then p-grounded(r) is the negation of p-notgrounded(r). This does not hold for partial a-coloring $\mathcal{C}$, since there may be nodes for which $\mathcal{C}$ is undefined. For this reason, we have to define both p-grounded (p-blocked) and p-notgrounded (p-notblocked), respectively, because they cannot be defined through each other wrt partial a-colorings. However, we have the following result for total a-colorings:

Theorem 1 *Let P be a logic program and let $\mathcal{C}$ be a total a-coloring of Γ_P. Then for each node $r \in P$ we have r is grounded wrt Γ_P and $\mathcal{C}$ iff p-grounded(r) wrt*

[4] A mapping $\mathcal{C} : P \mapsto C$ is called *total* iff for each node $r \in P$ there exists some $\mathcal{C}(r) \in C$. Oppositely, mapping $\mathcal{C}$ is called partial if there are some $r \in P$ for which $\mathcal{C}(r)$ is undefined.

[5] We say a rule is grounded or blocked (omitting $\mathcal{C}$) if it is clear from the context with respect to which mapping $\mathcal{C}$ the rule is grounded or blocked, respectively.

[6] $Dom(\mathcal{C})$ denotes the domain of mapping $\mathcal{C}$.

Γ_P *and* C. *Furthermore, we have* r *is blocked wrt* Γ_P *and* C *iff p-blocked*(r) *wrt* Γ_P *and* C.

Clearly, to be grounded (wrt Γ_P and C for node r) is a global concept wrt Γ_P whereas p-grounded(r) is defined locally wrt Γ_P. Furthermore, observe the difference between r is blocked and p-blocked(r) wrt Γ_P and C. Even if the definitions of both concepts are the same (cf Definition 3), the former is defined wrt to total a-colorings, whereas the later one is defined wrt partial a-colorings. In a situation like in Figure 2 we do not have p-blocked(r') and we do not have p-notblocked(r') wrt the depicted partial a-coloring. But we always have either that r is blocked or not blocked wrt total a-colorings.

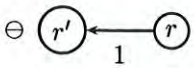


Figure 2: Some block graph with partial a-coloring.

Definition 6 *Let* C *be a partial a-coloring of* Γ_P *and let* U *be the set of uncolored nodes wrt* C. *Then each node* $r \in U$ *can be* colored $\oplus$ by propagation of C *iff we have p-grounded*(r) *and p-notblocked*(r) *wrt* C. *Node* r *can be* colored $\ominus$ by propagation of C *iff we have p-notgrounded*(r) *or p-blocked*(r) *wrt* C.

Notice, that propagation of partial a-colorings to uncolored nodes is global wrt Γ_P, since in order to propagate C as much as possible we have to check all nodes in U which in general are distributed over Γ_P. According to Definitions 5 and 6 nodes colored by propagation always have colored predecessors. Therefore we obtain a more procedural way to propagate partial a-colorings by *localized* propagation conditions.

Definition 7 *Let* P *be a logic program, let* Γ_P *be the corresponding block graph and let* C *be a partial a-coloring of* Γ_P. *We define an extented mapping* C^e *of* C *s.t. for each* $r \in Dom(C)$ *we have* $C^e(r) = C(r)$ *and for each* $r, r' \in P$ *the following conditions hold wrt* Γ_P *and* C^e:

(A) *if* $r \in Succ1(r')$ *and* $C^e(r') = \oplus$ *then* $C^e(r) = \ominus$

(B) *if* $r \in Succ1(r')$ *and* $C^e(r') = \ominus$ *and p-notblocked*(r) *and p-grounded*(r) *then* $C^e(r) = \oplus$

(C) *if* $r \in Succ0(r')$ *and* $C^e(r') = \oplus$ *and p-notblocked*(r) *and p-grounded*(r) *then* $C^e(r) = \oplus$

(D) *if* $r \in Succ0(r')$ *and* $C^e(r') = \ominus$ *and p-notgrounded*(r) *then* $C^e(r) = \ominus$.

We have the following result:

Theorem 2 *Let* P *be a logic program and let* C *and* $C^e : P \mapsto \{\ominus, \oplus\}$ *be partial mappings. Then we have if* C *is a partial a-coloring of* Γ_P *and* C^e *is an extension of* C *as in Definition 7 then* $C_\ominus \subseteq C^e_\ominus$, $C_\oplus \subseteq C^e_\oplus$ *and* C^e *is a partial a-coloring of* Γ_P.

This theorem gives the conditions for four different propagation cases in arc direction: if a node r is colored c ($c \in \{\ominus, \oplus\}$) then this color can be propagated over 1- and over 0-arcs to the neighbors of r, according to localized propagation conditions (A), (B), (C) and (D).

Now let P be some logic program. Let $C : P \mapsto \{\ominus, \oplus\}$ be a partial a-coloring. C is represented by a pair of (disjoint) sets $(C_\ominus, C_\oplus)$ s.t. $C_\ominus = \{r \in P \mid C(r) = \ominus\}$ and $C_\oplus = \{r \in P \mid C(r) = \oplus\}$. Since C is not total we do not necessarily have $P = C_\ominus \cup C_\oplus$. We refer to mapping C with the pair $(C_\ominus, C_\oplus)$ and vice versa. Assume that Γ_P is a global parameter of each of the presented procedure. Let U and N be sets of nodes s.t. U contains the currently uncolored nodes $U = P \setminus (C_\ominus \cup C_\oplus)$ and N contains colored nodes whose color has to be propagated. Figures 3, 4 and 5 show the main procedures of noMoRe in pseudo code. Notice that procedures **color**$_P$ and **propagate**$_P$ return some extended partial mapping through parameter C or fail. Procedure **choose**$_P$ either returns some uncolored node r s.t. we have p-grounded(r) wrt the current partial a-coloring or fails. Clearly, **choose**$_P$ implements the non-deterministic part of **color**$_P$. Oppositely, **propagate**$_P$ implements the deterministic consequences of noMoRe. Let $L_1(P) = \{r \in P \mid r \in \text{Pred1}(r)\}$ denote the set of all 1-loops in Γ_P. When calling **color**$_P$ the first time, we start with

```
function colorP(U, N:list; C:partial mapping)
 var r : node
  if propagateP(N,C) then
    U := U \ (C⊖ ∪ C⊕)
    if chooseP(U,C,r) then
      U := U \ {r}
      if colorP(U, {r}, (C⊖, C⊕ ∪ {r})) then
        return true
      else
        return colorP(U, {r}, (C⊖ ∪ {r}, C⊕))
    else
      return propagateP(U, (C⊖ ∪ U, C⊕))
  else
    return false
```

Figure 3: Basic noMoRe procedure.

```
function propagate_P(N:list, C:partial mapping)
 var r' : node  t : boolean
   t := true
   while (N ≠ ∅ and t = true) do
     select r' from N
     if (r' ∈ C_⊕) then
(A)    t := (propA_P(r', C) and
(C)          propC_P(r', C))
     else
(B)    t := (propB_P(r', C) and
(D)          propD_P(r', C))
   return t
```

Figure 4: Procedure $\mathbf{propagate}_P$.

$\mathcal{C} = (L_1(P), Facts(P))$, $U = P \setminus (Facts(P) \cup L_1(P))$ and $N = Facts(P) \cup L_1(P)$. That is, we start with all facts colored $\oplus$ and all 1-loops colored $\ominus$. Basically, $\mathbf{color}_P$ takes both a partial mapping $\mathcal{C}$ and a set of uncolored nodes U and aims at coloring these nodes. That is, $\mathbf{color}_P$ computes an extended partial mapping or if this is impossible it fails. This is done by choosing some uncolored node r ($r \in U$) with $\mathbf{choose}_P$ and by trying to color it $\oplus$ first. If this does not give a solution $\mathbf{color}_P$ tries to color node r with $\ominus$. If both possibilities fail $\mathbf{color}_P$ fails. Therefore, we say that node r is used as a *choice*. To be a choice is not a property of a node, because choices are dynamic wrt each solution. Therefore a node may be a choice in one run of $\mathbf{color}_P$ but not in every run, e.g. because there may be heuristics that uses different nodes as choices depending on the actual implementation of $\mathbf{choose}_P$. Observe, that all different a-colorings are obtained via backtracking over choices in $\mathbf{color}_P$.

Notice, that procedure $\mathbf{propagate}_P$ works locally according to conditions (A) to (D) of Definition 7. The color of a node is propagated immediately after getting colored, because the test whether the node was colored correctly is done during propagation. Observe that Theorem 4.1 in [12] implies that partial a-coloring $\mathcal{C}$ can not be extented to some total a-coloring if $\mathbf{propagate}_P$ fails. Therefore, $\mathbf{color}_P$ fails only during propagation. Procedures $\mathbf{propA}_P$, $\mathbf{propB}_P$, $\mathbf{propC}_P$, and $\mathbf{propD}_P$ are the implementations of (A), (B), (C) and (D), respectively. The purpose of $\mathbf{propagate}_P$ is to apply the corresponding propagation cases, e.g. if $\mathcal{C}(r') = \oplus$ then cases (A) and (C) have to be applied[7]. Accoring to Figure 3, partial a-colorings are propagated only in arc direction. That is, if some node is colored then we try to propagate the color of this node to its successors (if possible).

[7]The missing three procedures also used in $\mathbf{propagate}_P$ can be easily implemented analogously to $\mathbf{propB}_P$.

```
function propB_P(r' : node; C : partial mapping)
 var r : node  S : set of nodes  t : boolean
 S := {r ∈ Succ1(r') | condition (B) holds for r}
 t := true
 while (S ≠ ∅ and t = true) do
   select r from S
   if r ∈ C_⊖ then
     return false
   else
   if r ∉ C_⊕ then
     t := propagate_P(r, (C_⊖, C_⊕ ∪ {r}))
   else
     return false
 return t
```

Figure 5: Procedure $\mathbf{propB}_P$.

4 Backward Propagation

In [11, 1] it is stated that the number of choices can be reduced by introducing backward propagation, that is, partial a-colorings can also be propagated in opposite arc direction. Clearly, as for propagation in arc direction, we have four backward propagation cases. However, there is a problem with defining localized conditions for backward propagation (as in Definition 7). Assume that Figure 6 depicts a part of some block

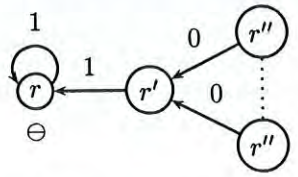


Figure 6: Part of some block graph with partial a-coloring.

graph together with some partial a-coloring. On the one hand, we know that r' has to be colored $\oplus$ (provided that there are no other predecessores of r), becauses this is the only way to block r and if r is not blocked there is no answer set. On the other hand, we cannot color r' with $\oplus$, because we do not have p-grounded(r') (see Definition 6). Therefore we need so-called *transitory a-colorings* .

Definition 8 *Let P be some logic program. We call a partial mapping $\mathcal{C} : P \mapsto \{\ominus, \oplus, +\}$ a* transitory a-coloring of Γ_P *iff $\mathcal{C}$ is an a-coloring of the induced subgraph of Γ_P with nodes $\mathcal{C}_\ominus \cup \mathcal{C}_\oplus$.*

That is, a transitory a-coloring is a partial a-coloring where some nodes may be uncolored or colored with $+$. Color $+$ is used instead of $\oplus$ to color node r' in situations like in Figure 6, where we do not have p-grounded(r') wrt the current partial a-coloring, but r' can still possibly be grounded. In order to transform some transitory a-coloring (during the execution of $\mathbf{color}_P$) to a total a-coloring color $+$ is replaced by color $\oplus$, if possible. This is achieved by propagation. Whenever a node is colored, this color is propagated to all its neighbors immediately, no matter whether these already have been colored or not. In case a node already colored $+$ ($\oplus$) has to be colored $\ominus$ via propagation, propagation fails due to contradiction. When a node already colored $+$ has to be colored $\oplus$ via propagation, color $+$ is simply replaced by $\oplus$. That is, either every color $+$ will become $\oplus$, or $\mathbf{color}_P$ fails. We need the following properties wrt transitory a-colorings:

Definition 9 *Let P be a logic program, let $\mathcal{C}$ be a transitory a-coloring of Γ_P and let $r \in P$ be some node. Then r is* groundable(r) wrt Γ_P and $\mathcal{C}$ *iff $\forall q \in body^+(r) : \exists r' \in Pred0(r)$ with $q = head(r')$ s.t. $\mathcal{C}(r') = \oplus$ or r' is uncolored.*

Here groundable(r) means that either r is grounded or that there is some uncolored 0-predecessor, which can possibly be colored $\oplus$ while extending $\mathcal{C}$. For $r \in P$ and $q \in body^+(r)$ we define $S_q \subseteq \text{Pred0}(r)$ as $S_q = \{r' \mid r' \in \text{Pred0}(r)$ and $q = head(r')\}$. Furthermore, for a set of rules $S \subseteq P$ we define p-grounded(S) wrt Γ_P and transitory a-coloring $\mathcal{C}$ iff there is some $r \in S$ s.t. $\mathcal{C}(r) = \oplus$. Now we are ready to define the four localized backward propagation cases.

Definition 10 *Let P be a logic program, let Γ_P be the corresponding block graph and let $\mathcal{C}$ be a transitory a-coloring of Γ_P. We define an extented mapping $\mathcal{C}^e : P \mapsto \{\ominus, \oplus, +\}$ of $\mathcal{C}$ s.t. for each $r \in Dom(\mathcal{C})$ we have $\mathcal{C}^e(r) = \mathcal{C}(r)$ and conditions (A) to (D) of Definition 7 and the following conditions hold for all $r, r' \in P$ wrt Γ_P and $\mathcal{C}^e$:*

(bA) *if $\mathcal{C}^e(r') = \oplus$ and $r \in Pred1(r')$ then $\mathcal{C}^e(r) = \ominus$*

(bB) *if $\mathcal{C}^e(r') = \ominus$ and p-grounded(r') and $r \in Pred1(r')$ s.t. and $\forall r'' \in Pred1(r') : (\mathcal{C}^e(r'') = \ominus$ iff $r'' \neq r)$ then $\mathcal{C}^e(r) = +$*

(bC) *if $\mathcal{C}^e(r') = \oplus$ and there is some $q \in body^+(r')$ s.t. $q = head(r)$ for some $r \in S_q$ and groundable(r) and for each $r'' \in S_q : (\mathcal{C}^e(r'') = \ominus$ iff $r'' \neq r)$ then $\mathcal{C}^e(r) = +$*

(bD) *if $\mathcal{C}^e(r') = \ominus$ and p-notblocked(r') and there is some $q \in body^+(r')$ with $q = head(r)$ for some $r \in S_q$ s.t. for each $q' \in body^+(r')$: (p-grounded($S_{q'}$) iff $S_{q'} \neq S_q$) then $\mathcal{C}^e(r) = \ominus$.*

Intuitively, these cases ensure that an already $\oplus$-colored node is grounded (bC) and not blocked (bA) while an already $\ominus$-colored node is blocked (bB) or not grounded (bD). So in a sense, the purpose of these cases is to justify the color of a node. Observe, that cases (bB) and (bC) use color $+$ instead of $\oplus$ (see Defintion 8). We have the following result corresponding to Theorem 2:

Theorem 3 *Let P be a logic program and let $\mathcal{C}$ and $\mathcal{C}^e : P \mapsto \{\ominus, \oplus, +\}$ be a partial mappings. Then we have if $\mathcal{C}$ is a transitory a-coloring of Γ_P and $\mathcal{C}^e$ is an extension of $\mathcal{C}$ as in Definition 10 then $\mathcal{C}_\ominus \subseteq \mathcal{C}^e_\ominus$, $\mathcal{C}_\oplus \subseteq \mathcal{C}^e_\oplus$ and $\mathcal{C}^e$ is a transitory a-coloring of Γ_P.*

Let us show how $\mathbf{color}_P$ computes the a-colorings of the block graph of program (1) (see Figure 1). At the beginning we cannot propagate anything, because there is no fact and no 1-loop. We take r_a as a choice. First, we try to color r_a with $\oplus$ by calling $\mathbf{color}_P(U, N, \mathcal{C})$ with $U = P \setminus \{r_a\}$, $N = \{r_a\}$ and $\mathcal{C} = (\emptyset, \{r_a\})$. Now, $\mathbf{propagate}_P(N, \mathcal{C})$ is executed. By propagating $\mathcal{C}(r_a) = \oplus$ with case (A) we get $\mathcal{C}(r_b) = \ominus$ and $\mathcal{C}(r_d) = \ominus$. Recursively, through case (B) $\mathcal{C}(r_c) = \oplus$ is propagated. This gives $\mathcal{C} = (\{r_b, r_d\}, \{r_a, r_c\})$. Since U becomes the empty set, $\mathbf{choose}_P$ fails and $\mathcal{C}$ is the first output. So far we did not need backward propagation.

Now, we color r_a with $\ominus$ through calling $\mathbf{color}_P(U, N, \mathcal{C})$ with U and N as above and $\mathcal{C} = (\emptyset, \{r_a\})$. Since no (backward) propagation is possible we have to compute the next choice. For $\mathbf{choose}_P$ all three uncolored nodes are possible choices s.t. **CP** holds. Assume $\mathcal{C}(r_b) = \oplus$ as next choice. Through propagation case (A) we get $\mathcal{C}(r_c) = \ominus$. This color of r_c has to be propagated by executing $\mathbf{propagate}_P(\{r_c\}, (\{r_a, r_c\}, \{r_c\}))$. By using propagation case (B) we obtain $\mathcal{C}(r_d) = \oplus$. Recursively, propagation of $\oplus$ for r_d gives no contradiction and $\mathcal{C} = (\{r_a, r_c\}, \{r_b, r_d\})$ is the second a-coloring. By assuming $\mathcal{C}(r_b) = \ominus$, that is, $\mathcal{C} = (\{r_a, r_b\}, \emptyset)$, r_c is colored with $\oplus$ through backward propagation case (bB). By propagation of this color with case (A) node r_d is colored with $\ominus$. By applying case (B) to the color of r_d we obtain that r_a has to be colored with $\oplus$, because it is not blocked, but this is a contradiction to $\mathcal{C}(p_a) = \ominus$. Thus, there is no further solution and we have found the two solutions with two choices. Observe that the usage of (bB) saves one additional choice, since without backward propagation the partial coloring $\mathcal{C} = (\{r_a, r_b\}, \emptyset)$ could not have been

extended any more and another choice would have been necessary.

5 Jumping

```
procedure propagate_P(N:list, C:part. mapping)
 var r' : node;
   while N ≠ ∅ do
     select r' from N;
     if (r' ∈ C_⊕) then
(A)    if propA_P(r', C) fails then fail;
(C)    if propC_P(r', C) fails then fail;
(bA)   if backpropA_P(r', C) fails then fail;
(bC)   if backpropC_P(r', C) fails then fail;
       if jumpC_P(r', C) fails then fail;
     else
(B)    if propB_P(r', C) fails then fail;
(D)    if propD_P(r', C) fails then fail;
(bB)   if backpropB_P(r', C) fails then fail;
(bD)   if backpropD_P(r', C) fails then fail;
       if jumpB_P(r', C) fails then fail;
       if jumpD_P(r', C) fails then fail;

procedure jumpB_P(r' :node; C :partial mapping,)
 var S : set of nodes;
   S := Succ1(r')
   while S ≠ ∅ do
     select r' from S;
     if C(r') = ⊖ then backpropB_P(r', C);
```

Figure 7: Extended propagation procedures including backward propagation and jumping.

A further improvement for the rule-based algorithm is so-called jumping. Backward propagation according to (bB), (bC) and (bD) requires certain conditions to be fulfilled, which may not be known when a node is colored. For (bA) this is not the case, because in (bA) there is no further condition. Take the following program P and its corresponding block graph Γ_P (see Figure 8):

$$P = \left\{ \begin{array}{ll} a \leftarrow not\ a, not\ b, not\ d & \\ b \leftarrow not\ c & d \leftarrow not\ e \\ c \leftarrow not\ b & e \leftarrow not\ d \end{array} \right\} \quad (2)$$

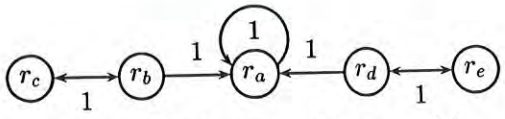


Figure 8: Block graph of program (2).

We know that $C(\{r_a\}) = \ominus$, otherwise there would not be an answer set at all. Since r_a is trivially grounded, it has to be blocked. This can be achieved by two rules, r_b and r_d, though one is sufficient. But we do not know yet, which one should be colored $\oplus$. Later on, when e.g. r_b has been used as a choice and is colored $\ominus$ this is achieved via jumping, that is, case (bB) is used again for node r_a and r_d is colored $\oplus$. Finally, with (A) node r_e is colored $\ominus$. In this way jumping helps to avoid unnecessary choices, because without jumping we would need another choice to color nodes r_d and r_e. Backward propagation would not be complete without jumping. In general, whenever a $\ominus$ is propagated along a 1-arc to an already $\ominus$-colored node, we check backward propagation case (bB) for this node again. Similarly, we check (bC) and (bD) again for $\oplus$-colored and $\ominus$-colored 0-successors of already colored and propagated nodes, respectively.

As an example, Figure 7 shows the implementation of procedure $\mathbf{jumpB}_P$, which jumps to an already colored node in order to check backward propagation (bB) again. By replacing procedure $\mathbf{propagate}_P$ in Figure 3 with $\mathbf{propagate}_P$ in Figure 7 we obtain an algorithm for computing a-colorings including backward propagation and jumping. The four procedures for the backward propagation cases are implemented similarly to procedure $\mathbf{propB}_P$ in Figure3, with the exception that we aim at coloring predecessors instead of successors now.

6 Results

For a transitory mapping $C : P \mapsto \{\ominus, \oplus, +\}$ we define the set of *corresponding answer sets* A_C as

$$A_C = \{X \mid X \text{ is answer set of } P \text{ and } C_\oplus \subseteq GR(P,X) \text{ and } C_\ominus \cap GR(P,X) = \emptyset\}.$$

If C is undefined for all nodes then A_C contains all answer sets of P. If C is a total mapping s.t. no node is colored with + then A_C contains exactly one answer set of P (if C is an a-coloring). With this notation we formulate the following result:

Theorem 4 *Let P be a logic program, let C and C' : $P \mapsto \{\ominus, \oplus, +\}$ be transitory mappings. Then for each $r \in (C_\ominus \cup C_\oplus \cup C_+)$ we have if $\mathbf{propagate}_P(\{r\}, C)$ succeeds and C' is the transitory mapping after its execution then $A_C = A_{C'}$.*

This theorem states that $\mathbf{propagate}_P$ neither discards nor introduces answer sets corresponding to some transitory mapping C. Hence, only nodes used as choices lead to different answer sets.

Finally, let C_P be the set of all solutions of $\mathbf{color}_P$. We obtain correctness and completeness of $\mathbf{color}_P$.

Theorem 5 *Let P be a logic program, let Γ_P be its block graph, let $\mathcal{C} : P \mapsto \{\ominus, \oplus\}$ be a mapping and let C_P the set of all solutions of* $\mathbf{color}_P$ *for program P. Then $\mathcal{C}$ is an a-coloring of Γ_P iff $\mathcal{C} \in C_P$.*

7 Experiments

As benchmarks, we have used some instances of NP-complete problems proposed in [2], namely, the independent set problem for circle graphs[8], the problem of finding Hamiltonian cycles in complete graphs and the problem of finding classical graph colorings. Furthermore we have tested some planning problems taken from [6] and the n-queens problem. In Table 1 we have counted the number of choices instead of measuring time, since the number of choices indicates how good an algorithm deals with a non-deterministic problem. For `smodels` results with and without lookahead (results in parenthesis) are shown[9].

	noMoRe			smodels	
backprop	no	yes	yes		
jumping	no	no	yes		
ham_k_7	14335	14335	2945	4814	(34077)
ham_k_8	82200	82200	24240	688595	(86364)
ind_cir_20	539	317	276	276	(276)
ind_cir_30	9266	5264	4609	4609	(4609)
p1_step4	-	464	176	7	(69)
p2_step6	-	13654	3779	75	(3700)
col4x4	27680	27680	7811	7811	(102226)
col5x5	-	-	580985	580985	(2.3 Mil)
queens4	84	84	5	1	(11)
queens5	326	326	13	9	(34)

Table 1: Number of choices (all solutions) for different problems.

The influence of backward propagation and jumping on the number of choices is clearly visible. There are also some problems where we did not obtain a solution after more than 12 hours without backward propagation. Table 1 impressively shows that `noMoRe` with backward propagation and jumping is now comparable with `smodels` on several problem classes; especially if we disable the lookahead of `smodels`. The difference between `smodels` and `noMoRe` for planning examples and the n-queens problems seems to come from different heuristics for making choices. We have just started to investigate the influence of more elaborated heuristics for making choices.

[8] A so-called circle graph Cir_n has n nodes $\{v_1, \cdots, v_n\}$ and arcs $A = \{(v_i, v_{i+1}) \mid 1 \leq i < n\} \cup \{(v_n, v_1)\}$.

[9] Since we feel that `dlv` does not give sufficient information on choices there are no results for `dlv` in Table 1. Whereas `smodels` and `noMoRe` make exactely one choice at each (choice) point in the search space, `dlv` makes several choices at the same point of the search space.

8 Conclusion

We have shown that by introducing backward propagation together with jumping the rule-based algorithm implemented in `noMoRe` can be greatly improved. A related method of backward propagation wrt answer set semantics for normal logic programs was proposed in [9]. However, a lot of the obtained improvement is due to the concept of a third color +. There seems to be a close relation between `noMoRe`'s color + and `dlv`'s must-be-true truth value [6], though this has to be studied more thoroughly, because `noMoRe` is rule-based and `dlv` (and `smodels`) is literal-based. Through the conducted experiments the impact of the improvements is shown. `NoMoRe` is now compareable to `smodels` on many different problem classes measuring the number of choices. This improvement was obtained by improving the deterministic consequences of `noMoRe`. However, there are still some interesting open questions. The main one is whether rule-based computation of answer sets is different from atom-based (literal-based) or not. During our experiments we have detected programs (with a large rule atom ratio) for which atom-based computations are more suitable and other programs (with a small rule atom ratio) for which rule-based computation performs better. Currently, we have no general answer to this question and a general comparison between atom-based and rules-based methods for logic programs will be necessary.

Another question for future work is whether rule-based approaches like implemented in `noMoRe` can lead to a system which is able to compete (considering time) against state of the art systems like `smodels` and `dlv`. Clearly, this question cannot be answered without having a system for the rule-based approach that is as elaborated implemented as `smodels` and `dlv` are. Right now for `noMoRe` this is not the case, because its development is in a state where `smodels` and `dlv` were some years ago, for example the heuristics of `noMoRe` still does not have a lookahead. In fact, the only very simple heuristic we use so far in `noMoRe` is to make choices for the color of a node r only if we have p-grounded(r). However, we think that our experiments show that there is a chance for rule-based methods to be able to compete against atom-based in the future. Therefore further work will also include the study of heuristics and different implementation techniques.

9 Acknowledgements

We would like to thank Ph. Besnard, K. Wang and T. Schaub for commenting on previous versions of this paper.

This work was partially supported by the German Science Foundation (DFG) within Project "Nichtmonotone Inferenzsysteme zur Verarbeitung konfligierender Regeln".

References

[1] C. Anger, K. Konczak, and T. Linke. NoMoRe: A system for non-monotonic reasoning with logic programs under answer set semantics. In R. Goré, A. Leitsch, and T. Nipkow, editors, *Automated Reasoning*, volume 2083 of *Lecture Notes in Computer Science*, pages 325–330. Springer, 2001. First International Joint Conference on Automated Reasoning.

[2] P. Cholewiński, V. Marek, A. Mikitiuk, and M. Truszczyński. Experimenting with nonmonotonic reasoning. In *Proceedings of the International Conference on Logic Programming*, pages 267–281. MIT Press, 1995.

[3] Y. Dimopoulos, B. Nebel, and J. Koehler. Encoding planning problems in non-monotonic logic programs. Proc. of the 4th European Conference on Planing, pages 169–181, Toulouse, France, 1997. Springer Verlag.

[4] D. East and M. Truszczyński. dcs: An implementation of datalog with constraints. In *Proceedings of the National Conference on Artificial Intelligence*. MIT Press, 2000.

[5] T. Eiter, N. Leone, C. Mateis, G. Pfeifer, and F. Scarcello. A deductive system for nonmonotonic reasoning. In J. Dix, U. Furbach, and A. Nerode, editors, *Proceedings of the Fourth International Conference on Logic Programming and Non-Monotonic Reasoning*, volume 1265 of *Lecture Notes in Artificial Intelligence*, pages 363–374. Springer Verlag, 1997.

[6] W. Faber, N. Leone, and G. Pfeifer. Pushing goal derivation in dlp computations. In M. Gelfond, N. Leone, and G. Pfeifer, editors, *Proceedings of the 5th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'99)*, volume 1730 of *Lecture Notes in Artificial Intelligence*, pages 177–191, El Paso, Texas, USA, 1999. Springer Verlag.

[7] W. Faber, N. Leone, and G. Pfeifer. Experimenting with heuristics for answer set programming. In B. Nebel, editor, *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 635–640. Morgan Kaufmann Publishers, 2001.

[8] M. Gelfond and V. Lifschitz. Classical negation in logic programs and deductive databases. *New Generation Computing*, 9:365–385, 1991.

[9] N. Iwayama and K. Satoh. Computing abduction by using tms with top-down expectation. *Journal of Logic Programming*, 44:179–206, 2000.

[10] V. Lifschitz. Answer set planning. In *Proceedings of the 1999 International Conference on Logic Programming*, pages 23–37. MIT Press, 1999.

[11] T. Linke. Graph theoretical characterization and computation of answer sets. In B. Nebel, editor, *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 641–645. Morgan Kaufmann Publishers, 2001.

[12] T. Linke. Rule-based computation of answer sets. 2002. submitted.

[13] X. Liu, C. Ramakrishnan, and S.A. Smolka. Fully local and efficient evaluation of alternating fixed points. Proc. of the 4th Int. Conf. on Tools and Algorithms for the Construction Analysis of Systems, pages 5–19, Lisbon, Portugal, 1998. Springer Verlag.

[14] I. Niemelä. Logic programming with stable model semantics as a constraint programming paradigm. *Annals of Mathematics and Artificial Intelligence*, 25(3,4):241–273, 1999.

[15] I. Niemelä and P. Simons. Smodels: An implementation of the stable model and well-founded semantics for normal logic programs. In J. Dix, U. Furbach, and A. Nerode, editors, *Proc. of the Fourth International Conference on Logic Programming and Nonmonotonic Reasoning*, pages 420–429. Springer, 1997.

On Logic Programs with Cardinality Constraints

Victor W. Marek
Department of Computer Science
University of Kentucky
Lexington, KY 40506, USA
marek@cs.uky.edu

Jeffrey B. Remmel
Department of Mathematics
University of California
La Jolla, CA 92093, USA
jremmel@ucsd.edu

Abstract

We investigate cardinality-constraint (CC) logic programs as implemented in the ASP solver *smodels*. Niemelä, Simons and collaborators in [NSS99, NS00, Syr01] defined a notion of stable model, which we call CC-stable, for CC-logic programs that differs from the usual notion of stable model of logic programs in a variety of ways. For example, it is not always the case that the set of CC-stable models of a CC-program form an anti-chain. The main result of this paper is to show that there is a natural transformation of a CC-logic program P into a standard logic program Q in an extended language such that the CC-stable models of P are the projections of the stable models of Q to the original language. Niemelä, Simons and Soinenen [NSS99] proved that the existence problem for CC-stable models of CC-logic programs is NP-complete. We show that the existence problem for CC-stable models for several restricted classes of CC-logic programs is also NP-complete. For example, the existence problem for CC-stable models for CC-logic programs where each clause has an empty body is already NP-complete.

1 Introduction

The current progress in computational knowledge representation and, in particular, the success of the current Answer Set Programming (ASP) engines [SK92, KS99, NS00, MT99, ASP01] makes it important to understand the semantics of underlying logical formalisms. In this extended abstract, we investigate cardinality-constraint logic programs (CC-logic programs) introduced by Niemelä, Simons and their collaborators in [NSS99, NS00, Syr01]. In particular, we study the notion of stable models for CC-logic programs as introduced by Niemelä, Simons and Soinene [NSS99], which we will call CC-stable models, and relate it to the standard notion of stable models for logic programs as introduced by Gelfond and Lifschitz [GL88].

CC-logic programs form a subclass of a larger class of programs called weight-constraint logic programs. The CC-stable semantics for this wider class of programs have been already implemented in the current version of stable semantics solver *smodels*. Niemelä and Simons [NS00] give numerous applications of CC-logic programs to various combinatorial optimization problems which show that CC-logic programs are natural and useful. Thus, since the CC-logic programs are accepted by a widely used solver, it is important to understand the CC-stable semantics of such programs. The goal of this paper is to show how the CC-stable semantics of a CC-logic program P can be viewed as the projection of the stable logic semantics of a logic program Q in an extended language. This will allow the programmer to understand the semantics of CC-logic programs as implemented by *smodels* as a natural generalization of the semantics of programs that were available in earlier versions.

A detailed explication of the notion of CC-stable models for CC-logic programs is given in [NSS99, NS00]. In those papers, the authors show that the CC-stable model semantics coincides with the usual stable model semantics of logic programs for those CC-logic programs with no cardinality constraints. However, the CC-stable model semantics for CC-logic programs differs from the usual stable model semantics for logic programs in two important ways. First, CC-stable models of CC-logic programs can be included one in another while a salient feature of stable models of ordinary logic programs is that they are minimal and, hence, they do not nest. Second, the definition of a

CC-stable model M of a CC-logic program P requires that M must first be a model of P, while in case of ordinary logic programs Q, the fact that a stable model is a model of Q comes for free. We will see that, with an appropriate point of view, both problems turn out to be illusory. We will show that there is a syntactic transformation $t(\cdot)$ which assigns to every CC-logic program P, an ordinary logic program in an extended language (i.e. one that admits more atoms) which preserves stable models in the following sense. Specifically, the restriction of the stable models of $t(P)$ to the language of P is a one-to-one projection. That is, for every stable model M of $t(P)$, the intersection M with the Herbrand universe of P is a CC-stable model of P. Conversely, every CC-stable model of P uniquely extends to a stable model of $t(P)$. Thus our mapping $t(\cdot)$ is a meta-interpretation of CC-logic programs with CC-stable model semantics in logic programs with the usual stable model semantics of Gelfond and Lifschitz.

Another way to view this result is that CC-stable models of a CC-logic program P is the result of implementing a **hide** operation, see [Syr01], p. 46, for the program $t(P)$. The operation **hide** is a useful construct that can improve the readability of an *smodels* output. For example, when **hide**(p) is declared, the atoms of the form $p(\bar{a})$ that belong to the computed stable model are not displayed. In our case, the auxiliary atoms that occur in $t(P)$ but not in P, together with denotations for cardinality constraints that occur explicitly in P, are not shown. Thus conceptually, one could view the operation of *smodels* on a CC-logic program as a solver which processes the larger program $t(P)$, but only outputs those atoms which explicitly occur in P. Thus one can think of the use of **hide** as eliminating the atoms that are employed in the search for a stable model but which are not important from the point of view of the problem solved by the program. It is easy to see that the **hide** operation can result in projected stable models which are included in each other. For example, consider the following program P, which relative to the usual stable semantics, can be used to generate all subsets of a given set A.

Example 1.1 The language of P has two predicates: $in(\cdot)$ and $out(\cdot)$. Then P contains the following pair of clauses for each $a \in A$.

$$in(a) \leftarrow \neg out(a)$$
$$out(a) \leftarrow \neg in(a)$$

It is easy to see that for any subset $S \subseteq A$,

$$\{in(a) : a \in S\} \cup \{out(a) : a \in A \setminus S\}$$

constitutes a stable model for P. It is also easy to see that when one hides the predicate $out(\cdot)$, the parts of stable models that are returned by the solver can nest.

For example, let $A = \{x, y, z\}$. Then $M_1 = \{in(x), out(y), out(z)\}$ and $M_2 = \{in(x), in(y), out(z)\}$ are two stable models of P. M_1 and M_2 do not nest, of course, but, if we declare **hide**(out) in *smodels*, then *smodels* will display: $M_1 = \{in(x)\}$ and $M_2 = \{in(x), in(y)\}$. □

We view our results as a confirmation of the correctness of the approach of the authors of [NSS99]. While there may exist other semantics of CC-logic programs that behave similarly to the stable semantics when limited to ordinary logic programs, our meta-interpretation result, as well our characterization of proof schemes for CC-logic programs to follow, seem to indicate that the semantics implemented by *smodels* is the correct one.

An alternative approach to relating the CC-stable model semantics of CC-logic program to a more standard semantics is offered by Ferraris and Lifschitz in [FL01]. While we focus on CC-logic programs, Ferraris and Lifschitz look at the entire class of weight-constraint programs (WC-logic programs). Ferraris and Lifschitz translate WC-programs into *nested expressions* theories, that is, theories in the formalism of *nested expressions* of Lifschitz, Tang and Turner [LTT99]. The translation provided in [FL01] has the advantage that the set of CC-stable models of a WC-program coincides with the set of answer sets of nested expression image of the translated program. The drawback of the approach of [FL01] from our point of view is that it does not explain the CC-stable model semantics of CC-logic programs in terms of the simpler stable model semantics of logic programs. Instead, the translation of [FL01] explains the extension of the stable model semantics for the richer class of CC-logic programs in terms of the extension of the stable model semantics for an alternative extension of logic programs.

This paper is structured as follows. In section 2, we review the original definition of CC-stable model semantics for CC-logic programs [NSS99]. Then we will show how the basic proof-theoretic characterization of stable models in terms of proof schemes [MNR90] can be lifted to the case of CC-logic programs. In Section 3, we introduce our translation of CC-logic programs into ordinary logic programs. In Section 4, we prove some simple complexity results for CC-logic programs. In particular, we shall show that the basic NP-completeness result for the existence of CC-stable models of [NSS99] already holds for a very restricted class of CC-logic programs. Specifically, while

Niemela, Simons, and Soinenen show that the existence problem for stable models of CC-logic programs is NP-complete, we show that this result holds for CC-programs where every clause has an empty body. We also give a reduction of propositional satisfiability to the existence of CC-stable models for CC-logic programs that have no constraints in the bodies of clauses. This leads to yet another strengthening of basic complexity results of [NS00]. Finally, in Section 5, we indicate some other directions for future research on CC-logic programs and their extensions.

2 Logic programs, CC-logic programs and their stable semantics

Recall that a clause C of for a logic program P is a rule of the form

$$p \leftarrow q_1, \ldots, q_m, \neg r_1, \ldots, \neg r_n. \tag{1}$$

where $p, q_1, \ldots, q_m, r_1, \ldots, r_n$ are atoms from the set of atoms At of the program. We shall refer to p as the head of C, $head(C)$, $\{q_1, \ldots, q_m\}$ as the premises of C, $prem(C)$, $\{r_1, \ldots, r_n\}$ as the constraints of C, $cons(C)$, and $q_1 \wedge \ldots \wedge q_m \wedge \neg r_1 \wedge \ldots \wedge \neg r_n$ as the body of p, $body(C)$. A logic program is a set P of clauses. To distinguish them from other clauses described below for CC-logic programs, we shall refer to clause of the form of (1) as an *ordinary* logic program clause. When $n = 0$, the clause C is called a *Horn clause*. A Horn program is a set of Horn clauses. A Horn program always has a least model which we denote that model by M_P.

The operator $GL(\cdot, \cdot)$ [GL88] assigns to a logic program P and a set of atoms M, the least model of the Horn program P^M where P^M consists of the set of all Horn clauses C^M obtained from a clause C of P of the form of (1) as follows.

$$C^M = \begin{cases} \textbf{nil} & \text{if for some } i, 1 \leq i \leq n, \\ & r_i \in M \\ p \leftarrow q_1, \ldots, q_m & \text{otherwise} \end{cases} \tag{2}$$

We say that M is a stable model of P if $M = GL(P, M)$. Gelfond and Lifschitz proved that every stable model M of P is a model of P, in fact, a minimal and supported model of P.

Next we define a natural proof-theoretic construct associated to logic programs called *proof schemes*. Let P be a logic program, then the set of proof schemes of P can be defined inductively as follows.

1. If $C = p \leftarrow \neg r_1, \ldots, \neg r_n$ is a clause of P (the case $n = 0$ is allowed), then $\langle \langle p \rangle, \langle C \rangle, \{r_1, \ldots, r_n\} \rangle$ is a proof scheme for p in P
2. If $\langle \langle s_1, \ldots, s_k \rangle, \langle C_1, \ldots, C_k \rangle, \{t_1, \ldots, t_l\} \rangle$ is a proof scheme and $C = p \leftarrow q_1, \ldots, q_m, \neg r_1, \ldots, \neg r_n$ is a clause in P and $\{q_1, \ldots, q_m\} \subseteq \{s_1, \ldots, s_k\}$, then

$$\langle \langle s_1, \ldots, s_k, p \rangle, \langle C_1, \ldots, C_k, C \rangle, \{t_1, \ldots, t_l, r_1, \ldots, r_n\} \rangle$$

is a proof scheme for p in P.

If $\mathfrak{S} = \langle \langle s_1, \ldots, s_k \rangle, \langle C_1, \ldots, C_k \rangle, \{t_1, \ldots, t_l\} \rangle$ is a proof scheme in P, we refer to s_k as the conclusion of $\mathfrak{S}$, $concl(\mathfrak{S})$, and say that $\mathfrak{S}$ is a proof scheme for s_k in P. We also refer to $\{t_1, \ldots, t_k\}$ as the constraints of of $\mathfrak{S}$, $const(\mathfrak{S})$.

One can think of a proof scheme for a logic program as the analogue of a derivation or proof in classical logic. However, a proof scheme $\mathfrak{S}$ for p not only contains the clauses that can be used to derive p but also keeps track of the set of atoms that must be absent from prospective stable model M, namely $const(\mathfrak{S})$, if p is to be an element of $GL(P, M)$. Thus we say that a proof scheme $\mathfrak{S}$ is *admitted* by M if $M \cap const(\mathfrak{S}) = \emptyset$.

Example 2.1 Let P consist of clauses:

$$C_1 = p \leftarrow q, \neg r$$
$$C_2 = q \leftarrow \neg s$$
$$C_3 = s \leftarrow \neg q.$$

The triple

$$\mathfrak{S}_1 = \langle \langle q, p \rangle, \langle C_2, C_1 \rangle, \{r, s\} \rangle$$

is a proof scheme for p in P. Similarly, $\mathfrak{S}_2 = \langle \langle q \rangle, \langle C_2 \rangle, \{s\} \rangle$ is a proof scheme for q and $\mathfrak{S}_3 = \langle \langle s \rangle, \langle C_3 \rangle, \{q\} \rangle$ is a proof scheme for s. The set $\{p, q\}$ is a stable model of P. It admits $\mathfrak{S}_1$ and $\mathfrak{S}_2$, but not $\mathfrak{S}_3$. □

The following result is proven in [MNR90].

Proposition 2.1 *Let M be a set of atoms contained the Herbrand base At_P of the a logic program P. Then M is a stable model of P if and only if*

1. *Every atom p of M possesses a proof scheme $\mathfrak{S}_p$ in P such that M admits $\mathfrak{S}_p$*
2. *No atom p in $At \setminus M$ possesses a proof scheme admitted by M.*

Proposition 2.1 implies the following property of models of programs.

Corollary 2.2 *Let P be a logic program and let M be a model of P. Then M is a stable model of P if and only if every element of M possesses a proof scheme in P admitted by M.*

The advantage of proof schemes is that they are entities associated with programs and atoms and not with models. Proof schemes carry within themselves the information about their own applicability. Notice that Corollary 2.2 establishes a condition for models of P that is easier to check than the conditions given in Proposition 2.1. Below we will extend the notion of proof scheme to CC-logic programs and prove a result analogous to Corollary 2.2. This is one reason why we believe that the definition of CC-stable models of cardinality-constraint programs is a natural generalization of stable semantics for ordinary logic programs.

We now formally define cardinality-constraint logic programs (CC-logic programs). The syntax of CC-logic programs admits two types of atoms: (i) ordinary atoms from set At and (ii) atoms of the form kXl where X is a finite set of atoms from At, k is a natural number (i.e. $k \in \omega$), $l \in \omega \cup \{\infty\}$ and $k \leq l$. Such new atoms will be called *cardinality constraints*. The intended meaning of an atom kXl is "out of atoms in X at least k but not more than l belong to the intended model."[1] Notice that the meaning of the negated atom, $\neg p$ is precisely the same as that of $\{p\}0$. Therefore we shall assume that the bodies of rules of CC-logic programs contain only atoms of the form kXl and atoms from At. That is, a CC-clause is either a clause of the form

$$p \leftarrow q_1, \ldots, q_m, k_1X_1l_1, \ldots, k_nX_nl_n \qquad (3)$$

or

$$kXl \leftarrow q_1, \ldots, q_m, k_1X_1l_1, \ldots, k_nX_nl_n. \qquad (4)$$

We note that either m or n can be zero. Thus the head of CC-clauses is either of the form p where p is an atom from At or kXl where k, X, and l satisfy the conventions described above. We say that a set of atoms $M \subseteq At$ satisfies the cardinality constraint kXl, $M \models kXl$ if $k \leq |X \cap M| \leq l$. Similarly we say that $M \models p$ where $p \in At$, if $p \in M$. By treating the commas in the bodies of clauses as conjunctions, we say that $M \models body(C)$ if all atoms occurring in $body(C)$ belong to M and all cardinality constraints occurring in $body(C)$ are satisfied by M. Finally, we say that M satisfies a clause C, $M \models C$, if either M does not satisfy the body of C or M satisfies the head of C.

A CC-logic program is a set of CC-clauses of the form (3) or (4). We say that M is model of P, $M \models P$, if M satisfies all CC-clauses $C \in P$.

There is a particular class of programs called Horn constraint programs that play a role similar to that of Horn programs in ordinary logic programming. A *Horn constraint clause* is a CC-clause where the head of the clause is an ordinary atom and all the cardinality-constraint atoms $k_iX_il_i$ in the body have $l_i = \infty$, i.e., it is of the form

$$H = p \leftarrow q_1, \ldots q_m, k_1X_1, \ldots, k_nX_n.$$

Niemelä, Simons and Soininen observe that the one step provability operator associated with a such Horn constraint program is monotone and hence a Horn constraint program P has a least fixed point, M^P. Moreover, they show that M^P is the least model of P.

Next we introduce the analogue of the Gelfond-Lifschitz reduct for CC-logic clauses which we call the NSS-reduct. The NSS-reduct of a cardinality-constraint clause C with respect to a set M of ordinary atoms is defined as follows.

1. if $C = p \leftarrow q_1, \ldots, q_m, k_1X_1l_1, \ldots, k_nX_nl_n$, then $C^M = p \leftarrow q_1, \ldots, q_m, k_1X_1, \ldots, k_nX_n$
2. If $C = kXl \leftarrow q_1, \ldots, q_m, k_1X_1l_1, \ldots, k_nX_nl_n$, then C^M is a collection of Horn constraint clauses of the form
   $p \leftarrow q_1, \ldots, q_m, k_1X_1, \ldots, k_nX_n$
   for each $p \in X \cap M$.

Given a CC-program P, we let P^M denote a Horn constraint program consisting of all NNS-reducts of clauses $C \in P$. Following [NSS99], we say that M is a *CC-stable model* of P if (i) M is a model of P and (ii) M is the least model of the Horn constraint program P^M.

Next we define the notion of a proof scheme for a CC-logic program and state a result analogous to Corollary 2.2. The basic idea is that a proof scheme should carry along all the information that is needed to see that an element p is in a CC-stable model M. In particular, when we deal with atoms of the form kXl, we need to know the information that $k \leq |M \cap X| \leq l$. Thus our proof schemes will carry along the information about what we expect $M \cap X$ to be. Formally, the notion of CC-proof scheme for a CC-logic program P is defined inductively as follows.

[1] Customarily we do not write the lower bound if it is 0 nor the upper bound, if it is ∞.

1. Whenever $C = p \leftarrow l'_1 X_1 l''_1, \ldots, l'_n X_n l''_n$ is a clause in P and for all $1 \leq i \leq n$, Y_i is a subset of X_i such that $l'_i \leq |Y_i| \leq l''_i$ then

$$\langle\langle p\rangle, \langle C\rangle, \langle (l'_1 X_1 l''_1, Y_1), \ldots, (l'_n X_n l''_n, Y_n)\rangle\rangle$$

is a CC-proof scheme for P. (The case $n = 0$ is allowed.)

2. Whenever $l' X l'' \leftarrow l'_1 X_1 l''_1, \ldots, l'_n X_n l''_n$ is a clause in P and for all $1 \leq i \leq n$, Y_i is a subset of X_i such that $l'_i \leq |Y_i| \leq l''_i$ and Y is a subset of X such that $k \leq |Y| \leq l$, then for every $p \in Y$

$$\langle\langle p\rangle, \langle C\rangle, \langle (l' X l'', Y), (l'_1 X_1 l''_1, Y_1), \ldots, (l'_n X_n l''_n, Y_n)\rangle\rangle$$

is a CC-proof scheme for P. (Again, the case $n = 0$ is allowed.)

3. Whenever

$$\mathfrak{S} = \langle\langle s_1, \ldots, s_w\rangle, \langle C_1, \ldots, C_w\rangle, \langle (k'_1 X_1 k''_1, Y_1), \ldots, (k'_r X_r k''_r, Y_k)\rangle\rangle$$

is a CC-proof scheme in P and

$$C = p \rightarrow q_1, \ldots, q_m, l'_1 Z_1 l''_1, \ldots, l'_n Z_n l''_n$$

is a clause in P such that $\{q_1, \ldots, q_m\} \subseteq \{s_1, \ldots, s_w\}$ and for all $1 \leq i \leq n$, T_i is a subset of Z_i such that $l'_i \leq |T_i| \leq l''_i$, then

$$\langle\langle s_1, \ldots, s_w, p\rangle, \langle C_1, \ldots, C_w, C\rangle, \langle (k'_1 X_1 k''_1, Y_1), \ldots, (k'_r X_r k''_r, Y_r), (l'_1 Z_1 l''_1, T_1) \ldots, (l'_n Z_n l''_n, T_n)\rangle\rangle$$

is a CC-proof scheme for P.

4. Whenever

$$\mathfrak{S} = \langle\langle s_1, \ldots, s_w\rangle, \langle C_1, \ldots, C_w\rangle, \langle (k'_1 X_1 k''_1, Y_1), \ldots, (k'_r X_r k''_r, Y_k)\rangle\rangle$$

is a proof scheme in P and

$$C = l' Z l'' \rightarrow q_1, \ldots, q_m, l'_1 Z_1 l''_1, \ldots, l'_n Z_n l''_n$$

is a clause in P such that $\{q_1, \ldots, q_m\} \subseteq \{s_1, \ldots, s_w\}$, for all $1 \leq i \leq n$, T_i is a subset of Z_i such that $l'_i \leq |T_i| \leq l''_i$ and Y is a subset of Z such that $l' \leq |Y| \leq l''$, then for every $p \in Y$

$$\langle\langle s_1, \ldots, s_w, p\rangle, \langle C_1, \ldots, C_w, C\rangle, \langle (l' Z l'', Y), (k'_1 X_1 k''_1, Y_1), \ldots, (k'_r X_r k''_r, Y_r), (l'_1 Z_1 l''_1, T_1) \ldots, (l'_n Z_n l''_n, T_n)\rangle\rangle$$

is a proof scheme for P.

Now, given a CC-proof scheme

$$\mathfrak{S} = \langle\langle s_1, \ldots, s_k\rangle, \langle C_1, \ldots, C_k\rangle, \langle (k'_1 X_1 k''_1, Y_1), \ldots, (k'_r X_r k''_r, Y_r)\rangle\rangle$$

for P, we say that $\mathfrak{S}$ is a CC-proof scheme for s_k in P. We shall refer to the sequence $\langle (k'_1 X_1 k''_1, Y_1), \ldots, (k'_r X_r k''_r, Y_r)\rangle$ as the *cardinality constraint sequence* of $\mathfrak{S}$. If $M \subseteq At$, then we say that $\mathfrak{S}$ is *admitted* by M if $M \cap X_1 = Y_1, \ldots, M \cap X_k = Y_k$. It is easy to see that if M admits a proof scheme

$$\mathfrak{S} = \langle\langle s_1, \ldots, s_k\rangle, \langle C_1, \ldots, C_k\rangle, \langle (k'_1 X_1 k''_1, Y_1), \ldots, (k'_r X_r k''_r, Y_r)\rangle\rangle$$

then $M \models k'_1 X_1 k''_1, \ldots, M \models k'_r X_r k''_r$ since the sets Y_i, $i = 1, \ldots, k$ witnesses that the corresponding constraints are satisfied.

Thus a proof scheme provides a derivation of an atom and proposes a way of satisfying constraints occurring in bodies of all clauses used in that derivation. Moreover, the proof schemes for ordinary programs can be easily transformed into the CC-proof schemes for the corresponding cardinality-constraint program. That is, instead of having an element r be in set of constraints in the third component of a proof scheme for an ordinary logic program, we simply add a pair $(\{r\}0, \emptyset)$ to the cardinality constraint sequence of the corresponding CC-proof scheme because a set $M \subseteq At$ will admit such a proof scheme if and only if $r \notin M$.

Example 2.2 Let P be the following CC-logic program:

$$C_1 = 1\{p, q\}2 \leftarrow r, \{t\}0$$
$$C_2 = r \leftarrow \{s\}0$$
$$C_3 = s \leftarrow \{r\}0$$

The CC-program P has four stable models: $M_1 = \{r, p\}$, $M_2 = \{r, q\}$, $M_3 = \{r, p, q\}$ and $M_4 = \{s\}$. M_1 and M_2 are included in M_3.

The triple $\langle\langle s\rangle, \langle C_3\rangle, \langle\{r\}0, \emptyset\rangle\rangle$ is a proof scheme for s in P. This proof scheme is not admitted by M_1, M_2 or M_3. It is admitted by M_4. The triple $\langle\langle r, p\rangle, \langle C_2, C_1\rangle, \langle(1\{p, q\}2, \{p\}), (\{s\}0, \emptyset), (\{t\}0, \emptyset)\rangle\rangle$ is admitted by M_1, but not by M_2. Also, the scheme $\mathfrak{S}_3$

$$\langle\langle r, p, q\rangle, \langle C_2, C_1, C_1\rangle, \langle(1\{p, q\}2, \{p, q\}), (\{s\}0, \emptyset), (\{t\}0, \emptyset)\rangle\rangle$$

is admitted by M_3 but not by M_1, because atom q does not belong to M_1. Notice that clause C_1 is used in $\mathfrak{S}_3$ twice, once to derive p and again to derive q. □

This given, we can prove the following analogue of Corollary 2.2.

Proposition 2.3 *Let P be a CC-logic program, and let $M \subseteq At$, $M \models P$. Then M is a stable model of P if and only if every element p of M possesses a proof scheme $\mathfrak{S}_p$ such that $\mathfrak{S}_p$ is admitted by M.*

As Proposition 2.3 is a natural generalization of Corollary 2.2, it provides an evidence that the stable models of [NSS99] correctly generalize the ordinary stable models.

3 Reduction of CC-logic programs to ordinary programs

In this section we construct a transformation of CC-logic programs to ordinary logic programs with the desired property of "lifting" CC-stable models of a CC-logic program to stable models of its image. Specifically, for every CC-logic program P we define an ordinary logic program $t(P)$ with the following properties:

1. For every stable model N of $t(P)$, the set $N \cap At$ is a CC-stable model of P
2. Every CC-stable model M of P is of the form $N \cap At$ for a unique stable model N of $t(P)$.

Thus the correspondence between the CC-stable models M of P and the stable models N of $t(P)$ will be one-to-one.

Let P be a CC-logic program and let At be the set of all ordinary atoms occurring in P. Let $\mathcal{X}$ be the set of all cardinality constraints, that is, the expressions of the form kXl, occurring in P. Here $k \in \omega$, X is a finite subset of At, $l \in \omega \cup \{\infty\}$, and $k \leq l$.

Next we describe the language of $t(P)$. For each such cardinality constraint kXl, we add two new atoms $Q^b_{k,l,X}$ and $Q^h_{k,l,X}$to the language of $t(P)$. Here the superscript b stands for *body* and the superscript h stands for *head*. We need two atoms for kXl because cardinality constraint atoms play different roles in CC-programs depending on whether they are in the body or the head. For each $kXl \in \mathcal{X}$ and every atom $a \in At$, we add two new atoms, $P_{k,l,X,a}$ and $\bar{P}_{k,l,X,a}$, to the language of $t(P)$. Finally we add one additional new atom, f, which will help us write various constraint clauses. The idea is that f will always be used in the context $f \leftarrow \mathit{condition}, \neg f$ where *condition* is set of atoms or constraint conditions. Moreover, these will be the only type of clauses that have f in the head. This means that f cannot be part of any stable model M of $t(P)$ since the only way to derive f from $t(P)^M$ would be to have $f \notin M$. Thus clauses of the form $f \leftarrow \mathit{condition}, \neg f$ state that *condition* cannot hold in a stable model[2].

Next we describe the program $t(P)$. It will consist of several groups of rules. In addition to a suitably modified copy of the program P itself, it will have additional rules used to enforce the correct interpretation of the new atoms that we added to the language of $t(P)$. The program $t(P)$ consists of seven groups of rules.

(I) (Translation) Substitute in P, $Q^b_{k,l,X}$ for every occurrence of the cardinality constraint kXl in the body of a clause and $Q^h_{k,l,X}$ for every occurrence of kXl in the head of a clause. The result of this translation is an ordinary, in fact a Horn, program.

(II) (Constraining size of each X) For each atom $Q^b_{k,l,X}$ such that $l \neq \infty$, for each pair of disjoint sets $\{a_1, \ldots, a_k\}$ and $\{b_1, \ldots, b_{|X|-l}\}$ contained in X, add the following clause to $t(P)$.

$$Q^b_{k,l,X} \leftarrow a_1, \ldots, a_k, \neg b_1, \ldots, \neg b_{|X|-l}$$

For each atom $Q^b_{k,l,X}$ such that $l = \infty$ and for each $\{a_1, \ldots, a_k\} \subseteq X$, add the following clause to $t(P)$.

$$Q^b_{k,l,X} \leftarrow a_1, \ldots, a_k$$

(III) (Constraining $Q^c_{k,l,X}$ to behave properly)

(III1) For each atom $Q^c_{k,l,X}$ with $c \in \{b, h\}$ and $l \in \omega$ and every set of $l+1$ elements $a_1, \ldots, a_{l+1}$ in X add the following clause to $t(P)$.

$$f \leftarrow P_{k,l,X,a_1}, \ldots, P_{k,l,X,a_{l+1}}, Q^c_{k,l,X}, \neg f$$

(III2) For each atom $Q^c_{k,l,X}$ where $c \in \{b, h\}$ and every set of $|X|-k+1$ elements $b_1, \ldots, b_{|X|-l+1}$ in X, add the following clause to $t(P)$.

$$f \leftarrow \bar{P}_{k,l,X,b_1}, \ldots, \bar{P}_{k,l,X,b_{|X|-l+1}}, Q^c_{k,l,X}, \neg f$$

(IV) (Choice) For each $kXl \in \mathcal{X}$ and $a \in At$, add the following clauses to $t(P)$ for both $c = b$ and $c = h$.

(IV1) $P_{k,l,X,a} \leftarrow \neg \bar{P}_{k,l,X,a}, Q^c_{k,l,X}$

(IV2) $\bar{P}_{k,l,X,a} \leftarrow \neg P_{k,l,X,a}, Q^c_{k,l,X}$

(IV3) $f \leftarrow \bar{P}_{k,l,X,a}, P_{k,l,X,a}, Q^c_{k,l,X}, \neg f$

[2] Of course, when writing *smodels* programs we do not need to use the special symbol f - instead of $f \leftarrow \mathit{condition}, \neg f$ we would simply write : $-\mathit{condition}$.

(V) (No extraneous atoms in X) For $kXl \in \mathcal{X}$ and $a \in At \setminus X$, add the following clause to $t(P)$ for both $c = b$ and $c = h$.

$$f \leftarrow P_{k,l,X,a}, Q^c_{k,l,X}, \neg f$$

(VI) (Alternative way of deriving ordinary atoms) For each $kXl \in \mathcal{X}$ and $a \in At$, add the following clause to $t(P)$.

$$a \leftarrow P_{k,l,X,a}, Q^h_{k,l,X}$$

(VII) (Limiting choice of additional atoms via ordinary atoms) For each $kXl \in \mathcal{X}$ and $a \in At$, add the following clauses to $t(P)$ for both $c = b$ and $c = h$.

(VII1) $P_{k,l,X,a} \leftarrow a, Q^c_{k,l,X}$

(VII2) $\bar{P}_{k,l,X,a} \leftarrow \neg a, Q^c_{k,l,X}$

The intention of the new atoms in the program $t(P)$ is as follows. First the atom $Q^b_{k,l,X}$ replaces the cardinality constraint kXl in bodies of clauses and $Q^h_{k,l,X}$ replaces kXl in the head of clauses. If M is a stable model of $t(P)$, $c \in \{b,h\}$ and $Q^c_{k,l,X} \in M$, then our intention is that $P_{k,l,X,a}$ holds if $a \in M$ and $\bar{P}_{k,l,X}$ holds if $a \notin M$.

Observe that an atom $Q^h_{k,l,X}$ can be derived via a translation of a clause in P. Similarly an atom $Q^b_{k,l,X}$ can be derived via an application of a clause in (II) which can happen only if $k \leq |M \cap X| \leq l$. This reflects exactly how kXl can participate in CC-proof schemes in the original CC-logic program P.

Atoms of the form $P_{k,l,X,a}$ and $\bar{P}_{k,l,X,a}$ can also be derived relative to $t(P)^M$ in two ways assuming that $Q^c_{k,l,X} \in M$ where $c \in \{b,h\}$. That is, the clauses of type (IV) say that we can derive $P_{k,l,X,a}$ from the absence of $\bar{P}_{k,l,X,a}$ and we can derive $\bar{P}_{k,l,X,a}$ from the absence of $P_{k,l,X,a}$. However the clauses of the type (IV3) ensure that it can not be the case that both $\bar{P}_{k,l,X,a}$ and $P_{k,l,X,a}$ hold. Then clauses of type (VII) ensure that $P_{k,l,X,a}$ holds if $a \in M$ and $\bar{P}_{k,l,X,a}$ if $a \notin M$ and the clauses of type (V) ensure that $P_{k,l,X,a}$ holds only if $a \in X$.

More generally, we can summarize the role the various sets of clauses as follows. The clauses of type (I) are just translations of clauses of P except that they treat cardinality constraints as new atoms. The remaining clauses (groups II–VII) make sure that the semantics assigned to cardinality constraints $Q^c_{k,l,X}$ is correct. The clauses of type (II) ensure that for any stable model M of $t(P)$, if $k \leq |M \cap X| \leq l$, then $Q^b_{k,l,X}$ can be derived. The clauses of type (IV) and (V) ensure that $\{a : P_{k,l,X,a} \in M\}$ is a subset of X. The clauses of type (III) then ensure that if $Q^c_{k,l,X} \in M$ for $c \in \{b,h\}$, then $k \leq |\{a : P_{k,l,X,a} \in M\}| \leq l$. These clauses combined with the clauses of type (VI) and (VII), then ensure that if $Q^c_{k,l,X} \in M$, then $k \leq |M \cap X| \leq l$.

This given, we then have the following theorem.

Theorem 3.1 *Let P be a CC-logic program, At denote the set of atoms that occur in P and $t(P)$ be the ordinary logic program consisting of clauses (I)-(VII). Then the mapping $N \mapsto M = N \cap At$ is a one-to-one projection from the set of stable models of $t(P)$ to the set of CC-stable models of P.*

We will briefly indicate how Theorem 3.1 is proven.

First we must show that given a CC-stable model M of P, there is a unique stable model N_M of $t(P)$ such that $N_M \cap At = M$. Given our discussion above, it is clear what N_M has to be. That is, let $\mathcal{X}^b_M = \{kXL \in \mathcal{X} : k \leq |M \cap X| \leq l\}$. Similarly let $\mathcal{X}^h_M$ denote the set of all $kXl \in \mathcal{X}$ such that there is a CC-proof scheme

$$\mathfrak{S} = \langle\langle s_1, \ldots, s_w\rangle, \langle C_1, \ldots, C_k\rangle, \langle (k'_1 X_1 k''_1, Y_1), \ldots, (k'_r X_r k''_r, Y_r), \rangle\rangle$$

admitted by M where the clause C_k has its head equal to kXL. Then

$$\begin{aligned} N_M \quad = \quad & M \cup \{Q^b_{k,l,X} : kXl \in \mathcal{X}^b_M\} \cup \\ & \{Q^h_{k,l,X} : kXl \in \mathcal{X}^b_M\} \cup \\ & \{P_{k,l,X,a} : kXl \in \mathcal{X}^b_M \ \&\ a \in M\} \cup \\ & \{\bar{P}_{k,l,X,a} : kXl \in \mathcal{X}^b_M \ \&\ a \notin M\}. \end{aligned}$$

The key idea of the proof of the fact that N_M is a stable model of $t(P)$ is to show that every CC-proof scheme admitted by M can be expanded to a proof scheme admitted by N_M. This argument relies on the fact that M is a model of P. Then it is easy to see that the clauses in (VII) and (III) will allow us to easily derive the rest of the atoms in N_M.

Conversely, suppose that we are given stable model N of $t(P)$. Then let $M = N \cap At$. First it is easy to see from the clauses of type (II) that $Q^b_{k,l,X} \in M$ if $k \leq |M \cap X| \leq l$. Next the clauses of type (III), (IV), (V), and (VIII) ensure that for $c \in \{b,h\}$, $Q^c_{k,l,X} \in N$ only if $k \leq |M \cap X| \leq l$. Thus $Q^b_{k,l,X} \in M$ if and only if $k \leq |M \cap X| \leq l$. It then follows that M is a model of P. That is, if there is some clause C such that

$$C = p \leftarrow q_1, \ldots, q_m, k_1 X_1 l_1, \ldots, k_n X_n l_n$$

such that M satisfies the body of C, then for all $1 \leq i \leq n$, $k_i \leq |M \cap X_i| \leq l_i$ so that $Q^b_{k_i,l_i,X_i} \in N$. But then the clause C' in $t(P)$ where

$$C' = p \leftarrow q_1, \ldots, q_m, Q^b_{k_1,l_1,X_1}, \ldots, Q^b_{k_n,l_n,X_n}.$$

Since N is a stable model, p must be in N and hence $p \in M$. Similarly if there is a clause D such that

$$D = kXl \leftarrow q_1, \ldots, q_m, k_1X_1l_1, \ldots, k_nX_nl_n$$

such that M satisfies the body of D, then again we can conclude that $Q^b_{k_i,l_i,X_i} \in N$ for $i = 1, \ldots, n$. But then there is a clause D' is in $t(P)$ such that

$$D' = Q^h_{k,l,X} \leftarrow q_1, \ldots, q_m, Q^b_{k_1,l_1,X_1}, \ldots, Q^b_{k_n,l_n,X_n}.$$

Thus since N is a stable model, $Q^h_{k,l,X} \in N$ and hence $k \leq |M \cap X| \leq l$. Thus M satisfies kXl.

Finally one can prove by induction on the length of proof schemes that an atom $a \in At$ is in M if only if there is CC-proof scheme for P of a admitted by M. That is, suppose

$$\mathfrak{S} = \langle \langle s_1, \ldots, s_w, a \rangle, \langle C_1, \ldots, C_k, C \rangle, \{r_1, \ldots, r_k\} \rangle$$

is a proof scheme for a admitted by N. Then there are two cases to consider:

(I) $C = a \leftarrow q_1, \ldots, q_m, Q^b_{k_1,l_1,X_1}, \ldots, Q^b_{k_n,l_n,X_n}$ where

$$C' = a \leftarrow q_1, \ldots, q_m, k_1X_1l_1, \ldots, k_nX_nl_n$$

is a clause in P or

(II) $C = a \leftarrow P_{k,l,X,a}, Q^h_{k,l,X}$.

In case (I), it must be case that $\{q_1, \ldots q_m\} \subseteq \{s_1, \ldots, s_k\}$. Hence they all have shorter proof schemes admitted by N so that, for $i = 1, \ldots, m$, there are CC-proof schemes $\mathfrak{S}_i$ for q_i admitted by M. Since the only rules that have $Q^b_{k_j,l_j,X_j}$ in the head are of type (II), it follows that for each $j = 1, \ldots, n$, there must be a rule C_{t_j} with $t_j \leq k$ of the form

$$C_{t_j} = Q^b_{k_j,l_j,X_j} \leftarrow a^j_1, \ldots, a^j_{r_j}, \neg b^j_1, \ldots, \neg b^j_{s_j}.$$

Again it must be the case that $\{a^j_1, \ldots a^j_{r_j}\} \subseteq \{s_1, \ldots, s_k\}$. Hence they all have shorter proof schemes admitted by N so that, for $i = 1, \ldots, r_j$, there are CC-proof schemes $\mathfrak{T}^j_i$ for a^j_i admitted by M. We can then easily combine all these CC-proofs schemes admitted by M together with the rule C' to construct a CC-proof scheme for a admitted by M.

Similarly in case (II), it must be the case some $t \leq k$ such that (I) $C_t = Q^h_{k,l,X} \leftarrow q_1, \ldots, q_m, Q^b_{k_1,l_1,X_1}, \ldots, Q^b_{k_n,l_n,X_n}$ where

$$D_t = kXl \leftarrow q_1, \ldots, q_m, k_1X_1l_1, \ldots, k_nX_nl_n$$

is a clause in P. Again it must be case that $\{q_1, \ldots q_m\} \subseteq \{s_1, \ldots, s_k\}$. Hence they all have shorter proof schemes admitted by N so that, for $i = 1, \ldots, m$, there are CC-proof schemes $\mathfrak{S}_i$ for q_i admitted by M. Since the only rules that have $Q^b_{k_j,l_j,X_j}$ in the head are of type (II), it follows that for each $j = 1, \ldots, n$, there must be a rule C_{t_j} with $t_j \leq t$ of the form

$$C_{t_j} = Q^b_{k_j,l_j,X_j} \leftarrow a^j_1, \ldots, a^j_{r_j}, \neg b^j_1, \ldots, \neg b^j_{s_j}.$$

Again it must be the case that $\{a^j_1, \ldots a^j_{r_j}\} \subseteq \{s_1, \ldots, s_k\}$. Hence they all have shorter proof schemes admitted by N so that, for $i = 1, \ldots, r_j$, there are CC-proof schemes $\mathfrak{T}^j_i$ for a^j_i admitted by M. We can then easily combine all these CC-proofs schemes admitted by M together with the rule D_t to construct a CC-proof scheme for a admitted by M. □

4 Some complexity issues

We will now investigate some complexity issues related to CC-logic programs. In [NSS99], Niemelä, Simons and Soininen show that the existence problems for stable models of CC-logic programs is NP-complete. In fact, as we will see, a much smaller class of CC-logic programs has this property. In [FMT02], a class of *generator* CC-logic programs is introduced which consists of all CC-logic programs P such that each clause C of P is a single fact, i.e. C is of the form $p \leftarrow$, or of the form $kXl \leftarrow$. A *generator* for a set At is a generator CC-logic program P such that every atom in At occurs in some clause of P. The following fact has been proved by M. Truszczyński[FMT02].

Proposition 4.1 *Let P be a generator for the set of atoms At. Then every model of P is a CC-stable model of P.*

We notice in passing that Proposition 4.1 follows immediately from Proposition 2.3.

Next we observe the following

Proposition 4.2 *The existence problem for stable models of generator CC-logic programs is NP-complete.*

Proposition 4.2 follows from the existence of the reduction of the VERTEX COVER problem to the existence problem for CC-stable models of generator CC-logic programs. Indeed, let $G = \langle V, E \rangle$ be a graph and $k \in N$. Consider the following generator CC-logic program.

$1\{x, y\}2 \leftarrow$
$Vk \leftarrow$

Here the first clause is added for every edge $(x, y) \in E$. Moreover, V is identified with the set of atoms At. Call the resulting program $P_{G,k}$. It is then easy to see that models (and thus CC-stable models) of $P_{G,k}$ are vertex covers for G of size at most k.

Thus even the existence of models or CC-stable models for CC-logic programs and for generator CC-logic programs is NP-complete.

Finally, we will consider a slightly larger class of CC-logic programs P where the body of clause C of P contains no cardinality constraints. That is, C is of the form

$$p \leftarrow q_1, \ldots, q_m$$

or C is of the form

$$kXl \leftarrow q_1, \ldots, q_m.$$

We call such programs *semi-generator* CC-logic programs.

We will now show how to reduce the satisfiability problem for propositional logic to the existence problem for CC-stable models of semi-generator CC-logic programs. To this end, given a CNF formula $\Phi = C_1 \wedge \ldots \wedge C_m$, we will write a semi-generator CC-logic program P_Φ as follows. First let S denote the set of propositional letters that occur in Φ. For each $s \in S$, let $d(s) = \bar{s}$ and $d(\neg s) = s$. Next let p_0 be some fixed element in S and let $T = \{\bar{p} : p \in S\} \cup S$. The P_Φ consists of the following set of clauses.

(1) $1\{p, \bar{p}\}1 \leftarrow$

(2) $2\{p_0, \overline{p_0}\}2 \leftarrow d(\neg l_1^i), \ldots, d(\neg l_{n_i}^i)$

The clause (1) is added for every $p \in S$. The clause (2) is added for every clause $C_i = l_1^i \vee \cdots \vee l_{n_i}^i$ in Φ, $i = 1, \ldots, m$.

Note that the clauses of type (1) ensure that for any CC-stable model M of P_Φ, exactly one of p and $\bar{p}$ is in M for each $p \in S$. In particular, we can not have both p_0 and $\bar{p}_0$ in M. Thus if for all $s \in S$, if we interpret $s \in M$ as s being true and $\bar{s} \in M$ as s being false, then it is easy to see that clauses in (2) ensure that truth assignment determined by M must satisfy all the clauses C_i. Then it is clear that there is a one-to-one correspondence between stable models of P_Φ and valuations of S satisfying Φ. Thus we have proved the following result.

Proposition 4.3 *1. The problem consisting of pairs $\langle P, a\rangle$ where P is a semi-generator CC-logic program such that a belongs to some CC-stable model of P is NP-complete.*

2. The problem consisting of pairs $\langle P, a\rangle$ where P is a semi-generator CC-logic program such that a belongs to all CC-stable models of P is co-NP-complete.

5 Conclusions and further research

In this paper, we studied the CC-stable model semantics of CC-logic programs and showed they can be related to the stable logic semantics of ordinary logic programs. Indeed, we showed that one can view the CC-stable stable semantics as arising from the *hide* operation applied to an ordinary logic program with a larger language. While our investigation related CC-stable models of CC-logic programs to stable models of ordinary logic programs, other classes of models for CC-programs can also be studied. For instance, it is possible to define the corresponding T_P operator [AvE82] for CC-logic programs. Its fixed points may play the role of supported models. Likewise, the well-founded semantics and the alternating fixed point construction may be of interest in the case of CC-logic programs. Our investigations indicate that there will always be at least two versions of such concepts, "direct" and "indirect." That is, we can modify a given semantics so that it can be applied in the original language of the program or we can apply the usual semantics to programs in an extended language and then project. Whether these concepts are essentially the same for semantics other than stable semantics remains to be seen. One expects that there are many natural constructs for logic programs that can be extended to CC-logic programs. For example, in another paper, we will show that there are natural analogues of stratified programs for CC-logic programs.

Finally we have not tried to make our translation optimal from the complexity point of view. Instead, we opted to have the translation that would make the proof of Theorem 3.1 as simple as possible. However, complexity results for CC-logic programs indicate that there may be similar translations to one constructed in this paper but of much smaller size.

Acknowledgments

The first author's research has been partially supported by the NSF grant IIS-0097278. Any opinions, findings, and conclusions or recommendations ex-

pressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

The second author's research has been partially supported by the ARO contract DAAD19-01-1-0724.

The authors express their gratitude to the anonymous referees for reading the paper and suggesting improvements.

References

[Ap90] K.R. Apt. Logic Programming. in *Handbook of Theoretical Computer Science*, pages 475–574. Elsevier, 1990.

[AvE82] K.R. Apt and M.H. van Emden. Contributions to the theory of logic programming. *Journal of the ACM*, 29(3):841–862, 1982.

[ASP01] Proceedings of the AAAI Spring 2001 Symposium on Answer Set Programming, Stanford, CA, USA, 2001.

[FL01] P. Ferraris and V. Lifschitz. Weight constraints as nested expressions. Available at: `http://www.cs.utexas.edu/users/vl/papers.html`, Unpublished draft, 2001.

[FMT02] R. Finkel, V. Marek, and M. Truszczynski. Constraint Lingo: Towards high-level answer-set programming, Submitted for publication.

[GJ79] M.R. Garey and D.S. Johnson. *Computers and intractability; a guide to the theory of NP-completeness.* W.H. Freeman, 1979.

[GL88] M. Gelfond and V. Lifschitz. The stable semantics for logic programs. In *Proceedings of the 5th International Symposium on Logic Programming*, pages 1070–1080. MIT Press, 1988.

[KS99] H.A. Kautz and B. Selman. Unifying sat-based and graph-based planning. *Proceedings of International Joint Conference on Artificial Intelligence*, pages 318-325. Morgan Kaufmann, 1999.

[Lif99] V. Lifschitz. Answer set planning. *Proceedings of the Fifth International Conference on Logic Programming and Nonmonotonic Reasoning, El Paso, TX., Springer Lecture Notes in Computer Science* 1730, pages 373–374, 1999.

[LTT99] V. Lifschitz, L.R. Tang, and H. Turner. Nested expressions in logic programs, *Annals of Mathematics and Artificial Intelligence* 25:369–389, 1999.

[MNR90] V. Marek, A. Nerode, and J. B. Remmel. Nonomonotonic rule systems I. *Annals of Mathematics and Artificial Intelligence*, 1: 241-273, 1990.

[MT99] V. Marek and M. Truszczyński. Stable Models and an Alternative Logic Programming Paradigm. *The Logic Programming Paradigm*, pages 375–398. Series Artificial Intelligence, Springer-Verlag, 1999.

[Nie98] I. Niemelä. Logic programs with stable model semantics as a constraint programming paradigm. In *Proceedings of the Workshop on Computational Aspects of Nonmonotonic Reasoning*, pages 72–79, 1998.

[NS97] I. Niemelä and P. Simons. Smodels — an implementation of the stable model and well-founded semantics for normal logic programs, *Proceedings of the Fourth International Conference on Logic Programming and Nonmonotonic Reasoning, Dagstuhl, Germany, Springer Lecture Notes in Computer Science* 1265, pages 420–429, 1997.

[NS00] I. Niemelä and P. Simons. Extending Smodels System with Cardinality and Weight Constraints. In J. Minker, editor, *Logic-Based Artificial Intelligence*, pages 491–521. Kluwer Academic Publishers, 2000.

[NSS99] I. Niemelä, P. Simons. and T. Soininen. Stable Model Semantics of Weight Constraint Rules. *Proceedings of the Fifth International Conference on Logic Programming and Nonmonotonic Reasoning, El Paso, TX., Springer Lecture Notes in Computer Science* 1730, 1999.

[SK92] B. Selman and H. A. Kautz. Planning as satisfiability. *Proceedings of the 10th European Conference on Artificial Intelligence*, pages 359-363. Wiley, 1992.

[Syr01] T. Syrjänen. Manual of Lparse version 1.0, `http://saturn.tcs.hut.fi/Software/smodels`

[VRS91] A. Van Gelder, K.A. Ross, and J.S. Schlipf. Unfounded sets and well-founded semantics for general logic programs. *Journal of the ACM*, 38:620–650, 1991.

Alternative Characterizations for Strong Equivalence of Logic Programs

Pedro Cabalar

AI Lab., Dept. of Computer Science
University of Corunna, Spain
cabalar@dc.fi.udc.es

Abstract

In this work we present additional results related to the property of strong equivalence of logic programs. This property asserts that two programs share the same set of stable models, even under the addition of new rules. As shown in a recent work by Lifschitz, Pearce and Valverde, strong equivalence can be simply reduced to equivalence in the logic of *Here-and-There* (HT). In this paper we provide two alternatives respectively based on classical logic and 3-valued logic. The former is applicable to general rules, but not for nested expressions, whereas the latter is applicable for nested expressions but, when moving to an unrestricted syntax, it generally yields different results from HT.

1 Introduction

There is no doubt that the application of logic programming (LP) as a tool for knowledge representation has influenced in the progressive evolution of LP towards a more logical-style orientation, avoiding the initial syntactic restrictions. Think, for instance, how the stable models semantics [4] has been successively modified to cope with explicit negation and disjunctive heads [5], default negation in the head [9, 6] or, finally, the full use of nested expressions [11]. Perhaps as a result of this evolution, the following question has become interesting: when can we consider that two (syntactically different) programs Π_1 and Π_2 *represent the same knowledge*?

From a traditional LP perspective, we would say that Π_1 and Π_2 are equivalent when they share the same set of stable models like, for instance, the programs $\{p\}$ and $\{p \leftarrow not\ q\}$ whose only stable model is $\{p\}$. However, nonmonotonicity may cause them to behave in a different way in the presence of additional rules (just add fact q to both programs). Thus, if we want to check whether Π_1 and Π_2 actually represent the *same knowledge*, we must require a stronger condition, talking instead about *strong equivalence*: for any Π, the stable models of $\Pi_1 \cup \Pi$ and $\Pi_2 \cup \Pi$ coincide.

An elegant solution to this problem is the recent characterization of stable models relying on Heyting's logic of *Here-and-There* (HT). In [14], Pearce first showed that stable models can be simply seen as some kind of minimal HT models. Then, in [10], Lifschitz, Pearce and Valverde proved that, in fact, this characterization fits with the semantics for nested operators independently proposed in [11] and, what is more important, that two programs are strongly equivalent iff they have the same set of HT models.

In this paper we provide two closely related alternatives to HT that rely on classical logic and 3-valued logic (L_3), respectively. These alternatives present the advantage of using very well-known formalisms, which may help for a better insight of strong equivalence (the main emphasis of this paper), but can be useful for implementation purposes too. Unfortunately, we also show how, in both cases, their scope of applicability seems to be smaller than in the HT case. This is evident for the classical encoding we present, which can only be used as a "direct" semantics[1] for non-nested logic programs, whereas the L_3 characterization properly handles nested expressions in a direct way, but loses some important properties when nesting is also allowed for rule conditionals.

The paper is structured as follows. The next section recalls the basic definition of stable models for general (non-nested) logic programs. Section 3 describes the

[1] Application of the classical encoding to nested expressions is also possible, but only after a previous *syntactic* transformation.

classical encoding. In Sections 4 and 5 we respectively describe nested expressions and the 3-valued formalization. After that, we briefly comment the differences between the HT and L_3 interpretations. Finally, Section 7 concludes the paper. Proofs of theorems have been included in an appendix.

2 Stable models

The syntax of logic programs is defined starting from a finite set of ground atoms Σ, called the *Herbrand base*, which will serve as propositional signature. We assume that all the variables have been previously replaced by their possible ground instances. Letters a, b, c, d, p, q will be used to denote atoms in Σ, and letters I, J to denote subsets of Σ. A *logic program* is defined as a collection of rules of the shape:

$$a_1; \ldots; a_m; not\ b_1; \ldots; not\ b_n \leftarrow c_1, \ldots, c_r, not\ d_1, \ldots, not\ d_s \quad (1)$$

We call *head* (resp. *body*) to the left (resp. right) hand side of the arrow in (1). The comma and the semicolon are alternative representations of conjunction $\wedge$ and disjunction $\vee$, respectively. When $m = n = 0$ we usually write $\bot \leftarrow B$ instead of $\leftarrow B$, whereas when $r = s = 0$ we directly write H instead of $H \leftarrow$ or $H \leftarrow \top$.

Sometimes, it will be convenient to think about program rules as classical propositional formulas, where $\leftarrow$ and *not* are respectively understood as material implication and classical negation. In this way, the usual expression $I \models R$ denotes that interpretation I satisfies rule R (seen as a classical formula), whereas $I \models \Pi$ means that I is a model of the program Π (seen as a classical theory).

The *reduct* of a program Π w.r.t. some set of atoms I, written Π^I, is defined as the result of replacing in Π any default literal *not* p by $\top$, if $p \notin I$, or by $\bot$ otherwise.

Definition 1 (Stable model) *A set of atoms $I \subseteq \Sigma$ is a* stable model *of a logic program Π iff I is a minimal model of Π^I.* □

3 Strong equivalence in classical logic

We can think about the definition of stable models as a try-and-error procedure which handles (propositional) interpretations for two different purposes. On the one hand, we start from some arbitrary interpretation I^a, we can call the initial *assumption*, used to compute the reduct Π^{I^a}. On the other hand, in a second step, we deal with minimal models of Π^{I^a} which, in principle, *need not to have any connection* with I^a. Each minimal model I^p can be seen as the set of propositions we can *prove* by deductive closure using the rules in Π^{I^a}. When the proved atoms coincide with the initial assumption, $I^p = I^a$, a stable model is obtained.

In order to capture this behavior, we reify all the atoms $p \in \Sigma$ to become arguments of two unary predicates, *assumed*(p) and *proved*(p), that respectively talk about I^a and I^p. Sort variable X will be used for ranging over any propositional symbol in Σ. When considering the models of any reified formula F, we will implictly assume that they actually correspond to $F \wedge \mathrm{UNA}$, where UNA stands for the unique names assumption for sort Σ. This allows us identifying any Herbrand model M of this type of formulas with a pair[2] (I^p, I^a) so that $M[assumed] = I^a$ and $M[proved] = I^p$. Expression $M \models F$ represents again satisfaction of reified formulas – ambiguity with respect to $I \models F$ is cleared by the shape of structures and formulas.

Given this simple framework, we provide two encodings: the first one is a *completely straightforward* translation to capture stable models, whereas the second one is a stronger translation to characterize strong equivalence.

Definition 2 (First translation) *For any logic program rule R like (1), we define the classical formula $\dot{R}$ as the material implication:*

$$\Big(\bigwedge_{i=1}^{r} proved(c_i)\Big) \wedge \Big(\bigwedge_{i=1}^{s} \neg assumed(d_i)\Big) \supset \Big(\bigvee_{i=1}^{n} proved(a_i)\Big) \vee \Big(\bigvee_{i=1}^{m} \neg assumed(b_i)\Big) \quad (2)$$

Given a logic program Π, the formula $\dot{\Pi}$ stands for the conjunction of all the $\dot{R}$, for each rule $R \in \Pi$. □

Intuitively, to obtain the minimal models I^p of Π^{I^a} we can use an ordering relation among pairs $(I^p, I^a) \preceq (J^p, J^a)$ that holds when both $I^a = J^a$ is fixed and $I^p \subseteq J^p$. The corresponding models $\preceq$-minimization have a simple syntactic counterpart[3]: predicate circumscription CIRC$[\dot{\Pi}; proved]$. Second, after obtaining minimal models, we must further require $I^p = I^a$, that is, we want pairs of shape (I, I) where what we

[2] The superscripts p and a, which stand here for *proved* and *assumed*, respectively correspond to the worlds *here* and *there* in HT or to the sets of *positive* and *non-negative* atoms in L_3.

[3] See Section 2.5 in [8].

assumed results to be exactly what we proved. These pairs of shape (I, I) will be called *total*. Clearly, forcing models to be total corresponds to including of the formula:

$$\forall X.\ (proved(X) \equiv assumed(X)) \tag{3}$$

The intuitions above are not new. In fact, they were used in Theorem 5.2 in [13] which, adapted[4] to our current presentation, states the following result:

Proposition 1 *Let Σ be a propositional signature. A set of atoms $I \subseteq \Sigma$ is a stable model of a logic program Π iff $M = (I, I)$ satisfies the formula:*

$$\text{CIRC}[\dot{\Pi}; proved] \wedge (3)$$

□

In order to capture strong equivalence of two programs, it seems that we should not only compare the final selected models, but also the set of non-minimal ones involved in the minimization. For instance, it is easy to see that, due to monotonicity of classical logic, the following proposition trivially applies:

Proposition 2 *Let Π_1 and Π_2 be two logic programs such that $\models \dot{\Pi}_1 \equiv \dot{\Pi}_2$. Then Π_1 and Π_2 are strongly equivalent.* □

Unfortunately, the opposite does not necessarily hold: Π_1 and Π_2 can be strongly equivalent while $\dot{\Pi}_1$ and $\dot{\Pi}_2$ have different models. This is because encoding in Definition 2 allows some models which are actually irrelevant for strong equivalence, as we will show next.

Definition 3 (Subtotal model) *For any reified theory T, a model (I^p, I^a) of T is called* subtotal *iff (I^a, I^a) is also model of T.* □

Let $\text{SUBT}(T)$ represent the set of subtotal models of T (note that total models are also included). It is clear that any model $M \notin \text{SUBT}(\dot{\Pi})$ is irrelevant for selecting the total $\preceq$-minimal models, i.e., for obtaining the stable models of Π. The next theorem shows that the coincidence of subtotal models is a necessary condition for strong equivalence. The proof (in the Appendix) constitutes a direct rephrasing of that for the main theorem in [10].

Theorem 1 *Two logic programs Π_1 and Π_2 are strongly equivalent iff* $\text{SUBT}(\dot{\Pi}_1) = \text{SUBT}(\dot{\Pi}_2)$ □

[4] In [13] they used a duplicated signature (atoms p and p') instead of reification and, therefore, they actually applied parallel circumscription. This result seems to have been first presented in [12].

Theorem 1 points out that the $\dot{\Pi}$ encoding is still too weak for a full characterization of strong equivalence. We show next how, using a more restrictive translation (that is, adding more formulas) it is possible to obtain theories for which all their models are subtotal. To understand how to do this, consider the example program $\Pi_0 = \{p \leftarrow q\}$ where:

$$\dot{\Pi}_0 \stackrel{\text{def}}{=} proved(q) \supset proved(p)$$

This formula has 12 models: it restricts the extent of *proved* to 3 cases ($\emptyset$, $\{p\}$ and $\{p, q\}$) leaving free, in each case, the 4 possibilities for *assumed*.

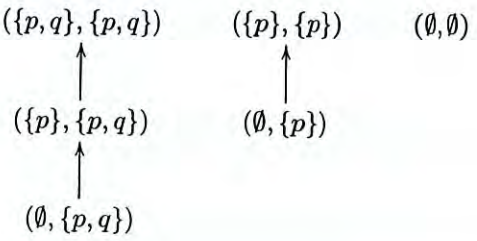


Figure 1: Subtotal models of $\dot{\Pi}_0$.

Figure 1 shows the 6 subtotal models of $\dot{\Pi}_0$, representing the $\preceq$-ordering relationships among them. Notice how subtotal models always satisfy $I^p \subseteq I^a$, that is, we can require:

$$\forall X.(proved(X) \supset assumed(X)) \tag{4}$$

Unfortunatley, the addition of this axiom is still not enough to rule out all the irrelevant models. For instance, $\dot{\Pi}_0 \wedge (4)$ has still one non-subtotal model: $(\emptyset, \{q\})$. This model, however, has the particularity that its assumed atoms $I^a = \{q\}$ do not satisfy the original program rule: $I^a \not\models p \leftarrow q$. As it is well-known, any stable model I of a program Π, is also a classical model: $I \models \Pi$. So, instead of starting from any arbitrary initial assumption I^a, we can begin requiring $I^a \models \Pi$. This can be easily incorporated into the encoding as follows. For each logic program rule R like (1), we define $\ddot{R}$ as:

$$\Big(\bigwedge_{i=1}^{r} assumed(c_i)\Big) \wedge \Big(\bigwedge_{i=1}^{s} \neg assumed(d_i)\Big) \supset \Big(\bigvee_{i=1}^{n} assumed(a_i)\Big) \vee \Big(\bigvee_{i=1}^{m} \neg assumed(b_i)\Big) \tag{5}$$

Again, $\ddot{\Pi}$ stands for the conjunction of $\ddot{R}$ for all $R \in \Pi$.

Definition 4 (Second translation) *For any logic program Π we define the formula $\Pi^* \stackrel{\text{def}}{=} \dot{\Pi} \wedge \ddot{\Pi} \wedge (4)$.* □

The proof for the following theorem uses well-known properties of circumscription (see [8]) to show that the additional formulas *do not affect* to the final set of stable models.

Theorem 2 *For any logic program* Π*:*
$\mathrm{CIRC}[\Pi^*; proved] \wedge (3) \equiv \mathrm{CIRC}[\dot{\Pi}; proved] \wedge (3)$ □

But, of course, the real interest of Π^* is that it finally rules out irrelevant models:

Property 1 *Let* Π *be a logic program. Then, any model* $M = (I^p, I^a)$ *of* Π^* *is subtotal.* □

Finally, this property, together with theorem 1, directly implies:

Theorem 3 *Two logic programs* Π_1 *and* Π_2 *are strongly equivalent iff* $\models \Pi_1^* \equiv \Pi_2^*$. □

4 Nested expressions.

The previous section has shown a way of reducing strong equivalence of logic programs into a simple equivalence test in classical logic, providing in this way a (we think) easier alternative to the HT characterization. However, although we gain in simplicity, it must be noticed that we lose in semantic quality: this classical encoding does not provide a general interpretation for program connectives but, instead, is limited to rules of shape (1). The HT characterization is clearly stronger in this sense, since it provides a direct interpretation for *any* possible nesting of program connectives.

In [11] a more general shape for program rules was considered. A *nested expression* is defined as any propositional combination of atoms with 0-ary operators $\bot$, $\top$, unary operator *not* and binary operators ',' and ';'. A logic program is now a set of rules $Head \leftarrow Body$ where *Head* and *Body* are nested expressions (notice that the rule conditional $\leftarrow$ is the only operator that cannot be nested). An example of such a rule could be, for instance:

$$a, b \leftarrow not\ (c; not\ d) \tag{6}$$

Stable models for this kind of programs can be easily described by a simple modification in the definition of program reduct. We define now Π^I as the result of replacing in Π every maximal occurrence[5] of *not* F by $\bot$ if $I \models F$ or by $\top$ otherwise. Note that the previous definition of reduct corresponds to the particular case in which F is an atom.

An interesting result derived from this modified semantics (proposition 7 in [11]) is that any program with nested expressions is strongly equivalent to some (non-nested) program, just consisting of rules like (1). To obtain this non-nested program, the following transformations are defined. Let F, G and H represent nested expressions. By $\alpha \Leftrightarrow \beta$ we mean that we replace some regular occurrence of α by β. Then, we handle the following strongly equivalent transformations:

(i) $F, G \Leftrightarrow G, F$ and $F; G \Leftrightarrow G; F$.

(ii) $(F, G), H \Leftrightarrow F, (G, H)$ and $(F; G); H \Leftrightarrow F; (G; H)$.

(iii) $F, (G; H) \Leftrightarrow (F, G); (F, H)$ and $F; (G, H) \Leftrightarrow (F; G), (F; H)$.

(iv) $not\ (F; G) \Leftrightarrow not\ F, not\ G$ and $not\ (F, G) \Leftrightarrow not\ F; not\ G$.

(v) $not\ not\ not\ F \Leftrightarrow not\ F$.

(vi) $F, \top \Leftrightarrow F$ and $F; \top \Leftrightarrow \top$.

(vii) $F, \bot \Leftrightarrow \bot$ and $F; \bot \Leftrightarrow F$.

(viii) $not\ \top \Leftrightarrow \bot$ and $not\ \bot \Leftrightarrow \top$.

(ix) $(F, G \leftarrow H) \Leftrightarrow (F \leftarrow H), (G \leftarrow H)$.

(x) $(F \leftarrow G; H) \Leftrightarrow (F \leftarrow G), (F \leftarrow H)$.

(xi) $(F \leftarrow G, not\ not\ H) \Leftrightarrow (F; not\ H \leftarrow G)$.

(xii) $(F; not\ not\ G \leftarrow H) \Leftrightarrow (F \leftarrow not\ G, H)$.

For instance, rule (6) can be successively transformed as follows:

$a, b \leftarrow not\ c, not\ not\ d.$	By (iv)
$a \leftarrow not\ c, not\ not\ d,$ $b \leftarrow not\ c, not\ not\ d.$	By (ix)
$a; not\ d \leftarrow not\ c,$ $b; not\ d \leftarrow not\ c.$	By (xi)

This treatment of nested expressions exceeds the applicability of our previous classical logic representation. From a practical point of view, such a limitation is not very important, since we can always unfold nested expressions by applying (i)-(xii). Nevertheless, from a theoretical point of view, this clearly points out that

[5] That is, any *not* F that is not in the scope of an outer *not*.

the classical encoding fails as a real semantic characterization for LP connectives.

As shown in [10], one of the important features of the HT formalization, apart from the result for strong equivalence, is that it preserves the above interpretation of nested expressions. We show next that a similar behavior can be obtained using standard 3-valued logic (L_3). Surprisingly, L_3 provides the same interpretation for nested expressions, but generally differs once free nesting of rule conditionals is allowed.

5 L_3: Three valued logic.

We will use propositional syntax plus Lukasiewicz's unary operator[6] $\boldsymbol{l}$. Intuitively, a formula $\boldsymbol{l}F$ is never unknown and points out that F is valuated to true. In this way, $\neg\boldsymbol{l}F$ would mean that "F is not true," i.e., it is either false or unknown. If F, G are L_3 formulas and p an atom of the signature Σ then:

$$p,\ \neg F,\ F \vee G,\ \top,\ \bot,\ \boldsymbol{l}F$$

are also L_3 formulas. Propositional derived operators $(\wedge, \supset, \equiv)$ are defined in the usual way.

A three valued interpretation M is a function $M : \Sigma \longrightarrow \{0, 1/2, 1\}$ assigning to each atom $p \in \Sigma$ a truth value $M(p)$ which can be 0 (false), 1/2 (unknown) or 1 (true). We will usually represent M as the pair of sets of atoms (I^p, I^a) respectively containing the *positive* (true) and *consistent* (non-false) atoms where, of course, we require consistence: $I^p \subseteq I^a$. Consequently:

$$M(p) = \begin{cases} 1 & \text{if } p \in I^p \\ 0 & \text{if } p \notin I^a \\ 1/2 & \text{otherwise} \end{cases}$$

Note that we use here the same notation as for the pairs we handled in the reified approach. This is not casual: the negative information of a 3-valued interpretation will be used to represent default negation, whereas the positive information will represent the set of proved atoms.

Definition 5 (L_3 valuation of a formula) *We extend the valuation function M to any formula F, $M(F) \in \{0, 1/2, 1\}$, so that:*

1) $M(\top) = 1$ and $M(\bot) = 0$

2) $M(\neg F) = 1 - M(F)$

3) $M(F \vee G) = max(M(F), M(G))$

4) $M(\boldsymbol{l}F) = \begin{cases} 1 & \text{if } M(F) = 1 \\ 0 & \text{otherwise} \end{cases}$

□

An interpretation M *satisfies* a formula F, written $M \models_3 F$ when $M(F) = 1$. When F is satisfied by *any* interpretation, we call it an L_3-*tautology* and write $\models_3 F$. As usual, an interpretation is a *model* of a theory when it satisfies all its formulas. Maintaining the previous terminology, a 3-valued interpretation M is called *total* iff it has the shape $M = (I, I)$, that is, it contains no unknown atoms. Clearly, when considering total interpretations, the $\boldsymbol{l}$ operator can be simply removed, and L_3 collapses into 2-valued propositional logic.

LP connectives are simply defined among the following derived operators:

$$\begin{aligned} \boldsymbol{m}F &\stackrel{\text{def}}{=} \neg\boldsymbol{l}\neg F \\ not\ F &\stackrel{\text{def}}{=} \neg\boldsymbol{m}F \\ G \leftarrow F &\stackrel{\text{def}}{=} (\boldsymbol{l}F \supset \boldsymbol{l}G) \wedge (\boldsymbol{m}F \supset \boldsymbol{m}G) \\ F \leftrightarrow G &\stackrel{\text{def}}{=} (F \leftarrow G) \wedge (G \leftarrow F) \end{aligned}$$

It is easy to check that the derived semantics for each one of these operators corresponds to:

5) $M(\boldsymbol{m}F) = \begin{cases} 1 & \text{if } M(F) \neq 0 \\ 0 & \text{otherwise} \end{cases}$

6) $M(not\ F) = \begin{cases} 1 & \text{if } M(F) = 0 \\ 0 & \text{otherwise} \end{cases}$

7) $M(G \leftarrow F) = \begin{cases} 1 & \text{if } M(F) \leq M(G) \\ 0 & \text{otherwise} \end{cases}$

8) $M(F \leftrightarrow G) = \begin{cases} 1 & \text{if } M(F) = M(G) \\ 0 & \text{otherwise} \end{cases}$

Operator $\boldsymbol{m}$ acts is the dual of $\boldsymbol{l}$, ($\boldsymbol{m}F$ can be read as "F is consistent") whereas implication $\leftarrow$ is the one proposed by Fitting [3] and Kunen [7]. When we represent some program Π inside L_3, we will consider it as a single formula consisting in the conjunction of all the program rules. Note also that when $\models_3 F \leftrightarrow G$, we can apply uniform substitution in L_3 as we would do in classical propositional logic. For instance, $\models_3 (not\ F) \leftrightarrow (\bot \leftarrow F)$ means that we can replace any occurrence of $(not\ F)$ by $(\bot \leftarrow F)$ and vice versa. Let $\circ$ and $\bullet$ be two meta-operators, any

[6] For instance, see [1], pag. 8, where $\boldsymbol{l}$ is denoted as □.

of them indistinctly representing $\boldsymbol{l}$ or $\boldsymbol{m}$. Then, the following formulas are also L_3 tautologies:

$$\circ(F \wedge G) \leftrightarrow \circ F \wedge \circ G \quad (7)$$
$$\circ(F \vee G) \leftrightarrow \circ F \vee \circ G \quad (8)$$
$$\bullet \circ F \leftrightarrow \circ F \quad (9)$$
$$\bullet \neg \circ F \leftrightarrow \neg \circ F \quad (10)$$

As $\boldsymbol{m}$ is defined in terms[7] of $\boldsymbol{l}$, this means that we can unfold any L_3 formula until $\boldsymbol{l}$ is exclusively applied to literals. Using these properties, the following lemma can be easily proved:

Lemma 1 *Let R be a pogram rule like (1), and let the pair $M = (I^p, I^a)$ have the common shape of an L_3-interpretation and a classical interpretation for proved/assumed. Then, $M \models_3 R$ iff $M \models \dot{R} \wedge \ddot{R}$.* □

Besides, by inspection on L_3 semantics, we also have that:

Lemma 2 *For any transformation $\alpha \Leftrightarrow \beta$ in* (i)-(xii): $\models_3 \alpha \leftrightarrow \beta$. □

Theorem 4 *Let Π_1 and Π_2 be two logic programs possibly containing nested expressions. Then Π_1 and Π_2 are strongly equivalent iff:* $\models_3 \boldsymbol{l}\Pi_1 \equiv \boldsymbol{l}\Pi_2$. □

Notice that we check $\boldsymbol{l}\Pi_1 \equiv \boldsymbol{l}\Pi_2$ instead of the stronger condition $\Pi_1 \leftrightarrow \Pi_2$. To understand the difference, consider $\Pi_1 = \{a\}$ and $\Pi_2 = \{a \leftarrow \top\}$. The interpretation $M = (\{a\}, \{a\})$ is the only model of both programs and so, $\models_3 \boldsymbol{l}\Pi_1 \equiv \boldsymbol{l}\Pi_2$. However, $\Pi_1 \leftrightarrow \Pi_2$ is not a tautology, since $M' = (\emptyset, \{a\})$ makes $M'(\Pi_1) = 1/2 \neq 1 = M'(\Pi_2)$.

6 Differences with respect to HT

Theorem 4 shows that HT and L_3 coincide in their interpretations of programs with nested expressions. The next natural question is, do the HT and L_3 interpretations coincide for any arbitrary theory? The answer to this question is negative, as we will show with a pair of counterexamples. Of course, due to theorem 4, these counterexamples cannot be just programs with nested expressions, as defined in Section 4. We study, for instance, a nested conditional, and the negation of a conditional.

Consider the theory consisting of the singleton formula $(a \leftarrow b) \leftarrow c$. In HT, this theory is equivalent to $(a \leftarrow b, c)$, which seems to be the most intuitive solution, whereas in L_3 it is actually equivalent to $(a; \mathit{not}\ c \leftarrow b)$. Both equivalences hold in classical propositional logic. However, for computing stable models, their behavior is quite different. For instance, the theory $\{b, (a \leftarrow c), (c \leftarrow a), ((a \leftarrow b) \leftarrow c)\}$ would have a unique stable model $\{b\}$ under the HT interpretation whereas, under L_3, an additional stable model $\{a, b, c\}$ is obtained.

The second example shows the most important problem of the L_3 interpretation: once we allow arbitrary theories, we may obtain non-subtotal models, something that does not happen[8] in HT. Let Π be the theory $\{b, \mathit{not}\ (a \leftarrow b)\}$. Its unique stable model is $\{b\}$ both in HT and L_3. However, while the pair $(\{b\}, \{b\})$ is the unique HT model[9] of Π, in L_3 there exists a second model $(\{b\}, \{a, b\})$ which is not subtotal. In other words, when using L_3 for this general syntax, the set of L_3 models does not fully characterize strong equivalence.

7 Discussion

The study of strong equivalence is probably one of the most active current topics in research in Logic Programming, as it becomes evident by the increasing amount of new results obtained recently (just to cite three examples [16, 15, 2]).

In [15], a classical logic characterization is also provided, which presents several similarities with the approach we present here. The main difference of Pearce et al's method is that it actually relies on a syntactic translation from HT into classical logic. This translation informally consists in a duplication of the atoms in the signature so that an atom p denotes our *proved* whereas an atom p' would denote *assumed*. In this paper, our initial motivation for using classical logic was to improve the presentation and the understanding. In this way, we have directly started from non-nested programs, trying to capture the definition of stable models in a way as direct as possible. As a result, our characterization does not provide an interpretation of nested connectives. In order to deal with them, we would need to apply a previous step, using transformations (i)-(xii). Pearce et al's encoding starts from HT logic, and so, deals with nested expressions (in the same way as HT does). Besides, the transformation presented in [15] has the additional advantage of being linear, while (i)-(xii) are not polynomial in the general case. Despite of these two advantages of

[7] Of course, we could have equally chosen the dual operator $\boldsymbol{m}$ as the basic one.

[8] See for instance Fact 1 in [10].

[9] In fact, the expression $\mathit{not}\ (a \leftarrow b)$ is HT-equivalent to the pair of constraints $(\bot \leftarrow \mathit{not}\ b)$ and $(\bot \leftarrow a)$.

Pearce et al's approach, it must be noticed that none of the two classical encodings can actually be considered a full-semantics for nested logic programs, since *in both cases*, a previous syntactic transformation is required. Therefore, translation to classical logic is very interesting for practical purposes, but is limited from a purely semantic point of view.

Another similarity between our classical encoding with respect to [15] is, not only how to decide strong equivalence, but how to obtain stable models. In our Section 3 we simply used to that purpose the result presented by Lin and Shoham in [13] and then included slight variations that we proved to be sound. In [15], a quantified boolean formula is used instead:

$$\phi' \wedge \neg\exists V((V < V') \wedge \tau_{HT}[\phi]) \qquad (11)$$

where V is the set of atoms, ϕ is the original program, ϕ' results from replacing any atom p by p' and finally $\tau_{HT}[\phi]$ is Pearce et al's translation from HT to classical logic. On the other hand, Lin and Shoham's result involving circumscription can be formulated[10] as:

$$(V = V') \wedge C[\phi] \wedge \neg\exists V((V < V') \wedge C[\phi]) \qquad (12)$$

where $C[\phi]$ simply replaces each *not* p by $\neg p'$. Notice how, at least structurally, (12) is very similar to (11).

As for the L_3 encoding, it must also be noticed that other logical characterizations have been obtained apart from HT. In [2], for instance, they use instead another logic, KC, and show that this is, in fact, the weakest intermediate logic (between intuitionistic and classical) that allows capturing strong equivalence of logic programs with nested expressions. An interesting open question is how logic KC deals with nested conditionals since, as we have shown, this is the case where HT and L_3 diverge.

The interest of the encoding we have presented is perhaps the use of a multi-valued logic which is commonly known and understood and yields a natural characterization. We believe that the use of L_3 may be especially suitable as a valid alternative for "by hand" studies of strong equivalence, since it allows applying intuitive properties of $\boldsymbol{l}$ and $\boldsymbol{m}$ operators (as an example, look at the proofs in the appendix) in a more or less intuitive way.

Acknowledgements

I want to thank Vladimir Lifschitz for his discussions and comments about a preliminary draft of the L_3 encoding, and to the anonymous referees for drawing my attention to part of the related work cited in this paper. This research is partially supported by the Government of Spain, grant TIC2001-0393.

[10] As described for instance in [8], propositional circumscription is nothing else but a quantified boolean formula.

References

[1] R. Bull and K. Segerberg. Basic modal logic. In D.M. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, volume 2, pages 1–88. D. Reidel Publishing Company, 1984.

[2] A. Hendriks D.H.J. de Jongh. Characterization of strongly equivalent logic programs in intermediate logics, 2001. Unpublished draft. `http://turing.wins.uva.nl/~lhendrik/` .

[3] M. Fitting. A kripke-kleene semantics for logic programs. *Journal of Logic Programming*, 2(4):295–312, 1985.

[4] M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In R. A. Kowalski and K. A. Bowen, editors, *Logic Programming: Proc. of the Fifth International Conference and Symposium (Volume 2)*, pages 1070–1080. MIT Press, Cambridge, MA, 1988.

[5] M. Gelfond and V. Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, pages 365–385, 1991.

[6] K. Inoue and C. Sakama. Negation as failure in the head. *Journal of Logic Programming*, 35(1):39–78, 1998.

[7] K. Kunen. Negation in logic programming. *Journal of Logic Programming*, 4(4):289–308, 1987.

[8] V. Lifschitz. Circumscription. In C.J. Hogger D.M. Gabbay and J.A. Robinson, editors, *Handbook of Logic in AI and Logic Programming*, volume 3, pages 298–352. Oxford University Press, 1993.

[9] V. Lifschitz. Minimal belief and negation as failure. *Artificial Intelligence*, 70:53–72, 1994.

[10] V. Lifschitz, D. Pearce, and A. Valverde. Strongly equivalent logic programs. *ACM Transactions on Computational Logic*, 2000. (to appear).

[11] V. Lifschitz, L. R. Tang, and H. Turner. Nested expressions in logic programs. *Annals of Mathematics and Artificial Intelligence*, 25:369–389, 1999.

[12] F. Lin. *A Study of Nonmonotonic Reasoning*. PhD thesis, Stanford, 1991.

[13] F. Lin and Y. Shoham. A logic of knowledge and justified assumptions. *Artificial Intelligence*, 57:271–289, 1992.

[14] D. Pearce. A new logical characterisation of stable models and answer sets. In *Non monotonic extensions of logic programming. Proc. NMELP'96. (LNAI 1216)*. Springer-Verlag, 1997.

[15] D. Pearce, H. Tompits, and S. Woltran. Encodings of equilibrium logic and logic programs with nested expressions. In P. Bradzil and A. Jorge, editors, *Lecture Notes in Artificial Intelligence*, volume 2258, pages 306–320. Springer Verlag, 2001.

[16] H. Turner. Strong equivalence for logic programs and default theories (made easy). In *In Proc. of the Sixth Int'l Conf. on Logic Programming and Nonmonotonic Reasoning (LPNMR'01)*, pages 81–92, 2001.

Appendix A. Proofs.

Proof of theorem 1
The right to left direction was proved in proposition 2. For the left to right direction, assume there is some $M = (I^p, I^a)$, $M \in \text{SUBT}(\dot{\Pi}_1)$ but $M \notin \text{SUBT}(\dot{\Pi}_2)$. Let M' be the total pair $M' = (I^a, I^a)$. We will show that it is possible to construct a program Π so that I^a is a stable model of $\Pi_1 \cup \Pi$ but not of $\Pi_2 \cup \Pi$, or vice versa. We consider two cases:

1. $M' \notin \text{SUBT}(\dot{\Pi}_2)$. As $M \in \text{SUBT}(\dot{\Pi}_1)$, by definition of SUBT, M' is model of $\dot{\Pi}_1$. We construct the program $\Pi = \{p : p \in I^a\}$. Clearly, M' is still model of $\dot{\Pi}_1 \wedge \Pi$ but, now, it is also minimal: a strict subset of *proved* atoms is not allowed. Therefore, I^a is a stable model of $\Pi_1 \cup \Pi$. On the other hand, as M' is total, but $M' \notin \text{SUBT}(\dot{\Pi}_2)$, then $M' \not\models \dot{\Pi}_2$. As a result, I^a cannot be stable model of $\Pi_2 \cup \Pi$.

2. $M' \in \text{SUBT}(\dot{\Pi}_2)$. Note first that, as $M \notin \text{SUBT}(\dot{\Pi}_2)$ and $M \preceq M'$ we conclude $M \not\models \dot{\Pi}_2$ and $M \prec M'$. We construct the program:

$$\begin{aligned}\Pi \;=\; & \{p : p \in I^p\} \cup \\ & \{(p \leftarrow q) : p, q \in (I^a - I^p), p \neq q\}\end{aligned}$$

It is easy to see that $M' \models \dot{\Pi}$ and so, $M' \models \dot{\Pi}_2 \wedge \dot{\Pi}$. We will show that M' is also minimal for that theory. Let us assume that there exists another model (J, I^a) strictly lower than M', $J \subset I^a$, and try to reach a contradiction. As atoms of I^p are facts of Π, but $M = (I^p, I^a)$ is not model of $\dot{\Pi}_2$, we must have: $I^p \subset J$. Now, take some $q \in J - I^p$ and some $p \in I^a - J$. The rule $p \leftarrow q$ belongs to Π but (J, I^a) does not satisfy the rule translation. Therefore, $(J, I^a) \not\models \dot{\Pi}$ and we reach a contradiction. Since we have proved that M' is minimal, I^a is stable model of $\dot{\Pi}_2 \cup \dot{\Pi}$. Finally, we study program Π_1. It can be easily checked that $M \models \dot{\Pi}$ and so $M \models \dot{\Pi}_1 \cup \dot{\Pi}$. But as $M \prec M'$, M' is not minimal for $\dot{\Pi}_1 \cup \dot{\Pi}$ and so I^a is not stable model for this program.

Proof of theorem 2
We start from $\text{CIRC}[\Pi^*; \mathit{proved}] \wedge (3)$, observing that the conjuncts $\ddot{\Pi}$ and (4) of Π^* can be moved outside the circumscription. The reason for this is that in $\ddot{\Pi}$, predicate *proved* does not occur, whereas in (4) it occurs negatively (considering $\alpha \supset \beta$ as $\neg\alpha \vee \beta$). Therefore, we can respectively apply equivalence (3.2) and proposition 3.3.1 in [8], to obtain the equivalent formula:

$$\text{CIRC}[\dot{\Pi}; \mathit{proved}] \wedge \ddot{\Pi} \wedge (4) \wedge (3)$$

Now, notice that, by definition, the circumscription of $\text{CIRC}[\dot{\Pi}; \mathit{proved}]$ has the shape of a conjunction $\dot{\Pi} \wedge \alpha$ (where α is a second order formula we do not need to detail). So, the above formula is actually:

$$\dot{\Pi} \wedge \alpha \wedge \ddot{\Pi} \wedge (4) \wedge (3)$$

Finally, notice that (3) implies (4), whereas $\dot{\Pi} \wedge (3)$ implies $\ddot{\Pi}$, and so, the implied conjuncts can be removed, obtaining the equivalent formula:

$$\dot{\Pi} \wedge \alpha \wedge (3)$$

which is nothing else but $\text{CIRC}[\dot{\Pi}; \mathit{proved}] \wedge (3)$.

Proof of property 1
We must show that $M' = (I^a, I^a)$ is also model of Π^*. Clearly, as $\ddot{\Pi}$ exclusively refers to predicate *assumed*, $M \models \ddot{\Pi}$ iff $M' \models \ddot{\Pi}$. On the other hand, it is clear that $M' \models (4)$ – in fact, it further satisfies (3). So, we only have to prove that $M' \models \dot{\Pi}$. Assume that, for some rule $R \in \Pi$, $M' \not\models \dot{R}$, which in classical logic is equivalent to $M' \models \neg(\dot{R})$, i.e., M' must satisfy:

$$\begin{aligned}&\Big(\bigwedge_{i=1}^{r} \mathit{proved}(c_i)\Big) \wedge \Big(\bigwedge_{i=1}^{s} \neg \mathit{assumed}(d_i)\Big) \wedge \\ &\qquad \Big(\bigwedge_{i=1}^{n} \neg \mathit{proved}(a_i)\Big) \wedge \Big(\bigwedge_{i=1}^{m} \mathit{assumed}(b_i)\Big)\end{aligned}$$

As in M', *proved* and *assumed* are equivalent, this implies that M' must also satisfy:

$$(\bigwedge_{i=1}^{r} assumed(c_i)) \wedge (\bigwedge_{i=1}^{s} \neg assumed(d_i)) \wedge (\bigwedge_{i=1}^{n} \neg assumed(a_i)) \wedge (\bigwedge_{i=1}^{m} assumed(b_i))$$

But this formula is exactly $\neg(\ddot{R})$ and so, M' cannot be model of $\ddot{\Pi}$ nor Π^*, reaching a contradiction.

Proof of lemma 1

Using properties (7)-(10) and the definitions of derived operators, a general logic program rule like (1) is equivalent to the conjunction of the pair of formulas:

$$\boldsymbol{l}c_1 \wedge \cdots \wedge \boldsymbol{l}c_r \wedge \neg \boldsymbol{m}d_1 \wedge \cdots \wedge \neg \boldsymbol{m}d_s \supset \boldsymbol{l}a_1 \vee \cdots \vee \boldsymbol{l}a_n \vee \cdots \vee \neg \boldsymbol{m}b_1 \vee \cdots \vee \neg \boldsymbol{m}b_m \quad \text{(A-1)}$$

$$\boldsymbol{m}c_1 \wedge \cdots \wedge \boldsymbol{m}c_r \wedge \neg \boldsymbol{m}d_1 \wedge \cdots \wedge \neg \boldsymbol{m}d_s \supset \boldsymbol{m}a_1 \vee \cdots \vee \boldsymbol{m}a_n \vee \cdots \vee \neg \boldsymbol{m}b_1 \vee \cdots \vee \neg \boldsymbol{m}b_m \quad \text{(A-2)}$$

Note that in formulas, operators $\boldsymbol{l}$ and $\boldsymbol{m}$ are only applied to atoms and, vice versa, atoms only appear in the scope of $\boldsymbol{l}$ and $\boldsymbol{m}$. As the interpretation of $\boldsymbol{l}$ and $\boldsymbol{m}$ is always bivalued, we can just consider the meaning of each $\boldsymbol{l}p$ and $\boldsymbol{m}p$ individually and then interprete the whole formula as a classical propositional one. Now, note that $\boldsymbol{l}p$ is true iff $p \notin I^p$ that is $M \models proved(a)$, and $\boldsymbol{m}p$ is true iff $p \in I^a$, that is, $M \models assumed(a)$. Therefore, (A-1) and (A-2) respectively correspond to $\dot{R}$ and $\ddot{R}$.

Proof of lemma 2

As ',' and ';' are simply understood as $\wedge$ and $\vee$, properties (i)-(iii), (vi) and (vii) are straightforward.

(iv) The formula *not* $(F;G)$ is, by definition, $\neg\boldsymbol{m}(F \vee G)$ which, by (8), is always valuated in the same way as $\neg(\boldsymbol{m}F \vee \boldsymbol{m}G)$. As L_3 satisfies the De Morgan's laws, this is equivalent to $\neg\boldsymbol{m}F \wedge \neg\boldsymbol{m}G$, which by definition, is the formula *not* F, *not* G. The other part of property (iv) is analogous.

(v) The formula *not not not* F is, by definition, $\neg\boldsymbol{m}\neg\boldsymbol{m}\neg\boldsymbol{m}F$. By (10), we replace $\boldsymbol{m}\neg\boldsymbol{m}F$ by $\neg\boldsymbol{m}F$ obtaining $\neg\boldsymbol{m}\neg\neg\boldsymbol{m}F$, that is, $\neg\boldsymbol{m}\boldsymbol{m}F$. Now, by (9) this is equivalent to $\neg\boldsymbol{m}F$, which is, by definition *not* F.

(viii) The formula *not* $\bot$ is $\neg\boldsymbol{m}\bot$. As $\boldsymbol{m}\bot$ is always valuated as 0 (false), this formula is equivalent to $\neg\bot$, that is, $\top$. The other part of (viii) is analogous.

(ix) The formula $(F, G \leftarrow H)$ is, by definition:

$$(\boldsymbol{l}H \supset \boldsymbol{l}(F \wedge G)) \wedge (\boldsymbol{m}H \supset \boldsymbol{m}(F \wedge G))$$

Applying (7) we obtain:

$$(\boldsymbol{l}H \supset \boldsymbol{l}F \wedge \boldsymbol{l}G) \wedge (\boldsymbol{m}H \supset \boldsymbol{m}F \wedge \boldsymbol{m}G)$$

This is equivalent to:

$$(\neg\boldsymbol{l}H \vee \boldsymbol{l}F) \wedge (\neg\boldsymbol{l}H \vee \boldsymbol{l}G) \wedge (\neg\boldsymbol{m}H \vee \boldsymbol{m}F) \wedge (\neg\boldsymbol{m}H \vee \boldsymbol{m}G)$$

(simply unfold the definition of $\supset$ and apply distributivity). Finally, note that, by definition of $\leftarrow$, the last formula is just:

$$(F \leftarrow H) \wedge (G \leftarrow H)$$

(ix) We just provide a proof sketch:

$(\boldsymbol{l}(G \vee H) \supset \boldsymbol{l}F) \wedge (\boldsymbol{m}(G \vee H) \supset \boldsymbol{m}F)$	Def. of $\leftarrow$
$(\boldsymbol{l}G \vee \boldsymbol{l}H \supset \boldsymbol{l}F) \wedge (\boldsymbol{m}G \vee \boldsymbol{m}H \supset \boldsymbol{m}F)$	(8)
$(\boldsymbol{l}G \supset \boldsymbol{l}F) \wedge (\boldsymbol{m}G \supset \boldsymbol{m}F) \wedge (\boldsymbol{l}H \supset \boldsymbol{l}F) \wedge (\boldsymbol{m}H \supset \boldsymbol{m}F)$	Def. of $\supset$
$(F \leftarrow G) \wedge (F \leftarrow H)$	Def. of $\leftarrow$

(x) Again, as a proof sketch:

$(\boldsymbol{l}(G \wedge \neg\boldsymbol{m}\neg\boldsymbol{m}H) \supset \boldsymbol{l}F) \wedge (\boldsymbol{m}(G \wedge \neg\boldsymbol{m}\neg\boldsymbol{m}H) \supset \boldsymbol{m}F)$	Def. of $\leftarrow$, *not*
$(\boldsymbol{l}(G \wedge \neg\neg\boldsymbol{m}H) \supset \boldsymbol{l}F) \wedge (\boldsymbol{m}(G \wedge \neg\neg\boldsymbol{m}H) \supset \boldsymbol{m}F)$	(10)
$(\boldsymbol{l}G \wedge \boldsymbol{l}\boldsymbol{m}H \supset \boldsymbol{l}F) \wedge (\boldsymbol{m}G \wedge \boldsymbol{m}\boldsymbol{m}H \supset \boldsymbol{m}F)$	(7)
$(\boldsymbol{l}G \wedge \boldsymbol{m}H \supset \boldsymbol{l}F) \wedge (\boldsymbol{m}G \wedge \boldsymbol{m}H \supset \boldsymbol{m}F)$	(9)
$(\boldsymbol{l}G \supset \boldsymbol{l}F \vee \neg\boldsymbol{m}H) \wedge (\boldsymbol{m}G \supset \boldsymbol{m}F \vee \neg\boldsymbol{m}H)$	Def. $\supset$
$F;$ *not* $H \leftarrow G$	Def. of $\leftarrow$, *not*

(xi) It is analogous to proof for (x).

Proof of theorem 4

First, from lemma 2 the corresponding non-nested program has the same set of L_3 models. Second, lemma 1 shows that this program has the same models than our classical encoding. Therefore, we can directly apply theorem 3.

Preferred well-founded semantics for logic programming by alternating fixpoints: Preliminary Report

Torsten Schaub* **and Kewen Wang**
Institut für Informatik, Universität Potsdam
Postfach 900327, D–14439 Potsdam, Germany
{torsten,kewen}@cs.uni-potsdam.de

Abstract

We analyze the problem of defining well-founded semantics for ordered logic programs within a general framework based on alternating fixpoint theory. We start by showing that generalizations of existing answer set approaches to preference are too weak in the setting of well-founded semantics. We then specify some informal yet intuitive criteria and propose a semantical framework for preference handling that is more suitable for defining well-founded semantics for ordered logic programs. The suitability of the new approach is convinced by the fact that many attractive properties are satisfied by our semantics. In particular, our semantics is still correct with respect to various existing answer sets semantics while it successfully overcomes the weakness of their generalization to well-founded semantics. Finally, we indicate how an existing preferred well-founded semantics can be captured within our semantical framework.

Keywords: well-founded semantics, preference, alternating fixpoints, extended logic programs.

1 Introduction

Preferences constitute a very natural and effective way of resolving indeterminate situations. For example, in scheduling not all deadlines may be simultaneously satisfiable, and in configuration various goals may not be simultaneously met. Preferences among deadlines and goals may allow for an acceptable, non-optimal solution. In legal reasoning, laws may apply in different situations, but laws may also conflict with each other. Conflicts are resolved by appeal to higher-level principles such as authority or recency. So federal laws will have a higher priority than state laws, and newer laws will take priority over old. Further preferences, such as authority holding sway over recency, may also be required. In fact, while logical preference handling constitutes already an indispensable means in legal reasoning systems (cf. [16, 22]), it is also advancing in other application areas such as intelligent agents and e-commerce [18], information-site selection [14], and the resolution of grammatical ambiguities [11].

The increasing practical interest in preferences is also reflected by the large number of proposals for preference handling in logic programming, including [23, 6, 15, 31, 17, 8, 13, 28], and related areas, such as default logic [3, 5, 12]. A common approach in such work has been to employ metaformalisms for characterizing "preferred answer sets". This has led to a diversity of approaches that are hardly comparable due to considerably different methods of formal characterization. As a consequence, there is no homogeneous account of preference.

In [24], we started addressing this shortcoming by proposing a uniform semantical framework for extended logic programming with preferences. To be precise, we develop an (alternating) fixpoint theory for so-called *ordered logic programs*, building on the basic ideas in [27]. An ordered logic program is an extended logic program whose rules are subject to a strict partial order. In analogy to standard logic programming, such a program is then interpreted by means of an associated fixpoint operator. Different semantics are obtained by distinguishing different subsets of the respective set of alternating fixpoints. As a result, several different approaches to defining preferred answer sets, including [9, 10, 13], can all be captured within our framework and each of these preference strategies is based on an operator, which plays the same role as the consequence operator in the setting of normal logic programs.

In this paper, we show that the preference strategies for

*Affiliated with the School of Computing Science at Simon Fraser University, Burnaby, Canada.

defining answer sets turn out to be too weak in the setting of well-founded semantics. For this reason, we propose a new approach to preference handling for logic programs that seems to be more appropriate for well-founded semantics. In fact, we show that for a resulting instance of this approach some attractive properties. We also discuss the relation of our preferred well-founded semantics to other approaches [21, 6, 30].

2 Definitions and notation

An *extended logic program* is a finite set of rules of the form

$$L_0 \leftarrow L_1, \ldots, L_m, not\ L_{m+1}, \ldots, not\ L_n, \quad (1)$$

where $n \geq m \geq 0$, and each L_i $(0 \leq i \leq n)$ is a *literal*, ie. either an atom A or the negation $\neg A$ of A. The set of all literals is denoted by Lit. Given a rule r as in (1), we let $head(r)$ denote the *head*, L_0, of r and $body(r)$ the *body*, $\{L_1, \ldots, L_m, not\ L_{m+1}, \ldots, not\ L_n\}$, of r. Further, let $body^+(r) = \{L_1, , \ldots, L_m\}$ and $body^-(r) = \{L_{m+1}, \ldots, L_n\}$. A program is called *basic* if $body^-(r) = \emptyset$ for all its rules; it is called *normal* if it contains no classical negation symbol $\neg$.

We define the reduct of a rule r as $r^+ = head(r) \leftarrow body^+(r)$. The *reduct*, Π^X, of a program Π *relative to* a set X of literals is defined by

$$\Pi^X = \{r^+ \mid r \in \Pi \text{ and } body^-(r) \cap X = \emptyset\}.$$

A set of literals X is *closed under* a basic program Π iff for any $r \in \Pi$, $head(r) \in X$ whenever $body^+(r) \subseteq X$. We say that X is *logically closed* iff it is either consistent (ie. it does not contain both a literal A and its negation $\neg A$) or equals Lit. The smallest set of literals which is both logically closed and closed under a basic program Π is denoted by $Cn(\Pi)$. With these formalities at hand, we can define *answer set semantics* for extended logic programs: A set X of literals is an *answer set* of a program Π iff $Cn(\Pi^X) = X$.

For capturing even more semantics within a similar framework, van Gelder defines in [27] the operator $C_\Pi(X)$ as $Cn(\Pi^X)$. It is important to note that the operator C_Π is anti-monotonic, which implies that the operator $A_\Pi(X) = C_\Pi(C_\Pi(X))$ is monotonic. A fixpoint of A_Π is called an *alternating fixpoint* for Π. Different semantics are captured by distinguishing different groups of fixpoints of A_Π. For instance, given a program Π, the least alternating fixed point of A_Π is known to amount to its *well-founded semantics*. Answer sets of Π are simply alternating fixed points of A_Π that are also fixed points of C_Π.

Alternative inductive characterizations for the operators Cn, C_Π, and A_Π can be obtained by appeal to *immediate consequence operators* [26, 19]. Let Π be a basic program and X a set of literals. The *immediate consequence operator* T_Π is defined as follows:

$$T_\Pi X = \{head(r) \mid r \in \Pi \text{ and } body(r) \subseteq X\}$$

if X is consistent, and $T_\Pi X = Lit$ otherwise. Iterated applications of T_Π are written as T_Π^j for $j \geq 0$, where $T_\Pi^0 X = X$ and $T_\Pi^i X = T_\Pi T_\Pi^{i-1} X$ for $i \geq 1$. It is well-known that $Cn(\Pi) = \bigcup_{i \geq 0} T_\Pi^i \emptyset$, for any basic program Π. Also, for any answer set X of program Π, it holds that $X = \bigcup_{i \geq 0} T_{\Pi^X}^i \emptyset$.

A reduction from extended to basic programs is avoidable with an extended consequence operator: Let Π be an extended program and X and Y be sets of literals. The *extended immediate consequence operator* $T_{\Pi,Y}$ is defined as follows:

$$T_{\Pi,Y} X = \{head(r) \mid r \in \Pi,\ body^+(r) \subseteq X, \text{ and } body^-(r) \cap Y = \emptyset\} \quad (2)$$

if X is consistent, and $T_{\Pi,Y} X = Lit$ otherwise. Iterated applications of $T_{\Pi,Y}$ are written as those of T_Π above. Clearly, we have $T_{\Pi,\emptyset} X = T_\Pi X$ for any basic program Π and $T_{\Pi,Y} X = T_{\Pi^Y} X$ for any extended program Π. Accordingly, we have for any answer set X of program Π that $X = \bigcup_{i \geq 0} T_{\Pi,X}^i \emptyset$. Finally, for dealing with the individual rules in (2), we rely on the notion of *activeness*:[1] Let $X, Y \subseteq Lit$ be two sets of literals in a program Π. A rule r in Π is *active* wrt the pair (X, Y), if $body^+(r) \subseteq X$ and $body^-(r) \cap Y = \emptyset$. Alternatively, we thus have that $T_{\Pi,Y} X = \{head(r) \mid r \in \Pi \text{ is active wrt } (X, Y)\}$.

Lastly, an *ordered logic program*[2] is simply a pair $(\Pi, <)$, where Π is an extended logic program and $< \subseteq \Pi \times \Pi$ is an irreflexive and transitive relation. Given, $r_1, r_2 \in \Pi$, the relation $r_1 < r_2$ is meant to express that r_2 has *higher priority* than r_1.[3]

3 Preferred (alternating) fixpoints

We start by describing the semantical framework given in [24], while concentrating on the formal details needed for capturing the approach introduced in [28]. The formal development of the approach in [8] and [13] is analogous and thus omitted here.

[1] Although *activeness* is implicitly present in standard logic programming (cf. definition of $T_{\Pi,Y} X$), the term as such was (to the best of our knowledge) coined in approaches dealing with preferences in default logic [3, 5]. There, however, activeness additionally stipulated that $head(r) \notin X$ in order to prevent multiple applications of the same rule.

[2] Also called *prioritized* logic program by some authors, as eg. in [31, 8].

[3] Some authors, eg. [8], attribute relation $<$ the inverse meaning.

The overall idea behind the obtained semantics for ordered logic program is to distinguish the "preferred" answers of a program $(\Pi, <)$ by means of fixpoint equations. That is, a set of literals X constitutes a collection of preferred answers from $(\Pi, <)$, if it satisfies the equation $C_{(\Pi,<)}(X) = X$ for some operator $C_{(\Pi,<)}$. In view of the classical logic programming approach described in Section 2, this makes us investigate semantics that interpret preferences as inducing selection functions on the set of standard answer sets of the underlying non-ordered program Π.

Standard answer sets are defined via a reduction of extended logic programs to basic programs. Such a reduction is inappropriate when resolving conflicts among rules by means of preferences since all such conflicts are simultaneously resolved when turning Π into Π^X. Rather conflict resolution must be addressed among the original rules in order to account for blockage between rules. In fact, once the negative body $body^-(r)$ is eliminated there is no way to detect whether $head(r') \in body^-(r)$ holds in case of $r < r'$. Our idea is therefore to characterize preferred answer sets by an inductive development that agrees with the given ordering rather than a simultaneous reduction. In terms of a standard answer set X, this means that we favor its formal characterization as $X = \bigcup_{i \geq 0} T^i_{\Pi,X}\emptyset$ over $X = Cn(\Pi^X)$. This leads us to the following definition. [4]

Definition 1 *Let $(\Pi, <)$ be an ordered logic program and let X and Y be sets of literals.*

We define the set of immediate consequences of X with respect to $(\Pi, <)$ and Y as

$$\mathcal{T}_{(\Pi,<),Y}X = \left\{ head(r) \;\middle|\; \begin{array}{ll} I. & r \text{ active wrt } (X,Y); \\ II. & \text{there is no rule } r' \\ & \text{with } r < r' \text{ such that} \\ & (a)\ r' \text{ active wrt } (Y,X) \\ & (b)\ head(r') \notin X \end{array} \right\}$$

if X is consistent, and $\mathcal{T}_{(\Pi,<),Y}X = Lit$ otherwise.

Note that $\mathcal{T}_{(\Pi,<),Y}$ is a refinement of its classical counterpart $T_{\Pi,Y}$. To see this, observe that Condition *I* embodies the standard application condition for rules given in (2)

The actual refinement takes place in Condition *II*. The idea is to apply a rule r only if the "question of applicability" has been settled for all higher-ranked rules r'. Let us illustrate this in terms of iterated applications of $\mathcal{T}_{(\Pi,<),Y}$. In these cases, X contains the set of conclusions that have been derived so far, while Y provides the putative answer set (or: $Lit \setminus Y$ provides a set of literals that can be falsified). Then, the "question of applicability" is considered to be settled for a higher ranked rule r'

- if the prerequisites of r' will never be derivable, viz. $body^+(r') \not\subseteq Y$, or
- if r' is defeated by what has been derived so far, viz. $body^-(r) \cap X \neq \emptyset$, or
- if r' or another rule with the same head have already applied, viz. $head(r') \in X$.

The first two conditions show why activeness of r' is stipulated wrt (Y, X), as opposed to (X, Y) in Condition *I*. The last condition serves somehow two purposes: First, it detects whether the higher ranked rule r' has applied and, second, it suspends the preference $r < r'$ whenever the head of the higher ranked has already been derived by another rule. This suspension of preference constitutes a distinguishing feature of the approach at hand; this is discussed in detail in [24] in connection with other approaches to preference handling.

As with T_Π and $T_{\Pi,Y}$, iterated applications of $\mathcal{T}_{(\Pi,<),Y}$ are written as $\mathcal{T}^j_{(\Pi,<),Y}$ for $j \geq 0$, where $\mathcal{T}^0_{(\Pi,<),Y}X = X$ and $\mathcal{T}^i_{(\Pi,<),Y}X = \mathcal{T}_{(\Pi,<),Y}\mathcal{T}^{i-1}_{(\Pi,<),Y}X$ for $i \geq 1$. This allows us to define the counterpart of fixpoint operator C_Π for ordered programs:

Definition 2 *Let $(\Pi, <)$ be an ordered logic program and let X be a set of literals.*

We define $C_{(\Pi,<)}(X) = \bigcup_{i \geq 0} \mathcal{T}^i_{(\Pi,<),X}\emptyset$.

In analogy to $\mathcal{T}_{(\Pi,<),Y}$ and $T_{\Pi,Y}$, operator $C_{(\Pi,<)}$ is a refinement of its classical counterpart C_Π. The major difference of our definition from van Gelder's is that we directly obtain the consequences from Π (and Y). Unlike this, the usual approach (without preferences) first obtains a basic program Π^Y from Π and then the consequences are derived from this basic program Π^Y.

A preferred answer set is defined as a fixpoint of $C_{(\Pi,<)}$.

In analogy to van Gelder [27], we may define the *alternating transformation* for an ordered logic program $(\Pi, <)$ as $\mathcal{A}_{(\Pi,<)}(X) = C_{(\Pi,<)}(C_{(\Pi,<)}(X))$. A fixpoint of $\mathcal{A}_{(\Pi,<)}$ is called an *alternating fixpoint* of $(\Pi, <)$. Given that $C_{(\Pi,<)}$ is anti-monotonic [24], we get that $\mathcal{A}_{(\Pi,<)}(X)$ is monotonic. According to results tracing back to Tarski [25], this implies that $\mathcal{A}_{(\Pi,<)}$ possesses a least and a greatest fixpoint, denoted by $lfp\,\mathcal{A}_{(\Pi,<)}$ and $gfp\,\mathcal{A}_{(\Pi,<)}$, respectively.

Different semantics of ordered logic programs are obtained by distinguishing different subsets of the respective set of alternating fixpoints. In fact, the preferred answer set semantics constitute instances of the overall framework. To

[4] Fixpoint operators for the approaches in [8] and [13] are obtained by appropriate modifications to Condition I and II in Definition 1; cf. [24].

see this, observe that each fixpoint of $C_{(\Pi,<)}$ is also a fixpoint of $\mathcal{A}_{(\Pi,<)}$.

4 Preferring least alternating fixpoints?

Let us now investigate the least alternating fixpoint of $\mathcal{A}_{(\Pi,<)}$and with it the comportment of the previous fixpoint operator in the setting of *well-founded* semantics. As opposed to answer sets semantics, this semantics relies on 3-valued models (or, partial models). Such a model consists of three parts: the set of true literals, the set of false literals, and the set of unknown literals. Given that the union of these three sets is Lit, it is sufficient to specify two of the three sets for determining a 3-valued interpretation. Accordingly, a 3-valued interpretation I is a pair (X, Y) where X and Y are sets of literals with $X \cap Y = \emptyset$. That is, $L \in X$ means that L is true in I, while $L \in Y$ means that L is false in I. Otherwise, L is considered to be unknown in I.

Well-founded semantics constitutes another major semantics for logic programs. In contrast to answers sets semantics, it aims at characterizing skeptical conclusions comprised in a single so-called *well-founded* model of the underlying program. This model can be characterized within the alternating fixpoint theory in terms of the least fixpoint of operator A_Π. That is, the well-founded model of a program Π is given by the 3-valued interpretation $(lfp\, A_\Pi, Lit \setminus C_\Pi lfp\, A_\Pi)$. Hence, it is sufficient to consider the least alternating fixpoint of a program, since it determines its well-founded model. We therefore refer to the least alternating fixpoint of Π as the *well-founded set* of Π. The set $Lit \setminus C_\Pi lfp\, A_\Pi$ is usually referred to as the *unfounded* set of Π.

After extending these concepts to preference handling, that is, substituting the classical operators A_Π and C_Π by $\mathcal{A}_{(\Pi,<)}$ and $C_{(\Pi,<)}$, respectively, one can show that (i) each ordered logic program has a unique preferred well-founded model; (ii) the preferred well-founded set is contained in any preferred answer set (while the unfounded one is not); and (iii) whenever we obtain a two-valued well-founded model, its underlying well-founded set is the unique answer set of the program.[5]

One often criticized deficiency of the standard well-founded model is that it is too skeptical. Unfortunately, this is *not* remedied by alternating the fixpoint operators of the previous sections, no matter which strategy we consider. To see this, consider the ordered logic program $(\Pi_3, <)$:

$$\begin{array}{lllll} r_1 & = & a & \leftarrow & not\ b \\ r_2 & = & b & \leftarrow & not\ a \end{array} \qquad r_2 < r_1 \qquad (3)$$

[5]No matter whether we consider the fixpoint operators for the approach in [28], [8], or [13], respectively.

The well-founded model of Π_3 is given by $(\emptyset, \emptyset)$. The same model is obtained by alternating operator $C_{(\Pi_3,<)}$. Observe that $C_{(\Pi_3,<)}(\emptyset) = \{a, b\}$ and $C_{(\Pi_3,<)}(\{a, b\}) = \emptyset$. Consequently, $\emptyset$ is the least alternating fixpoint of $(\Pi_3, <)$.

The question is now why these operators are still too skeptical in defining well-founded semantics (although they work nicely in the setting of answer sets and regular semantics). In fact, the great advantage of a setting like that of answer sets semantics is that we deal with *direct* fixpoint equations, like $C_{(\Pi,<)}(X) = X$, where the context X represents the putative answer set. This is different in the setting of well-founded semantics, where we usually start by applying an operator to a rather small context, eg. initially the empty set; this usually results in a larger set, sometimes even Lit, that constitutes then the context of the second application of the operator. Now, looking at the underlying definitions, we see that the actual preference handling condition, eg. Condition II in Definition 1 takes advantage of X for deciding applicability. The alternating character in the well-founded setting does not support this sort of analysis since it cannot provide the (putative) final result of the computation.

5 Towards a preferred well-founded semantics

In view of the failure of the above fixpoint operator(s) in the setting of well-founded semantics, the obvious question is now whether an appropriate *alternating* fixpoint operation is definable that yields a reasonable well-founded semantics for ordered logic programs. As informal guidelines, we would like that the resulting semantics (i) allows for deriving more conclusions than the standard well-founded semantics by appeal to given preferences; (ii) coincides with standard well-founded semantics in the absence of preferences; and finally (iii) approximates the previous preferred answer sets semantics.

The standard well-founded model is defined by means of the least fixpoint of the operator $A_\Pi = C_\Pi C_\Pi$. As above, we aim at integrating preferences by elaborating upon the underlying immediate consequence operator $T_{\Pi,Y}X$ given in (2). As well, the basic idea is to modify this operator so that more conclusions can be derived by employing preferences. However, as discussed at the end of the previous section, the alternating iterations of C_Π face two complementary situations: those with smaller contexts and those with larger ones. Since preferences exploit these contexts, it seems reasonable to distinguish alternating applications or, at least, to concentrate on one such situation while dealing with the other one in the standard way.[6] For strengthening $A_\Pi = C_\Pi C_\Pi$, we thus have two options: either we make the outer operator derive more literals or we make the

[6]Such an approach is also pursued in [6].

inner operator derive less literals.

In what follows, we adopt the former option and elaborate upon the outer operator. The general idea is then to reduce the context considered in the second application of C_Π by appeal to preferences in order to make more rules applicable. For this purpose we remove those literals that are derived by means of less preferred, defeated rules.

Definition 3 *Let $(\Pi, <)$ be an ordered logic program and let X and Y be sets of literals.*

We define the set of immediate consequences of X with respect to $(\Pi, <)$ and Y as

$$\mathcal{T}^\circ_{(\Pi,<),Y}X = \{head(r) \mid r \in \Pi \text{ is active wrt } (X, Y \backslash D^r_{(X,Y)})\}$$

where

$$D^r_{(X,Y)} = \left\{ L \;\middle|\; \begin{array}{l} \text{for all rules } r' \in \Pi, \\ \quad \text{if } L = head(r') \quad \text{and} \\ \quad\quad body^+(r') \subseteq Y, \\ \quad \text{then } r' < r \quad \text{and} \\ \quad\quad (head(r) \cup X) \cap body^-(r') \neq \emptyset \end{array} \right\}$$

if X is consistent, and $\mathcal{T}^\circ_{(\Pi,<),Y}X = Lit$ otherwise.

We say that r defeats r' wrt X if $(head(r) \cup X) \cap body^-(r') \neq \emptyset$. The set of removed literals $D^r_{(X,Y)}$ consists thus of those rule heads, all of whose corresponding rules are less preferred than r and defeated by r or X, viz. the literals derived so far. In fact, this condition only removes a literal such as $head(r')$ from Y, if all of its applicable generating rules like r' are defeated by the preferred rule r. Note that $D^r_{(X,Y)}$ is normally different for different rules r.

For illustration consider the rules in Π_3. For $X = \emptyset$ and $Y = \{a, b\}$, we get $D^{r_1}_{(\emptyset,\{a,b\})} = \{b\}$ and $D^{r_2}_{(\emptyset,\{a,b\})} = \emptyset$. In such a situation, activeness of r_1 is checked wrt $(\emptyset, \{a, b\} \setminus \{b\})$ while that of r_2 is checked wrt $(\emptyset, \{a, b\})$. When applying r_1, the removal of $D^{r_1}_{(\emptyset,\{a,b\})} = \{b\}$ from context $\{a, b\}$ allows us to discard the conclusion of the less preferred rule r_2 that is defeated by the preferred rule r_1. This example is continued below.

Notably, the choice of $D^r_{(X,Y)}$ is one among many options. Unfortunately, it leads beyond the scope of this paper to investigate the overall resulting spectrum, so that we concentrate on the above definition and discuss some alternatives at the end of this section. From a general perspective, the above definition offers thus a parameterizable framework for defining well-founded semantics including preferences.

In analogy to the previous sections, we can define a consequence operator as follows.

Definition 4 *Let $(\Pi, <)$ be an ordered logic program and let X be a set of literals.*

We define $C^\circ_{(\Pi,<)}(X) = \bigcup_{i\geq 0}(\mathcal{T}^\circ_{(\Pi,<),X})^i\emptyset$.

Of particular interest in view of an alternating fixpoint theory is that $C^\circ_{(\Pi,<)}$ enjoys *anti-monotonicity*:

Theorem 1 *Let $(\Pi, <)$ be an ordered logic program and X_1, X_2 sets of literals.*

If $X_1 \subseteq X_2$, then $C^\circ_{(\Pi,<)}(X_2) \subseteq C^\circ_{(\Pi,<)}(X_1)$.

Given this, we may define a new alternating transformation of $(\Pi, <)$ as

$$\mathcal{A}^\circ_{(\Pi,<)} = C^\circ_{(\Pi,<)}C_\Pi.$$

Since both $C^\circ_{(\Pi,<)}$ and C_Π are anti-monotonic, $\mathcal{A}^\circ_{(\Pi,<)}$ is monotonic.

Definition 5 *Let $(\Pi, <)$ be an ordered logic program and let X be a set of literals.*

We define X as a preferred well-founded set of $(\Pi, <)$ iff $lfp\mathcal{A}^\circ_{(\Pi,<)} = X$.

By Tarski's Theorem [25], we get that each ordered logic program has a unique preferred well-founded set.

Theorem 2 *Let $(\Pi, <)$ be an ordered logic program.*

Then, there is a unique preferred well-founded set of $(\Pi, <)$.

Given the notion of the preferred well-founded set, we define the preferred well-founded model of an ordered program as follows.

Definition 6 *Let $(\Pi, <)$ be an ordered logic program and let X be the well-founded set of $(\Pi, <)$.*

We define the preferred well-founded model of $(\Pi, <)$ as $(X, Lit \setminus C_\Pi(X))$.

It is well-known that the standard well-founded semantics for extended logic programs has time complexity $O(n^2)$ [29, 4]. The complexity of the preferred well-founded semantics is still in polynomial time but it is in $O(n^3)$. The reason is that we have to additionally compute $D^r_{(X,Y)}$ for each $r \in \Pi$.

We first obtain the following corollary to Theorem 2.

Corollary 3 *Every ordered logic program has a unique preferred well-founded model.*

This result shows that our preferred well-founded semantics is as robust as the standard well-founded semantics.

The relationship between the standard well-founded model and the preferred well-founded model can be stated as follows.

Theorem 4 *Let (X,Y) be the preferred well-founded model of $(\Pi,<)$ and let (X',Y') be the well-founded model of Π.*

Then, we have

1. *$X' \subseteq X$ and $Y' \subseteq Y$ and*
2. *$(X,Y) = (X',Y')$, if $< = \emptyset$.*

Let us reconsider $(\Pi_3,<)$. While $(\emptyset,\emptyset)$ is the well-founded model of Π_3, its ordered counterpart $(\Pi_3,<)$ has the preferred well-founded model $(\{a\},\{b\})$. To see this, observe that $C_{\Pi_3}\emptyset = \{a,b\}$ and $C^{\circ}_{(\Pi_3,<)}(\{a,b\}) = \{a\}$. Clearly, $\{a\}$ is a fixpoint of C_{Π_3} and $C^{\circ}_{(\Pi_3,<)}$. Thus, $\{a\}$ is an alternating fixpoint of $(\Pi_3,<)$. Also, we see that $\emptyset$ is not an alternating fixpoint. This implies that $\{a\}$ is the least alternating fixpoint of $(\Pi_3,<)$.

This example along with the last result show that preferences allow us to strengthen the conclusions obtained by the standard well-founded semantics. That is, whenever certain conclusions are not sanctioned in the standard framework one may add appropriate preferences in order to obtain these conclusions within the overall framework of well-founded semantics.

For a complement, consider the following variation of $(\Pi_3,<)$, also discussed in [6].

$$\begin{array}{lcll} r_1 & = & a \leftarrow & not\ b \qquad r_2 < r_1 \\ r_2 & = & b \leftarrow & not\ c \end{array} \tag{4}$$

Observe that Π_4 has well-founded model $(\{b\},\{a,c\})$. In contrast to $(\Pi_3,<)$, the preferred well-founded model of $(\Pi_4,<)$ is also $(\{b\},\{a,c\})$. As discussed in [6] this makes sense since preferences should only enrich but not "override" an underlying well-founded model.

Another attractive property of this instance of preferred well-founded semantics is that it provides an approximation of preferred answer sets semantics.

Theorem 5 *Let (X,Y) be the preferred well-founded model of $(\Pi,<)$ and let Z be a preferred answer set of $(\Pi,<)$.*

Then, we have $X \subseteq Z$ and $Y \subseteq Lit \setminus Z$.

Notably, this can be shown for all aforementioned preferred answer sets semantics, no matter whether we consider the approach in [28], [8], or [13], respectively.

Finally, let us briefly discuss some alternative choices for $D^r_{(X,Y)}$. In fact, whenever we express the same preferences among (negative) rules having the same head the previous definition of $D^r_{(X,Y)}$ is equivalent to $\{head(r') \mid r' < r \text{ and } (head(r) \cup X) \cap body^-(r') \neq \emptyset\}$. However, this conceptually simpler definition is inadequate when it comes to attributing different preferences to rules with the same heads as in the following example.

Consider the ordered program $(\Pi_5,<)$.

$$\begin{array}{lcll} r_1 & = & a \leftarrow & \qquad r_3 < r_2 < r_1 \\ r_2 & = & b \leftarrow & not\ a \\ r_3 & = & a \leftarrow & not\ b \end{array} \tag{5}$$

The preferred well-founded semantics of $(\Pi_5,<)$ gives $(\{a\},\{b\})$, while the conceptually simpler one yields $(\{a,b\},\emptyset)$, a clearly wrong result! In the simplistic setting $D^{r_2}_{(\emptyset,\{a,b\})}$ would contain the head of the third rule, discarding the fact that r_1 already defeats r_2.

Another alternative choice for $D^r_{(X,Y)}$ is indicated by the difference between the strategies employed in [28] and [13]. In fact, the latter implicitly distinguishes between same literals stemming from different rules. This amounts to distinguishing different occurrences of literals. For this, we may rely on the aforementioned simplistic definition of $D^r_{(X,Y)}$ and suppose that $head(r)$ provides us with occurrences of literals, like b^{r_2} instead of b. Without entering details, let us illustrate this idea by appeal to $(\Pi_5,<)$. An approach distinguishing occurrences of literals would yield $C_{\Pi_5}\emptyset = \{a^{r_1},b^{r_2},a^{r_3}\}$ and $C^{\circ}_{(\Pi_5,<)}(\{a^{r_1},b^{r_2},a^{r_3}\}) = \{a^{r_1},a^{r_3}\}$. When considering r_2, we check activeness wrt $(\emptyset,\{a^{r_1},b^{r_2},a^{r_3}\}\setminus\{a^{r_3}\})$, viz. $(\emptyset,\{a^{r_1},b^{r_2}\})$. Unlike just above, a^{r_1} remains in the reduced context and r_2 is inapplicable. An elaboration of this avenue is beyond the scope of this paper, in particular, since it involves an occurrence-based development of well-founded semantics.

6 Relationships

In contrast to answer set semantics, the extension of well-founded semantics to ordered logic program has been rarely studied before. In this section we will discuss the relation of our approach to [6, 21, 30].

6.1 Relation to Brewka's Approach

Brewka defines in [6] a well-founded semantics for ordered logic programs. Notably, this approach is based on a paraconsistent extension of well-founded semantics that tolerates inconsistencies among the result of the inner operator without trivializing the overall result. Despite this deviation from standard well-founded semantics, the question remains whether Brewka's semantics can be captured within our semantical framework.

In fact, both approaches are based on quite different intuitions. While the underlying idea of Brewka's approach is to define a criterion for selecting the intended rules by employing preference, we integrate preferences into the

immediate consequence operaor by individually restricting the context of application for each rule.

Nonetheless, it turns out that Brewka's semantics can be captured through an alternating fixpoint construction. As we show below, Brewka's modification boils down to using an alternate fixpoint operator of the form "$C^{\star}_{(\Pi,<)}C^{\star}_{\Pi}$". To this end, let us first consider the difference among the underlying operators $C^{\star}_{\Pi}$ and C_{Π}. Define $Cl(\Pi)$ as the smallest set of literals which is closed under a basic program Π. Then, given a set X of literals, $C^{\star}_{\Pi}(X)$ is defined as $Cl(\Pi^X)$. Dropping the requirement of logical closure results in a paraconsistent inference operation. For example, given $\Pi = \{a \leftarrow, \neg a \leftarrow, b \leftarrow\}$, we get $Cn(\Pi) = Lit$, while $Cl(\Pi) = \{a, \neg a, b\}$. Although the corresponding adaptions are more involved, the surprising result is now that Brewka's semantics can also be captured within our overall framework, if we use the closure operator Cl instead of Cn.

Moreover, we need the following. Let $(\Pi, <)$ be an ordered logic program and X be a set of literals. We define Π^r_X as the set of rules defeated by r wrt X and $<$ as

$$\Pi^r_X = \{r' \in \Pi \mid r' < r,\ r \text{ defeats } r' \text{ wrt } X\}.$$

Notice that Π^r_X is a set of rules while $D^r_{(X,Y)}$ is a set of literals. Π^r_X is also different from Brewka's *Dom* (set of *dom*inated rules) in that Π^r_X is defined wrt a set X of literals rather than a set of rules.

Write $(\Pi^r_X)^+ = \{(r')^+ \mid r' \in \Pi^r_X\}$. Let $\mathcal{T}^{\star}_{(\Pi,<)}$ be the operator obtained from $\mathcal{T}^{\circ}_{(\Pi,<)}$ (in Definition 3) by replacing $Y \setminus D^r_{(X,Y)}$ with $Cl(\Pi^Y \setminus (\Pi^r_X)^+)$. This results in a fixpoint operator $\mathcal{C}^{\star}_{(\Pi,<)}$.

As we show in the full version of this paper, Brewka's well-founded set corresponds to the least fixpoint of the alternating operator $\mathcal{C}^{\star}_{(\Pi,<)}C^{\star}_{\Pi}$. This means Brewka's well-founded semantics also enjoys an alternating fixpoint characterization.

6.2 Relation to Other Approaches

In [30], it is mentioned that a well-founded semantics with preference can be defined in terms of their operator but default negation is not allowed in their syntax. However, even for ordered logic programs without default negation, our basic semantic approach is different from the well-founded semantics in priority logic [30]. The main reason is that they interpret the priority relation $r < r'$ in a quite different way: r is blocked whenever r' is applicable. While we attribute to the program

$$\begin{array}{lcll} r_1 & = & p & \leftarrow \\ r_2 & = & q & \leftarrow \end{array} \qquad r_2 < r_3 \qquad (6)$$

a preferred well-founded model, containing both p and q, the well-founded model of Π_6 in priority logic is $\{p\}$. That is, q cannot be inferred.

Another skeptical semantics for preference is defeasible logic, which was originally introduced by D. Nute [21] and received extensive studies in recent years [1, 2, 20]. Defeasible logic distinguishes the strict rules from defeasible rules. This already makes its semantics different from our preferred well-founded semantics.

Consider an example from [7]. The following is a theory in defeasible logic:

$$\begin{array}{lcll} r'_1 & & \Rightarrow & p \\ r'_2 & p & \rightarrow & q \\ r'_3 & & \Rightarrow & \neg q \end{array} \qquad r'_2 < r'_3 \qquad (7)$$

In defeasible logic, $+\delta q$ is not derivable, i. e., q cannot be defeasibly derived. As pointed out by Brewka, this means a defeasible rule having higher priority can defeat a strict rule.

The above theory can be directly translated into an ordered logic program $(\Pi, <)$ as follows:

$$\begin{array}{lcll} r_1 & = & p & \leftarrow \ not\ \neg p \\ r_2 & = & q & \leftarrow \ p \\ r_3 & = & \neg q & \leftarrow \ not\ q \end{array} \qquad r_2 < r_3 \qquad (8)$$

It can be verified that the preferred well-founded model (in our sense) is $\{p, q\}$. Therefore, q is derivable under our preferred well-founded semantics.

7 Conclusion

We have looked into the issue of how van Gelder's alternating fixpoint theory [27] for normal logic programs can be suitably extended to define the well-founded semantics for ordered logic programs (extended logic programs with preference). The key of the alternating fixpoint approach is how to specify a suitable consequence relation for ordered logic programs. We argue that the preference strategies for defining answer sets are not suitable for defining preferred well-founded semantics and then some informal criteria for preferred well-founded semantics are proposed. Based on this analysis, we have defined a well-founded semantics for ordered logic programs. This semantics allows an elegant definition and satisfies some attractive properties: (1) Each ordered logic program has a unique preferred well-founded model; (2) The preferred well-founded reasoning is no less skeptical than the standard well-founded reasoning; (3) Any conclusion under the preferred well-founded semantics is also derivable under some major preferred an-

swer sets semantics. Our semantics is different from defeasible logic and the skeptical priority logic. An important result is the equivalence of Brewka's preferred well-founded semantics and our semantics introduced in Section 5.

Acknowledgments. This work was supported by DFG under grant FOR 375/1-1, TP C.

References

[1] G. Antoniou, D. Billington, G. Governatori, and M. Maher. A flexible framework for defeasible logics. In *Proc. AAAI/IAAI'2000*, pages 405–410. AAAI Press, 2000.

[2] G. Antoniou, D. Billington, G. Governatori, and M. Maher. Representation results for defeasible logic. *ACM Transactions on Computational Logic*, 2(2):255–287, 2001.

[3] F. Baader and B. Hollunder. How to prefer more specific defaults in terminological default logic. In R. Bajcsy, editor, *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 669–674. Morgan Kaufmann Publishers, 1993.

[4] C. Baral and M. Gelfond. Logic programming and knowledge representation. *Journal of Logic Programming*, 12:1–80, 1994.

[5] G. Brewka. Adding priorities and specificity to default logic. In L. Pereira and D. Pearce, editors, *European Workshop on Logics in Artificial Intelligence (JELIA'94)*, Lecture Notes in Artificial Intelligence, pages 247–260. Springer-Verlag, 1994.

[6] G. Brewka. Well-founded semantics for extended logic programs with dynamic preferences. *Journal of Artificial Intelligence Research*, 4:19–36, 1996.

[7] G. Brewka. On the relationship between defeasible logic and well-founded semantics. In T. Eiter, W. Faber, and M. Truszczynski, editors, *Proceedings of the Sixth International Conference on the Logic Programming and Nonmonotonic Reasoning*, pages 121–132. Springer-Verlag, 2001.

[8] G. Brewka and T. Eiter. Preferred answer sets for extended logic programs. *Artificial Intelligence*, 109(1-2):297–356, 1999.

[9] G. Brewka and T. Eiter. Prioritizing default logic. In St. Hölldobler, editor, *Intellectics and Computational Logic — Papers in Honour of Wolfgang Bibel*, pages 27–45. Kluwer Academic Publishers, 2000.

[10] F. Buccafurri, W. Faber, and N. Leone. Disjunctive logic programs with inheritance. In *Proceedings of the International Conference on Logic Programming*, pages 79–93. The MIT Press, 1999.

[11] B. Cui and T. Swift. Preference logic grammars: Fixed-point semantics and application to data standardization. *Artificial Intelligence*, 2001. To appear.

[12] J. Delgrande and T. Schaub. Compiling reasoning with and about preferences into default logic. In M. Pollack, editor, *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 168–174. Morgan Kaufmann Publishers, 1997.

[13] J. Delgrande, T. Schaub, and H. Tompits. Logic programs with compiled preferences. In W. Horn, editor, *Proceedings of the European Conference on Artificial Intelligence*, pages 392–398. IOS Press, 2000.

[14] T. Eiter, M. Fink, G. Sabbatini, and H. Tompits. A generic approach for knowledge-based information-site selection. In *Proceedings of the Eighth International Conference on the Principles of Knowledge Representation and Reasoning*. Morgan Kaufmann Publishers, 2002.

[15] M. Gelfond and T. Son. Reasoning with prioritized defaults. In J. Dix, L. Pereira, and T. Przymusinski, editors, *Third International Workshop on Logic Programming and Knowledge Representation*, volume 1471 of *Lecture Notes in Computer Science*, pages 164–223. Springer-Verlag, 1997.

[16] T. Gordon. *The pleading game: An Artificial Intelligence Model of Procedural Justice*. Dissertation, Technische Hochschule Darmstadt, Alexanderstraße 10, D-64283 Darmstadt, Germany, 1993.

[17] B. Grosof. Prioritized conflict handling for logic programs. In J. Maluszynsk, editor, *Logic Programming: Proceedings of the 1997 International Symposium*, pages 197–211. The MIT Press, 1997.

[18] B. Grosof. Business rules for electronic commerce. `http://www.research.ibm.com/rules/papers`. 1999. IBM Research.

[19] J. Lloyd. *Foundations of Logic Programming*. Symbolic Computation. Springer-Verlag, 2nd edition, 1987.

[20] M. Maher, J. Rock, G. Antoniou, D. Billington, and D. Miller. Efficient defeasible reasoning systems. In *Proceedings of the 12th International Conference on Tools with Artificial Intelligence*, pages 384–392. IEEE Press, 2000.

[21] D. Nute. Defeasible reasoning. In *Proceedings of the 20th Hawaii International Conference on Systems Science*, pages 470–477. IEEE Press, 1987.

[22] H. Prakken. *Logical Tools for Modelling Legal Argument.* Kluwer Academic Publishers, 1997.

[23] C. Sakama and K. Inoue. Representing priorities in logic programs. In M. Maher, editor, *Proceedings of the 1996 Joint International Conference and Symposium on Logic Programming*, pages 82–96, Cambridge, 1996. The MIT Press.

[24] T. Schaub and K. Wang. A comparative study of logic programs with preference. In B. Nebel, editor, *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 597–602. Morgan Kaufmann Publishers, 2001.

[25] A. Tarski. A lattice-theoretic fixpoint theorem and its applications. *Pacific Journal of Mathematics*, 5:285–309, 1955.

[26] M. H. van Emden and R. A. Kowalski. The semantics of predicate logic as a programming language. *Journal of the ACM*, 23(4):733–742, 1976.

[27] A. van Gelder. The alternating fixpoint of logic programs with negation. *Journal of Computer and System Science*, 47:185–120, 1993.

[28] K. Wang, L. Zhou, and F. Lin. Alternating fixpoint theory for logic programs with priority. In *Proceedings of the First International Conference on Computational Logic*, volume 1861 of *Lecture Notes in Computer Science*, pages 164–178. Springer-Verlag, 2000.

[29] C. Witteveen. Partial semantics for truth maintenance. In J. van Eijck, editor, *Logics in AI*, volume 478 of *Lecture Notes in Artificial Intelligence*, pages 544–561. Springer-Verlag, 1991.

[30] J. You, X. Wang, and L. Yuan. Nonmonotonic reasoning as prioritized argumentation. *IEEE Transactions on Knowledge and Data Engineering*, 2001. To appear.

[31] Y. Zhang and N. Foo. Answer sets for prioritized logic programs. In J. Maluszynski, editor, *Proceedings of the International Symposium on Logic Programming (ILPS-97)*, pages 69–84. The MIT Press, 1997.

Answer Sets Generated by Selector Functions

Sibylle Schwarz
Institut für Informatik, Universität Leipzig
Leipzig, Germany, Augustusplatz 10–11

Abstract

This paper presents a framework to define various declarative semantics of generalized logic programs (programs with nested expressions). Each of these semantics contains all models of a generalized logic program that can be reached from the least interpretation by a monotonic chain of interpretations. These chains are parameterized by selector functions. We identify selector functions that generate well-known classes of preferred models like minimal, stable, stable generated and least Herbrand models.
An interesting example of a selector-generated semantics is the end-regulated semantics. We study its connections to other semantics and the effect of some program transformations.

1 Motivation

Generalized logic programs are sets of rules that contain arbitrary quantifier free first order formulas in both their bodies and their heads. Definite, normal and disjunctive logic programs form proper subclasses of this program class.

The meaning of generalized logic programs is given by a declarative semantics. The program represents a set of logic formulas. One way to define a declarative semantics is to select a subset of the set of all models of these formulas.

The least Herbrand model of a definite program represents the intended meaning of a definite program in a unique and intuitive way. Things are not as easy if composed formulas are allowed in the program rules. Stable model semantics (Gelfond and Lifschitz, 1988) and, more generally, answer set semantics define the properties a model has to satisfy to be an intended model of a program.

In most approaches, answer sets are minimal models of a transformed program. Almost all generalizations of answer sets are based on stable models presented by Gelfond and Lifschitz in (Gelfond and Lifschitz, 1988). Essentially, for each extension to programs with more expressive syntax, an appropriate program transformation has to be found (Przymusinski, 1994), (Brass et al., 1998), (Sakama and Inoue, 1995). In this way, Lifschitz, Tang and Turner defined in (Lifschitz et al., 1999) stable models of generalized logic programs (programs with nested expressions).

This paper presents an alternative approach to extend the stable model semantics to generalized logic programs. This extension does not depend on program transformations. It is a generalization of stable generated models, defined by Herre and Wagner in (Herre and Wagner, 1997b). We select intended models by connecting them to zero-truth by a chain of interpretations.

In Section 2, some notations are recalled. The basic idea in our approach is the reachability of models by special chains of interpretations. These chains are defined in Section 3. In Section 4 we study some selector functions and their influence to models they generate. End-regulated models are an interesting example of selector generated models, that has been hardly examined so far. In Section 5 and Section 6 we study their properties and the effect of some program transformations.

2 Background

We assume that the reader is familiar with the standard notions of propositional and first order logic, such as atom, literal, formula, conjunctive and disjunctive normal form, Herbrand base B_P for a program P and

the usual model relation $\models$ between formulas and two-valued interpretations. We use the connectives $\mathtt{not}, \vee$ and $\wedge$.

Our framework defines the semantics of a logic program in terms of its ground instantiation. As usual in answer set semantics, we can restrict our attention to propositional programs. Atoms correspond to propositional variables and the base of a program is the set of all propositional variables that correspond to atoms occurring in the ground instantiation of the program.

Rules r in generalized logic programs are pairs of propositional formulas $\mathtt{body}(r)$ and $\mathtt{head}(r)$.
We denote rules by $\mathtt{body}(r) \Rightarrow \mathtt{head}(r)$. Note that the meaning of the rule arrow $\Rightarrow$ is different from classical implication. A *generalized logic program* is a set of rules.

Program classes with restricted rule syntax are

definite logic programs
$\mathtt{body}(r) = \bigwedge_{i\in\{1,\dots,n\}} a_i$
all a_i and $\mathtt{head}(r)$ are atoms

normal logic programs
$\mathtt{body}(r) = \bigwedge_{i\in\{1,\dots,n\}} a_i \wedge \bigwedge_{j\in\{1,\dots,m\}} \mathtt{not}\ b_j$
all a_i, b_j and $\mathtt{head}(r)$ are atoms
We will use the notation
$\mathtt{body}^+(r) := \{a_i \mid i \in \{1,\dots,n\}\}$ and
$\mathtt{body}^-(r) := \{b_j \mid j \in \{1,\dots,m\}\}$

disjunctive logic programs
$\mathtt{head}(r) = \bigvee_{i\in\{1,\dots,k\}} c_i \vee \bigvee_{j\in\{1,\dots,l\}} \mathtt{not}\ d_j$
all c_i, d_j are atoms,
bodies as in normal logic programs

The sets $\mathtt{head}(P)$ and $\mathtt{body}(P)$ contain all heads and bodies of rules in P, respectively.

An *interpretation* $I : B_P \to W$ of a program is a total function from the base of the program to a set of truth values W.

Any ordering on the set of truth values induces an ordering on interpretations by pointwise comparison.

In this paper, two-valued interpretations ($W = \{\mathbf{t}, \mathbf{f}\}$ with the ordering $\mathbf{f} < \mathbf{t}$) are used. If convenient (mostly in examples), we identify an interpretation with its pre-image of $\mathbf{t}$. The set of all two-valued interpretations of the program P is denoted by $\mathbf{I}_P$.

Note that the framework is not restricted to this special case. The definitions given here are general enough to allow any complete lattice for W.

The interpretation $I_\mathbf{f}$ that maps each atom to $\mathbf{f}$ is the *least* interpretation. Given a set $\mathbf{I}$ of interpretations, the set of all its minimal elements is $\mathrm{Min}(\mathbf{I})$ and the least upper bound of $\mathbf{I}$ is denoted by $\sup \mathbf{I}$.

For *intervals* of interpretations we write

$$[I, J] := \{K \mid I \leq K \leq J\}$$

The set $\mathrm{Mod}(\Phi)$ of interpretations contains exactly all models of a set of formulas Φ. An interpretation I is a *model of a program rule* r (written $I \models r$), if it is a model of the formula $\mathtt{not}\ \mathtt{body}(r) \vee \mathtt{head}(r)$ where $\mathtt{not}$ is interpreted as classical negation. An interpretation is a *model of a logic program* P, if it is a model of all rules in P. A set of interpretations $\mathbf{I}$ satisfies a formula φ, if each interpretation in $\mathbf{I}$ satisfies φ (written $\mathbf{I} \models \varphi$).

A set $\mathbf{I}$ of interpretations selects the set of rules from a program P, whose bodies are satisfied in each interpretation from $\mathbf{I}$.

Definition 1

$$P_\mathbf{I} := \{r \in P \mid \mathbf{I} \models \mathtt{body}(r)\}$$

The *immediate consequence operator* of a definite program P is $T_P : \mathbf{I}_P \to \mathbf{I}_P$ where
$T_P(I) = \{a \mid \exists r \in P : \mathtt{head}(r) = a \text{ and } I \models \mathtt{body}(r)\}$.

The *Gelfond-Lifschitz transformation* defined in (Gelfond and Lifschitz, 1988) depends on an interpretation M and maps a normal program P to its definite *reduct*

$$P^M := \{ \bigwedge \mathtt{body}^+(r) \Rightarrow a \mid \exists r \in P : \mathtt{head}(r) = a \text{ and } \mathtt{body}^-(r) \cap M = \emptyset\}$$

Stable models of a normal logic program P are the fixed points M of the operator T_{P^M} (Gelfond and Lifschitz, 1988).

3 Models Generated by Chains

In general, not every model of a logic program represents the intended meaning of this program. A subset of intuitively appropriate models has to be selected. These models should be

grounded (can be created bottom-up from zero truth by iterated application of program rules) and

supported (any atom in the model is justified by a program rule whose body is satisfied in this model).

In other generalizations of stable models to languages more expressive than normal programs, *minimality*

is considered an important property of an intended model. We do *not* require minimality, because this not always seems to be intuitive.

Example 1. Someone is entering a cafeteria to order a glass of water or an espresso. Espresso comes with a glass of water. This situation can be represented by the program $P = \{\Rightarrow e \vee w, e \Rightarrow w\}$. P has the models $M_1 = \{w\}$ and $M_2 = \{e, w\}$. Both of them are intended, as the choice between espresso or water should not depend on the way they are served, and the decision for espresso does not mean a decision against water. But $M_1 = \{e, w\}$ is not a minimal model of P. In minimal model semantics it is impossible to have an espresso in this situation.

To guarantee groundedness of the selected models, we require that they can be reached from the least interpretation by iterative application of a successor function.

Depending on the program P, a guessed interpretation M, and a selector function $\mathtt{sel} : \mathbf{I}_P \to 2^{\mathbf{I}_P}$, we define chains of interpretations.

Definition 2 ($(\mathtt{sel}, P, M)$-chain) *A $(\mathtt{sel}, P, M)$-chain is a sequence $(I_\alpha)_{\alpha \leq \kappa} \subseteq [I_\mathbf{f}, M]$ of Herbrand interpretations of the program P satisfying*

1. *$I_0 = I_\mathbf{f}$ and $I_\kappa = M$,*
2. *for successor ordinals α : $I_\alpha \in S^P_{\mathtt{sel}}(I_{\alpha-1})$,*
3. *for limit ordinals α : $I_\alpha = \sup\{I_\beta \mid \beta < \alpha\}$.*

Interpretations in a chain are connected by the successor function S, that works like a non-deterministic consequence operator.

Definition 3 (Successor function $S^P_{\mathtt{sel}}$)

$S^P_{\mathtt{sel}} : {\mathbf{I}_P}^2 \to 2^{\mathbf{I}_P}$

$S^P_{\mathtt{sel}}(M, I) := \mathrm{Min}\left(\mathrm{Mod}\left(\mathtt{head}\left(P_{\mathtt{sel}(M,I)}\right)\right) \cap [I, M]\right)$

By Definition 1 in the previous section, $P_{\mathtt{sel}(M,I)}$ is the set of all rules from P whose bodies are satisfied in each interpretaion in $\mathtt{sel}(M, I)$.

Remark. Because of the minimality of the successors we know that if an interpretation is a stable successor of itself ($I \in S^P_{\mathtt{sel}}(M, I)$) then it is the only successor ($S^P_{\mathtt{sel}}(M, I) = \{I\}$). Interpretations I with this property are called *fixed points* of $S^P_{\mathtt{sel}}(M, \bullet)$.

In the next section, we will study the influence of several selector functions on the sets of models they generate. Some prominent sets of intended models are special cases of the following definition.

Definition 4 (sel-generated interpretation) *We call an interpretation $M \in \mathbf{I}_P$ a* sel-generated *interpretation of the program P if there is an $(S^P_{\mathtt{sel}}, M)$-chain and M is a fixed point of $S^P_{\mathtt{sel}}(M, \bullet)$.*

Note that a sel-generated interpretation of P is not necessarily a model of P. To this end, sel has to satisfy an additional condition.

Lemma 1 *If $\mathtt{sel}(M, M) = \{M\}$ and M is a fixed point of $S^P_{\mathtt{sel}}(M, \bullet)$ then M is a model of P.*

Proof. $\{M\} = S^P_{\mathtt{sel}}(M, M)$. By Definition 3 this is $\{M\} = \mathrm{Min}\left(\mathrm{Mod}\left(\mathtt{head}\left(P_{\mathtt{sel}(M,M)}\right)\right) \cap [M, M]\right)$.
Hence $M \in \mathrm{Mod}\left(\mathtt{head}\left(P_{\mathtt{sel}(M,M)}\right)\right)$.
By assumption $\mathtt{sel}(M, M) = \{M\}$ this is
$M \in \mathrm{Mod}\left(\mathtt{head}\{r \in P \mid M \models \mathtt{body}(r)\}\right)$ and therefore $M \models P$. □

Definition 5 (sel-generated semantics) *The* sel*-generated semantics of a generalized logic program is the set of all its* sel*-generated models.*

4 Some Well-known Answer Sets and their Generating Selector Functions

In this section we analyze some selector functions. We will recognize some prominent model types that are generated by these functions.

The first candidate selects just the guessed interpretation.

Definition 6

$$\mathtt{sel_u}(M, I) := \{M\}$$

The corresponding successor function applies the same set of rules in each step.

Lemma 2 *Any minimal model of a generalized program is* $\mathtt{sel_u}$*-generated.*

Proof. We show, if M is a minimal model of P then it is $\mathtt{sel_u}$-generated. M is a model of P iff
$M \in \mathrm{Mod}(\mathtt{head}(\{r \in P \mid M \models \mathtt{body}(r)\}))$ or, by definition of $P_{\mathtt{sel_u}(M,M)}$, $M \in \mathrm{Mod}(\mathtt{head}(P_{\mathtt{sel_u}(M,M)}))$.
Then $\{M\} = \mathrm{Min}\left(\mathrm{Mod}(\mathtt{head}(P_{\mathtt{sel_u}}(M, M))) \cap [M, M]\right)$ and M is a fixed point of $S^P_{\mathtt{sel}}(M, \bullet)$.
As $\mathtt{sel_u}$ ddoes not depend on the second argument $\mathtt{sel_u}(M, M) = \mathtt{sel_u}(M, I_\mathbf{f})$
As M is a minimal model of P, M is the only model of P in $[I_\mathbf{f}, M]$. Therefore
$\{M\} = \mathrm{Min}\left(\mathrm{Mod}(\mathtt{head}(P_{\mathtt{sel_u}}(M, I_\mathbf{f}))) \cap [I_\mathbf{f}, M]\right)$. By Definition 3 this is $\{M\} = S^P_{\mathtt{sel_u}}(M, I_\mathbf{f})$

Thus M is a $\mathtt{sel_u}$-successor of I_f and the chain (I_f, M) proves that M is $\mathtt{sel_u}$-generated. □

Note that not every $\mathtt{sel_u}$-generated model is a minimal model, as the following example shows.

Example 2. The model $M = \{a, c\}$ of program $P = \{\Rightarrow\ a \vee b, c \Rightarrow c\}$ is $\mathtt{sel_u}$-generated via the chain $(\emptyset, M)$. Both rules can fire in M, $P_{\mathtt{sel_u}}(M, \emptyset) = P_{\mathtt{sel_u}}(M, M) = P$ and M is a minimal extension of $\emptyset$ that satisfies both heads $a \vee b$ and c. However M is not a minimal model of P, as $\{a\} \subset M$ is a model of P.

Another function selects in each step the current interpretation.

Definition 7

$$\mathtt{sel_l}(M, I) := \{I\}$$

With this selector function the successor function works like the usual immediate consequence operator.

Lemma 3 *The only $\mathtt{sel_l}$-generated interpretation of a definite program P is the least Herbrand model of P.*

Proof. For all n, $T_P(I_n) = \{a \mid \exists r \in [P] : \mathtt{head}(r) = a$ and $I_n \models \mathtt{body}(r)\}$ is the unique extension that satisfies all rules whose bodies are true in I_n. Each $\mathtt{sel_l}$-chain starts in $I_0 = \emptyset$. Hence $T_P^n(\emptyset)$ and $(I_n)_{n \leq \omega}$ are the same sequences and have the same fixed points. □

The next selector function allows to apply all rules that are satisfied in both the current interpretation I and the guessed interpretation M.

Definition 8

$$\mathtt{sel_{lu}}(M, I) := \{I, M\}$$

In some sense, this selector function mimics the Gelfond-Lifschitz transformation.

Lemma 4 *The set of all $\mathtt{sel_{lu}}$-generated interpretations of a normal logic program P contains exactly all stable models of P.*

Proof. Let M be $\mathtt{sel_{lu}}$-generated by the chain $(I_n)_{n \leq \omega}$. We show that for all n, $I_{n+1} = T_{P^M}(I_n)$.
For $I \subseteq M$ it can be seen easily that $\{I, M\} \models \mathtt{body}(r)$ iff $\mathtt{body}^+(r) \subseteq I$ and $M \cap \mathtt{body}^-(r) = \emptyset$.
The unique minimal extension of I_n is $I_{n+1} = \{a \mid \exists r \in [P] : \mathtt{head}(r) = a, \mathtt{body}^+(r) \subseteq I_n$ and $\mathtt{body}^-(r) \cap M = \emptyset\}$. Each rule $\mathtt{body}(r) \Rightarrow a$ in $[P]$, with $\mathtt{body}^-(r) \cap M = \emptyset$ has a counterpart $\bigwedge \mathtt{body}^+(r) \Rightarrow a$ in the reduct P^M.
Thus $I_{n+1} = \{a \mid \exists r \in P^M : \mathtt{head}(r) = a, \mathtt{body}^+(r) \subseteq I_n\} = T_{P^M}(I_n)$. Therefore T_{P^M} and $(I_n)_{n \leq \omega}$ have the same fixed point M. □

The interval selector function requires the body of a firing rule to be satisfied in the whole interval between the current and the guessed interpretation.

Definition 9

$$\mathtt{sel_i}(M, I) := [I, M]$$

The definition of stable generated models by Herre and Wagner (Herre and Wagner, 1997b) is just another formulation of the condition to be a $\mathtt{sel_i}$-generated interpretation.

In (Herre and Wagner, 1997b) it is shown that for normal logic programs the sets of stable models and stable generated models coincide. Therefore $\mathtt{sel_i}$ also generates stable models of normal and least Herbrand models of definite programs.

It is a strong condition on a rule that its body must be satisfied in the whole interval from the current to the guessed interpretation.

Example 3. The model $M = \{a, b\}$ of the program $P = \{a \vee \mathtt{not}\ b \Rightarrow a, a \Rightarrow b\}$ is not stable generated. The first rule cannot be applied in $\emptyset$, because its body is not satisfied in $\{b\} \in [\emptyset, M]$.

If a selector function $\mathtt{sel}$ would allow the first rule to fire in $\emptyset$, there is the $(S^P_{\mathtt{sel}}, M)$-chain $(\emptyset, \{a\}, \{a, b\})$. All its elements satisfy the body of the first rule.

This leads to the definition of the chain selector function. It has the complete chain for an additional parameter.

Definition 10

$$\mathtt{sel_c}(M, I, (I_n)_{n \leq \omega}) := [I, M] \cap \{I_n \mid n \leq \omega\}$$

Unlike in $\mathtt{sel_i}$, the body of a firing rule has to be persistent only with respect to the *chain* that generates the model.

Remark. Chains defined by one of the selector functions above reach their fixed points after at most ω steps. Hence the length of each chain can be restricted to ω without losing information. This is a straightforward generalization of Claim 1 in (Herre and Wagner, 1997b).

5 End-regulated Models

The selector function $\mathtt{sel_c}$ in Definition 10 generates an interesting class of models. In (Herre and Wagner,

1997a), $\mathtt{sel_c}$-generated models are called *end-regulated* but are not studied further.

Definition 11 *An* end-regulated model *of a generalized program P is a* $\mathtt{sel_c}$*-generated interpretation of P. The* end-regulated semantics *maps a generalized program to the set of all its end-regulated models.*

Like stable generated models, end-regulated models are a conservative extension of stable models.

Theorem 1 *The set of end-regulated models of a normal logic program P coincides with the set of all stable models of P.*

Proof. If a conjunction of literals is satisfied in both the upper and lower bound of an interval, then it is satisfied in each interpretation of this interval. As all bodies in a normal logic program are conjunctions of literals, in each step $P_{\mathtt{sel_{lu}}}$, $P_{\mathtt{sel_i}}$, $P_{\mathtt{sel_c}}$ (parameterized with the correct chain) contain exactly the same rules from a ground normal program P. Thus $\mathtt{sel_{lu}}$, $\mathtt{sel_i}$ and $\mathtt{sel_c}$ generate the same set of models.
From Lemma 4 we know that all stable models of normal programs are $\mathtt{sel_{lu}}$-generated, hence they are $\mathtt{sel_c}$-generated. □

The same argument proves that end-regulated and stable generated models coincide for any program that contains only rules whose bodies are conjunctions of literals. The heads may be arbitrarily complex. Disjunctive programs satisfy this syntactic restriction. Claim 10 in (Herre and Wagner, 1997b) says that stable models of disjunctive programs are stable generated. Both facts together prove the following theorem.

Theorem 2 *Each stable model of a disjunctive program P is an end-regulated model of P.* □

Example 1 (espresso or water) from Section 3 shows that the converse does not hold. The model $M_2 = \{e, w\}$ is an end-regulated model of the program $P = \{\Rightarrow e \vee w, e \Rightarrow w\}$ via the chain $(\emptyset, \{e\}, M)$, but not a minimal model of the reduct $P^M = P$.

There are interesting differences between stable generated and end-regulated models. A model can be end-regulated but not stable generated for two reasons.

The first case is illustrated by Example 3 from the previous section. The model $\{a, b\}$ is end-regulated via the chain $(\emptyset, \{a\}, \{a, b\})$, but not stable generated. The rule $a \vee \mathtt{not}\ b \Rightarrow a$ cannot fire with respect to $\mathtt{sel_i}$ in $\emptyset$, because there is an interpretation $\{b\}$ *besides* the chain, that does not satisfy the body.

The second example utilizes the ability of chains to jump.

Example 4. $P = \{(\mathtt{not}\ a \vee b) \wedge (a \vee \mathtt{not}\ b) \Rightarrow a, \Rightarrow b\}$.
$M = \{a, b\}$ is end-regulated via the chain $(\emptyset, M)$ as the bodies of both rules are true in $\emptyset$ and in M.
The first rule cannot fire with respect to $\mathtt{sel_i}$ in $\emptyset$, because its body is false in $\{a\}$ and $\{b\}$.
Hence M is not stable generated.

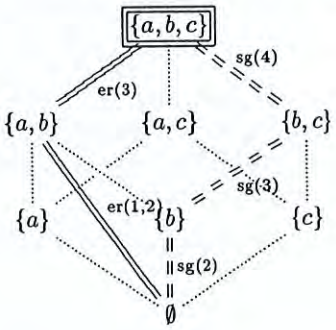
An interpretation J can be a $\mathtt{sel_c}$-successor of another interpretation I even if all interpretations in $[I, J]$ different from I and J do *not* satisfy the bodies of rules that are fired in I. As $\mathtt{sel_c}$ is only interested in the truth of the bodies on the chain, it is not affected by these interpretations. In this case, M is not stable generated, because interpretations *between* two links of the chain do not satisfy the body.

Even if a model of a generalized program is end-regulated and stable generated, the chains proving these properties need not be the same.

Example 5. The model $M = \{a, b, c\}$ is a stable generated and end-regulated model of

$$P = \{(\mathtt{not}\ a \vee b) \wedge (a \vee \mathtt{not}\ b) \Rightarrow a \ \ (1),$$
$$\Rightarrow b \ \ (2), b \Rightarrow c \ \ (3), c \Rightarrow a \ \ (4)\ \}.$$

But the chains that prove these properties differ.
The unique $S^P_{\mathtt{sel_i}}$-chain $(\emptyset, \{b\}, \{b, c\}, \{a, b, c\})$ proves that M is stable generated, and $(\emptyset, \{a, b\}, \{a, b, c\})$ is the unique $S^P_{\mathtt{sel_c}}$- chain proving M to be end-regulated. The following picture shows both chains in the interval $[\emptyset, M]$ with numbers of the applied rules.



With respect to the end-regulated semantics, rules of the form
$\varphi \vee \psi \rightarrow \psi$ can express that ψ is true in the intended model whenever φ becomes true at least once in the iterative construction of this model. The model itself need not satisfy φ. Such rules provide a nice way to express that once a formula ψ had been proved, it is kept in mind, but the proof itself is no longer interesting. Stable and stable generated model semantics do not provide this possibility as they do not always allow to apply the rule $\varphi \vee \psi \rightarrow \psi$ if φ is not true in the model itself.

6 Program Transformations

Two programs are said to be *end-regulated equivalent* (ER-equivalent) if they have the same set of end-regulated models.

The definition of end-regulated models does not depend on the inner structure of the bodies and heads of the program rules. Hence it is clear that any logically equivalent transformation of formulas can be applied separately to the body or the head of a rule. For instance, turning all bodies in disjunctive and all heads in conjunctive normal form does not change the end-regulated semantics of the program.

Splitting a rule with a conjunctive head leaves the end-regulated semantics unchanged.

Lemma 5 *The programs* $P \cup \{\varphi \Rightarrow \psi_1 \wedge \psi_2\}$ *and* $P \cup \{\varphi \Rightarrow \psi_1, \varphi \Rightarrow \psi_2\}$ *are end-regulated equivalent.*

Proof. We set $P_1 = P \cup \{\varphi \Rightarrow \psi_1 \wedge \psi_2\}$ and $P_2 = P \cup \{\varphi \Rightarrow \psi_1, \varphi \Rightarrow \psi_2\}$.
The interpretation M is an end-regulated model of P_1 if there is an $S^{P_1}_{\mathtt{sel_c}}$-chain $(I_n)_{n\leq\omega}$ for M. We show that this chain is also an $S^{P_2}_{\mathtt{sel_c}}$-chain for M.
If $I_n \models \varphi$, then I_{n+1} is a minimal extension of I_n that satisfies $\psi_1 \wedge \psi_2$ and a set F of heads from P. $F \cup \{\psi_1 \wedge \psi_2\}$ and $F \cup \{\psi_1, \psi_2\}$ have the same set of models and therefore the same set of minimal extensions of I_n. Hence each $(I_n)_{n\leq k}$ is an initial part of an $S^{P_2}_{\mathtt{sel_c}}$-chain for M. Thus $(I_n)_{n\leq\omega}$ is an $S^{P_2}_{\mathtt{sel_c}}$-chain for M.

The other direction can be proved analogously. □

Because any rule is equivalent to a rule with a head in conjunctive normal form, a generalized program can be transformed ER-equivalently to a program whose heads contain no conjunction.

Lifschitz, Tang and Turner proved in (Lifschitz et al., 1999) that the programs
$P \cup \{\varphi_1 \vee \varphi_2 \Rightarrow \psi\}$ and $P \cup \{\varphi_1 \Rightarrow \psi, \varphi_2 \Rightarrow \psi\}$ are equivalent with respect to their definition of stable models. Therefore in this sense every generalized logic program is equivalent to a disjunctive logic program.

Stable generated and end-regulated models are not invariant with respect to this transformation. This indicates that these semantics are more expressive than that from (Lifschitz et al., 1999).

Two examples illustrate the differences.

Example 6. Program $P = \{b \vee \mathtt{not}\ b \Rightarrow b\}$ has the stable generated and end-regulated model $\{b\}$. Program $Q = \{b \Rightarrow b, \mathtt{not}\ b \Rightarrow b\}$ has neither stable generated nor end-regulated models, because no body is satisfied in both interpretations $\emptyset$ and $\{b\}$.

In some cases, the end-regulated semantics is even less tolerant than the stable generated semantics. End-regulated models may differ on two programs even if the sets of stable generated models coincide.

Example 7. The interpretation $M = \{a, b\}$ is a model of both programs
$P = \{a \vee \mathtt{not}\ b \Rightarrow a, a \Rightarrow b\}$ and
$Q = \{a \Rightarrow a, \mathtt{not}\ b \Rightarrow a, a \Rightarrow b\}$.
Neither P nor Q have stable generated models, because no body is satisfied in all interpretations of $[\emptyset, \{a, b\}]$. But M is an end-regulated model of P via the chain $(\emptyset, \{a\}, \{a, b\})$ and M is not an end-regulated model of Q.

7 Further research

The presented framework is a starting point to many interesting investigations. The syntax of the rules in generalized programs can be extended to even more expressive rules. As the definitions of chain-generated models is very general, it generalizes to programs that contain different logical connectives. This can be done similarly to (Alferes et al., 2000) for the extensions of the stable generated model semantics to programs with two kinds of negation.

Similar to the lifting from propositional to first-order programs by grounding, selector-generated models can be defined for logic programs with constraints (Maher, 1999). This is a generalization of stable generated models to programs with constraints as done in (Schwarz, 2001).

As mentioned in the definition of interpretations, the definition of selector-generated models can be extended to many-valued models. It would be interesting to compare them to partial stable models (Przymusinski, 1991) and partial stable generated models (Alferes et al., 2000).

From (Przymusinski, 1990) we know that the well-founded semantics (Gelder et al., 1991) of a normal logic program can be defined by partial models. Analogously, it is possible to define a class of well-founded semantics based on partial selector-generated models of generalized programs.

Especially for end-regulated models, there is another interesting question. To prove a model to be end-regulated, we have to construct a chain. This construction already depends on the chain that is constructed, as the bodies of all firing rules have to be persistent on the upper part of this chain. So we have to guess the model (as in stable and stable generated

models) and the chain. This is an additional source of non-determinism and increases the complexity of the test. It would be nice to have a bottom-up-algorithm to construct a chain that guarantees the persistence of the bodies during this construction.

8 Conclusions

We presented a framework to define several declarative semantics of generalized logic programs. These semantics are defined by selector functions that selects a set of interpretations from the interval from the least up to a guessed interpretation. Each of these semantics consists of the set of models that can be constructed by iterative application of the steps

1. Depending on the current and the guessed interpretation, the selector function chooses a set of interpretations.
2. Select all rules from program P, whose bodies are satisfied in all interpretations from 1.
3. Find all models of all heads of rules from 2, that are contained in the interval between the current and the guessed interpretation.
4. The successor is a minimal element of the set of interpretations from 3.

We found selector functions that generate minimal, stable, and stable generated models.

We studied properties of end-regulated models. They are a conservative extension of stable models of normal programs. It was shown that the end-regulated semantics is less tolerant with respect to program transformations than the stable generated model semantics. For programs containing rules with disjunctive bodies the end-regulated semantics is more fine-grained than Lifschitz, Tang and Turner's generalization of the stable semantics (Lifschitz et al., 1999).

References

Alferes, J. J., Herre, H., and Pereira, L. M. (2000). Partial models of extended generalized logic programs. In *Computational Logic*, pages 149–163.

Brass, S., Dix, J., Niemelä, I., and Przmusinski, T. C. (1998). A comparison of the static and the disjunctive well-founded semantics and its implementation. In Cohn, A. G., Schubert, L., and Shapiro, S. C., editors, *KR'98: Principles of Knowledge Representation and Reasoning*, pages 74–85. Morgan Kaufmann, San Francisco, California.

Gelder, A. V., Ross, K. A., and Schlipf, J. S. (1991). The well-founded semantics for general logic programs. *Journal of the ACM*, 38(3):620–650.

Gelfond, M. and Lifschitz, V. (1988). The stable model semantics for logic programming. In Kowalski, R. A. and Bowen, K., editors, *Proceedings of the Fifth International Conference on Logic Programming*, pages 1070–1080, Cambridge, Massachusetts. The MIT Press.

Herre, H. and Wagner, G. (1997a). Semantical foundation of extended generalized logic programs. Research Report 7, Universität Leipzig.

Herre, H. and Wagner, G. (1997b). Stable models are generated by a stable chain. *Journal of Logic Programming*, 30(2):165–177.

Lifschitz, V., Tang, L. R., and Turner, H. (1999). Nested expressions in logic programs. *Annals of Mathematics and Artificial Intelligence*, 25(3-4):369–389.

Maher, M. J. (1999). Adding constraints to logic-based formalisms. In Apt, K., Marek, V. W., Truszczyński, M., and Warren, D., editors, *The Logic Programming Paradigm: a 25 Years Perspective*, Artificial Intelligence Series, pages 313–331. Springer.

Przymusinski, T. (1990). Well-founded Semantics Coincides With Three-Valued Stable Semantics. *Fundamenta Informaticae*, XIII:445–463.

Przymusinski, T. (1991). Stable Semantics for Disjunctive Programs. *New Generation Computing Journal*, 9:401–424.

Przymusinski, T. (1994). Well-founded and stationary models of logic programs. *Annals of Mathematics and Artificial Intelligence*, 12:141–187.

Sakama, C. and Inoue, K. (1995). Paraconsistent stable semantics for extended disjunctive programs. *Journal of Logic and Computation*, 5(3):265–285.

Schwarz, S. (2001). Stable generated models of generalized constraint logic programs. In *Proceedings of the 10th International Workshop on Functional and (Constraint) Logic Programming (WFLP 2001)*. Universität Kiel.

Uncertainty Frameworks in NMR

Evaluating Defaults

Henry E. Kyburg, Jr.[1,2]
kyburg@cs.rochester.edu
[1]Computer Science and Philosophy
University of Rochester
Rochester NY 14627, USA

Choh Man Teng[2]
cmteng@ai.uwf.edu
[2]Institute for Human and Machine Cognition
University of West Florida
Pensacola FL 32501, USA

Abstract

We seek to find normative criteria of adequacy for nonmonotonic logic similar to the criterion of validity for deductive logic. Rather than stipulating that the conclusion of an inference be true in all models in which the premises are true, we require that the conclusion of a nonmonotonic inference be true in "almost all" models of a certain sort in which the premises are true. This "certain sort" specification picks out the models that are relevant to the inference, taking into account factors such as specificity and vagueness, and previous inferences. The frequencies characterizing the relevant models reflect known frequencies in our actual world. The criteria of adequacy for a default inference can be extended by thresholding to criteria of adequacy for an extension. We show that this avoids the implausibilities that might otherwise result from the chaining of default inferences. The model proportions, when construed in terms of frequencies, provide a verifiable grounding of default rules, and can become the basis for generating default rules from statistics.

Keywords: probability, frequency, default logic

1 Introduction

Non-monotonic reasoning, for example default logic [Reiter, 1980], models the intuitive process of making non-deductive inferences in the face of certain supportive but not conclusive evidence. Given a default theory $\Delta = \langle D, F \rangle$, we can obtain its extensions by following a prescribed set of steps. However, on what grounds do we employ a particular default rule? Some writers would regard this as an inappropriate question, since they take as their goal the representation of human inference. To this end defaults represent rules that we take to be intuitively appropriate. But then when we apply these rules, we may be led to counterintuitive results [Reiter and Criscuolo, 1981; Łukaszewicz, 1988; Poole, 1989]. The underlying principle seems to be circular: the original default rules are "intuitively good" at first glance, but when we discover that they do not give rise to the desired results, we tweak the rules until they give us those results. It seems that we have to know what results we want first before constructing the default theory, rather than having the default theory tell us what conclusions are warranted. This is precisely the reason why we need an independent measure of validity for default rules and default extensions. We think of nonmonotonic logic as sharing the normative character of other logics. From this point of view default rules require some defense. We will concentrate on default logic here, though much of what we have to say will apply to other nonmonotonic approaches as well. Much of the work on nonmonotonic logic has concerned the syntactic manipulation of the nonmonotonic rules, rather than their basic justification.

1.1 Selective Preference

For a *default rule* $d = \frac{\alpha:\beta_1,\ldots,\beta_n}{\gamma}$, α is the *prerequisite*, $\beta_1, \ldots, \beta_n$ are the *justifications*, and γ is the *consequent* of d. Loosely speaking, the rule conveys the idea that if α is provable, and $\neg\beta_1, \ldots, \neg\beta_n$ are each not provable, then by default we conclude that γ is true. A *default theory* is an ordered pair $\langle D, F \rangle$, where D is a set of default rules and F is a set of "facts". A theory extended from F by applying the default rules in D is known as an *extension* of the default theory.

Consider the following canonical example.

Example 1 We have a default theory $\Delta = \langle D, F \rangle$, where

$$D = \{\frac{R(x):T(x)}{T(x)}, \frac{S(x):\neg T(x)}{\neg T(x)}\},$$
$$F = \{R(a), S(a)\}.$$

We get two extensions, one containing $T(a)$ and the other containing $\neg T(a)$. If we take "$R(x)$" to mean that x is a bird, "$S(x)$" to mean that x is a penguin, and "$T(x)$" to mean that x flies, then we would like to reject the extension containing "$T(a)$" (a flies) in favor of the extension containing "$\neg T(a)$" (a does not fly). However, if we take "$S(x)$" to mean that x is an animal instead and keep "$R(x)$" and "$T(x)$" the same, we would want to reverse our preference. Now the extension containing "$T(a)$" (a flies) seems better. □

Note that each of the default rules involved in the example above is intuitively appealing when viewed by itself against our background knowledge: birds fly; penguins do not fly; and animals in general do not fly either. Moreover, both instantiations (penguins and animals) are syntactically identical. Thus, we cannot base our decision to prefer one default rule over the other by simply looking at their syntactic structures.

It is the interaction between the default rules and evidence that gives rise to the selective preference above. We have the evidence that "a is a bird". If in addition we also have "a is a penguin", we prefer the penguin rule. If instead we have "a is an animal", we prefer the bird rule.

There are several approaches to circumventing this conceptual difficulty. The first is to revise the default theory so that the desired result is achieved [Reiter and Criscuolo, 1981]. We can amend the default rules by adding the exceptions as justifications, for example $\frac{B(x):F(x),\neg P(x)}{F(x)}$ and $\frac{A(x):\neg F(x),\neg B(x)}{\neg F(x)}$. With this approach we have to constantly revise the default rules to take into account additional exceptions. We have little guidance in constructing the list of justifications except that the resulting default rule has to produce the "right" answer in the given situation.

Another approach is to establish some priority structure over the set of defaults. For example, we can refer to a specificity or inheritance hierarchy to determine which default rule should be used in case of a conflict [Touretzky, 1984; Horty *et al.*, 1987]. The penguin rule is more specific than the bird rule, when both are applicable, and therefore we use the penguin rule and not the bird rule. However, conflicting rules do not always fit into neat hierarchies (for example, adults are employed, students are not, how about adult students? [Reiter and Criscuolo, 1981]). It is not obvious how we can extend the hierarchical structure without resorting to explicitly enumerating the priority relations between the default rules [Brewka, 1989; Brewka, 1994].

The third approach is to appeal to probabilistic analysis. Defaults are interpreted as representing properties of conditional probabilities. For example, the conditional probability of a being able to fly given that a is a bird is "high" [Pearl, 1988; Pearl, 1990] or increases from the prior probability [Neufeld *et al.*, 1990], while the conditional probability of a being able to fly given that a is a penguin is "low" or decreases. This approach provides a probabilistic semantics for default rules, but in a way which does not represent the fact that the conclusions are *accepted*. The default conclusion is "Tweety flies," not "Probably Tweety flies." This is in contrast to the spirit of nonmonotonic reasoning: default conclusions should be accepted as new facts, and we should be able to chain default rules and build upon the conclusions of previous default applications to obtain further conclusions.

1.2 Justifying Nonmonotonic Inference

The justification of beliefs is a long standing issue in epistemology. There is not much that is problematic about the justification of beliefs obtained by deductive inference (though there are plenty of problems that surround deduction — see [Kyburg, 1958; Haack, 1976; Dummett, 1978], not to mention the voluminous literature on paraconsistent logic [Priest, 1989]). The reason is that we can show that the ordinary rules of deduction lead from premises to conclusions that are true in every model in which the premises are true. This is exactly what is not true of ampliative inference, and it is what has led some writers (e.g., [Morgan, 1998]) to deny that there *is* any such thing as a nonmonotonic logic. This has been disputed in [Kyburg, 2001].

But other kinds of justifications of beliefs have been proposed. Isaac Levi [Levi, 1967; Levi, 1980; Levi, 1996] has argued for many years that the way to understand ampliative (inductive, nonmonotonic) argument is in terms of decision theory: we choose (decide) to accept a hypothesis in a given context provided that the expected epistemic utility of doing so in that context is greater than the expected utility of any other epistemic act, such as suspending belief totally, or accepting a stronger hypothesis.

Levi's approach employs a rich and detailed structure for acceptance, and allows drawing many important

distinctions. This structure requires three things that make it less than perfect as a vehicle for ordinary nonmonotonic inferential systems. First, in keeping with a long tradition in pragmatism [Dewey, 1938; Peirce, 1903; James, 1948] the context of inquiry must be tied to a specific problem: We need the answer to a question. Second, the epistemic expectation of an answer is the expected value of the *information* contained in that answer. Thus we need to presuppose an information measure on the language of our inquiry [Levi, 1996, p. 169]. Third, we need to have available a credal or inductive probability, based on a measure (or convex set of measures) on the sentences of the language, in terms of which a conditional probability (or convex set of conditional probabilities) can be defined [Levi, 1980, p. 52].

It is our belief both that in some contexts in which we might wish to use nonmonotonic mechanisms, this overhead is unnecessary, and perhaps itself difficult to justify, and that we would like to be able to explicate the justification of inference in a less context dependent way.

Another approach that has attracted considerable attention in the philosophical community in recent years is that of "reliabilism" whose best known exponent is Alvin Goldman [Goldman, 1979]. According to this view, what justifies a belief is the fact that it is obtained by a "reliable cognitive process..." [Goldman, 1979, p. 20] Of course there are a number of additional hedges to the view that are required for philosophical accuracy, and even with those hedges there remains a certain vagueness in the view. These details need not detain us, since we are seeking inspiration rather than philosophical precision.

What does "reliable" mean? We will construe reliability in terms of frequency or propensity to yield truth when applied. Specifically, we will say that the belief ϕ is nonmonotonically justified by a default rule if the rule would frequently lead to truth and rarely to error, given what we know — given our background knowledge.

A deductive argument is justified (valid) if its conclusion is true in every model of its premises. We will attempt to provide an analog of the justification of deductive rules: a default argument is justified if its conclusion is true in a high proportion of the relevant models in which its premises are true. To make this idea precise requires an excursion into model theory.

2 Model Theory

We will suppose that the underlying object language is a first order language that does not involve such intensional predicates as "know" or "believe." A number of nonmonotonic formalisms (specifically autoepistemic logic [Moore, 1985]) do involve such locutions within the object language, but they can be dispensed with in default logic. The default rule $\frac{\alpha:\beta_1,\ldots,\beta_n}{\gamma}$ can be read in terms of the nonmembership of $\ulcorner\neg\beta_i\urcorner$ in a specified set of expressions Γ. In original default logic, Γ would just be an extension.

There are a number of immediate problems associated with the idea of looking at the "proportion" of models. The least of them is choosing a *level* at which to regard the evidence as adequate. Should we require that the proportion be 0.95? Or 0.99? Or 0.995? This is just the sort of question that arises in statistical hypothesis testing or in confidence interval estimation. We shall suppose that in a given context there is some agreed-upon level of security δ; we will accept a conclusion if the proportion of models in which we could be committing an error is no greater than δ.

This approach is to be contrasted with those of Adams [Adams, 1975; Adams, 1966], Pearl [Pearl, 1988] and Bacchus et al [Bacchus *et al.*, 1993]. Adams requires that for A to be a reasonable consequence of the set of sentences S, for *any* ϵ there must be a positive δ such that for every probability function, if the probability of every sentence in S is greater than $1-\delta$, then the probability of A is at least $1-\epsilon$ [Adams, 1966, p. 274]. Pearl's approach similarly involves quantification over possible probability functions. Bacchus et al again take the degree of belief of a statement to be the limiting proportion of first order models in which the statement is true. All of these approaches involve matters that go well beyond what we may reasonably suppose to be available to us as empirical enquirers. Our δ, on the other hand, serves much like the α of statistical testing.

We must restrict the number of models under consideration to a finite number so that the idea of looking at proportions makes sense.[1] We will be taking account of statistical information, and to this end will want each model to have a finite domain. Roughly speaking, we take as a model of our language one in which the domain of empirical individuals is of finite cardinality. This may be regarded as problematic (it entails the

[1] We could, instead, seek to develop a way of proceeding to a limit; this still would require restrictions to arrive at a countable number of models, and would entail a large expository cost for little gain in plausibility.

falsity of "every person has two parents and nobody is his own ancestor") but with reasonable spatial and temporal bounding it can be rendered plausible.

Even so, to ensure that the set of models is finite we must restrict the empirical domain even further. Not only must it consist of a finite set of physical entities, but this same set of physical entities $\mathcal{D}$ must be taken to be the empirical domain of every model.

We assume that it is possible to express statistical knowledge in this language. For example, if "$B(x)$" is the predicate "is a bird" and "$F(x)$" is "can fly", we can express the fact that between 85% and 95% of birds fly by the formula $0.85 < \frac{|\{x:B(x)\wedge F(x)\}|}{|\{x:B(x)\}|} < 0.95$. Employing the notation of [Kyburg and Teng, 2001] we write this as "$\%x(F(x), B(x), 0.85, 0.95)$." This renders "%" a variable binding operator on 4-sequences of expressions: two formulas and two fractions.

We distinguish, as do Pearl and Geffner [Pearl and Geffner, 1990, p. 70] between immediate evidence, represented by a finite set of sentences E concerning particular facts (to be distinguished from the general body of factual knowledge F invoked by classical default logic), and a finitely axiomatizable set of sentences K representing general background knowledge. What defaults are plausible depends, of course, on background knowledge. If it were not for what we take to be the typical (or natural, or frequent) behavior of birds, the world's best known example of a default rule would not be plausible. On the other hand no one has proposed the default rule $\frac{\text{fish}(a):\text{mackerel}(a)}{\text{can}-\text{talk}(a)}$.

Thus in general we will represent the set of default rules of a default theory as Δ_K rather than D, since we take them to be a function of our body of general knowledge K. Given an error tolerance δ, we will take a default rule to be *δ-valid* if, for every set of possible input sentences E consistent with K, the application of the rule to E leads to a false conclusion in a proportion of at most δ of the relevant models. More precisely, a default rule is δ-valid if and only if for every set of input sentences E consistent with K to which the rule is applicable, the proportion of models of $E \cup K$ in which the conclusion of the rule is false is no more than δ.

To fix our ideas, let us begin with a simple example. Suppose K includes a statement to the effect that at least $1-\delta$ and not more than $1 - \epsilon$ of birds fly and nothing else; that is, "$\%x(F(x), B(x), 1-\delta, 1-\epsilon)$." Consider the rule $\frac{B(x):F(x)}{F(x)}$. This rule is "applicable" to immediate evidence E only if $E \cup K$ entails a sentence of the form $\ulcorner B(a) \urcorner$ and no corresponding sentence of the form $\ulcorner \neg F(a) \urcorner$.

Our models have a single domain $\mathcal{D}$ of finite cardinality. We will write "$\mathcal{I}_m(\phi)$" for the interpretation of ϕ in the model m. The constraint imposed by K is that for every model m the proportion of objects in $\mathcal{I}_m(B)$ that are also in $\mathcal{I}_m(F)$ lies in $[1-\delta, 1-\epsilon]$.

There are three cases. First, suppose $E \cup K$ does not entail a sentence of the form $\ulcorner B(a) \urcorner$. Then the rule is inapplicable. Second, suppose that for some term a, $E \cup K$ entails "$B(a)$" and also entails "$\neg F(a)$". The rule is again inapplicable, because it is blocked by the failure of a justification. Third, suppose for some term a, $E \cup K$ entails "$B(a)$" but not "$\neg F(a)$". Then $\mathcal{I}_m(a) \in \mathcal{I}_m(B)$. There are $|\mathcal{I}_m(B)|$ interpretations of a that make $E \cup K$ true; of these at least $1-\delta$ make "$F(a)$" true. We have said nothing about interpreting the rest of the language, but however many interpretations there are (we have seen to it that there are only a finite number) the proportion that renders "$F(a)$" true will remain unchanged; it will be at least $1-\delta$. Thus, given the background knowledge that we have posited, the rule is δ-valid: if it is applicable it will lead to error no more than $1-\delta$ of the time.

Now let us consider a somewhat more complex example: Suppose we know that typically birds fly, and that typically penguins don't. If that is in our background knowledge K, as well as "$\forall x(P(x) \supset B(x))$", then the flying default becomes $\frac{B(x):F(x),\neg P(x)}{F(x)}$, and we also have the default $\frac{P(x):\neg F(x)}{\neg F(x)}$. If E entails "$P(a)$", only the second default is applicable. In no more than δ of the models of E will a fly, unless $E \cup K$ entails that a can fly.

Another example: Suppose K contains vague information about the frequency with which red birds fly (perhaps because we have encountered few red birds). Say that we know the frequency to be between 0.50 and 1.0. Since the interval for birds in general $[1-\delta, 1-\epsilon]$ is included in $[0.5, 1.0]$, this additional piece of information should not interfere with our inference of flying ability. There is no *conflict* between the two intervals, just less precision in one. The rule about birds in general can be applied to red birds. However, if K contains the knowledge that between 0.5 and r of red birds fly, where r is less than $1-\epsilon$, then this information *should* interfere. In this case the general rule should be so construed that it does not apply to red birds. If b is a red bird and not a penguin, no conclusion about flying ability is justified.

We can arrange this by judiciously adding or deleting justifications in the general bird rule, in accordance with the statistical information in K: in the first case we allow red birds; in the second we must require that

we do not know a is red: "$\neg R(x)$" must be among the justifications of the rule. This statistical approach provides exactly the normative guidance that is lacking in the ad-hoc approach of tweaking default rules in order to arrive at the "intuitive" results.

More generally, we can give recipes for constructing δ-valid defaults for conclusions of the form $\ulcorner\phi(a)\urcorner$ from background knowledge K and immediate evidence E.[2] Let K contain $\ulcorner\%x(\phi(x),\psi(x),p,q)\urcorner$ and $\ulcorner\%x(\phi(x),\psi'(x),p',q')\urcorner$. We consider three cases:

1. K entails $\ulcorner\forall x(\psi(x) \supset \psi'(x))\urcorner$. There are three subcases according to the relation among p, p', q, q':
   (a) $(p \leq p'$ and $q \leq q')$ or $(p' \leq p$ and $q' \leq q)$ $\frac{\psi(x):\phi(x)}{\phi(x)}$ and $\frac{\psi'(x):\neg\psi(x),\phi(x)}{\phi(x)}$ are candidate defaults.
   (b) $p \leq p'$ and $q' \leq q$ $\frac{\psi'(x):\phi(x)}{\phi(x)}$ is the only candidate default, since the justification $\neg\psi'(x)$ of $\frac{\psi(x):\phi(x),\neg\psi'(x)}{\phi(x)}$ is inconsistent with the prerequisite $\psi(x)$ and K.
   (c) $p' \leq p$ and $q \leq q'$ $\frac{\psi(x):\phi(x)}{\phi(x)}$ and $\frac{\psi'(x):\neg\psi(x),\phi(x)}{\phi(x)}$ are candidate defaults.
2. K entails $\ulcorner\forall x(\psi'(x) \supset \psi(x))\urcorner$. This is symmetrical to case 1.
3. K entails neither $\ulcorner\forall x(\psi(x) \supset \psi'(x))\urcorner$ nor $\ulcorner\forall x(\psi'(x) \supset \psi(x))\urcorner$. Again there are three subcases:
   (a) $(p \leq p'$ and $q \leq q')$ or $(p' \leq p$ and $q' \leq q)$ The candidate defaults are $\frac{\psi(x):\phi(x),\neg\psi'(x)}{\phi(x)}$ and $\frac{\psi'(x):\neg\psi(x),\phi(x)}{\phi(x)}$.
   (b) $p \leq p'$ and $q' \leq q$ $\frac{\psi(x):\phi(x),\neg\psi'(x)}{\phi(x)}$ and $\frac{\psi'(x):\phi(x)}{\phi(x)}$ are candidate default rules.
   (c) $p' \leq p$ and $q \leq q'$: This is symmetrical to case 3(b).

Having generated a list of candidate default rules based on our background knowledge K, we delete those rules derived from statistics with a lower measure less than $1-\delta$. The remainder is the set of defaults Δ_K.

We have not taken account of relations among default conclusions that may be entailed by K. If K contains $\ulcorner\forall x(\phi(x) \equiv \phi'(x))\urcorner$ then the default conclusion $\ulcorner\phi(a)\urcorner$ behaves just like the default conclusion $\ulcorner\phi'(a)\urcorner$. If K contains $\ulcorner\forall x(\phi(x) \supset \phi'(x))\urcorner$, then since $\ulcorner\phi(x)\urcorner$ is equivalent to $\ulcorner\phi(x) \wedge \phi'(x)\urcorner$ and $\ulcorner\phi'(x)\urcorner$ is equivalent to $\ulcorner\phi(x) \vee \phi'(x)\urcorner$ we can make use of the obvious entailment relations.

Soundness of a system of deductive logic requires that the conclusion of any inference be true in every model in which the premises are true. Clearly nonmonotonic inference should not be sound. But there is a property that is *like* soundness that applies to default inference. It is the property that the conclusion is false in at most a fraction δ of the models of the premises $K \cup E$.

Theorem 1 (Default Soundness) *For every set of observations E, if $d \in \Delta_K$ is applicable to E, the proportion of models of $E \cup K$ in which the conclusion of d is false is less than δ.*

The proof of this theorem is provided by the soundness theorem for evidential probability [Kyburg and Teng, 2001, p. 241], since the rules for deriving defaults are a subset of the rules for computing evidential probabilities. □

3 Interactions within an Extension

Having determined which default rules are justified with respect to the background knowledge, the next step is to investigate the interaction between default rules in generating an extension. A default extension is a minimal deductively closed set that contains the given facts and the consequents of all applicable default rules. Given an evidence set, we need to determine how to control the compound effects of multiple defaults in an extension.

Take for example, a default version [Poole, 1989] of the probabilistic lottery paradox [Kyburg, 1961]. There are n species of birds, $S_1, \ldots, S_n$. We can say that penguins are atypical in that they cannot fly; hummingbirds are atypical in that they have very fine motor control; parrots are atypical in that they could talk; and so on. If we apply this train of thought to all n species of birds, there is no typical bird left, as for each species there is always at least one aspect in which it is atypical. A parallel scenario is formulated below.

Example 2 K contains

[2] Of course any default conclusion can be given this form, particularly if we allow the term a to be an n-sequence of terms. Furthermore, any such conclusion can be taken to be an instance of the consequent of a statistical generalization, in virtue of the fact that statistical generalizations merely impose bounds. We are not imposing serious limitations on default rules. For details, see [Kyburg and Teng, 2001].

$B(x) \equiv S_1(x) \vee \ldots \vee S_n(x)$
[an exhaustive list of bird species]
$S_i(x) \supset \neg S_j(x)$, for all $j \neq i$
[species are mutually exclusive]
$\%(S_i(x), B(x), \epsilon_i, \delta_i)$, for $1 \leq i \leq n$
[the proportion of each S_i species of birds is "small"]

From K we can derive n δ^*-valid default rules for Δ_K:

$$d_i = \frac{B(x) : \neg S_i(x)}{\neg S_i(x)}, \text{for } 1 \leq i \leq n$$

where δ^* is the maximum of $\delta_1, \ldots, \delta_n$.

Now consider the evidence set $E = \{B(a)\}$. In the original formulation of default logic, we would get n extensions, each one containing one $S_i(a)$ and the negations of all the other $S_j(a)$'s. Thus, for each extension, we would conclude that a is a particular species of bird, which seems to be an over commitment, considering we have $\%(S_i(x), B(x), \epsilon_i, \delta_i)$ in K. □

Note that each of the n default rules is δ-valid when considered individually, but in an extension the rules interact to sanction a set of conclusions that when taken together seems implausible according to our knowledge of model frequencies. The definition of an extension dictates that we must keep applying rules until all "applicable" ones are exhausted. The "applicability" condition is based on maximizing logical strength: for $d = \frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}$, as long as α is derivable, and the β's are consistent with the extension, we must apply d and add γ to the extension. Thus, for each of the extensions above, we have to keep applying the rules until we have drawn $n-1$ conclusions: $\neg S_j(a)$ for all $j \neq i$. Then the consistency requirement blocks the last default rule d_i, as $B(x) \supset S_1(x) \vee \ldots \vee S_n(x)$ together with $\neg S_j(a)$ for all $j \neq i$ gives us $S_i(a)$, contradicting the β of d_i. From $\%(S_i(x), B(x), \epsilon_i, \delta_i)$, we know the proportion of models in which $S_i(a)$ is true, and thus the proportion of models satisfying this extension, given E, is at most δ_i, a small ratio.

3.1 Sequential Thresholding

The validity criteria for individual default rules can be extended to extensions resulted from the application of a chain of default rules. We can think of the task of regulating the compound effect of multiple default rules as adjusting the set of relevant models by taking into account the default conclusions of all previously applied rules in the chain of reasoning.

One way to accomplish this is by *sequential thresholding* [Teng, 1997]. The applicability condition of a default rule $\frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}$ in an extension can be modified to take into account the validity of the rule. In addition to requiring that α is provable and that none of $\neg\beta_1, \ldots, \neg\beta_n$ are provable, we require that the default rule be "above threshold", that is, the proportion of relevant models satisfying the consequent γ be greater than a threshold $1 - \epsilon^*$.

The set of relevant models shrinks in a stepwise fashion. We start out with all the models satisfying the background knowledge and evidence we have. As default rules are applied sequentially, the consequent of the applied rule at each step is taken as true in all subsequent steps. The relevant models at a particular step are then those that are consistent with the given facts and all the consequents of the rules applied in the previous steps. A default rule, even if it is δ-valid with respect to the background knowledge, would be blocked from application if it does not satisfy the thresholding criterion.

In [Teng, 1997], the thresholding metric is based on a simple probability measure of possible worlds. We can easily extend this metric to employ the same measure as that used for evaluating the δ-validity of default rules.

Example 3 Reconsider Example 2. Let us take $\epsilon^* \geq \delta^*$. We start out with the set $\mathcal{M}$ of all models satisfying K and E. From $\%(S_1(x), B(x), \epsilon_1, \delta_1)$ we know that d_1 is above threshold, and it satisfies the other conditions for applicability. Therefore we apply d_1 and conclude $\neg S_1(a)$.

Now consider d_2. The set $\mathcal{M}'$ of relevant models at this point is a subset of $\mathcal{M}$; it contains only those models in $\mathcal{M}$ that satisfy our new conclusion $\neg S_1(a)$ as well. We have eliminated the models in which $S_1(a)$ is true. Since $S_1(x) \supset \neg S_2(x)$, and $S_1(x) \supset B(x)$, all the models eliminated satisfy $B(a)$, and none satisfies $S_2(a)$. Thus, in $\mathcal{M}'$, the number of models satisfying $S_2(a)$ is the same as in $\mathcal{M}$. However, the number of models satisfying $B(a)$ is lower in $\mathcal{M}'$ as a result of the addition of $\neg S_1(a)$. This gives rise to a higher proportion of models satisfying $S_2(a)$ in $\mathcal{M}'$ (δ_2') than in $\mathcal{M}$ (δ_2). If $\delta_2' \leq \epsilon^*$, d_2 is still above threshold after the application of d_1, and we can apply it to obtain $\neg S_2(a)$. Otherwise, d_2 is below threshold, and we cannot apply it even though it was above threshold before the application of d_1.

After each step of applying a rule, the set of relevant models shrinks, and the proportion of $S_i(a)$ of any unapplied rule d_i increases. After a number of steps, all the remaining rules would be below threshold, and we thus obtain an extension containing only a portion

of the conclusions that would otherwise be present in the non-thresholding version of the extension. □

Note that the size of ϵ^* determines how much risk is tolerated in an extension. The higher the ϵ^*, the more of the rules can be applied and the longer they can stay above threshold. Reiter's non-thresholding version corresponds to the case when $\epsilon^* = 1$; that is, every rule whose associated proportion is above 0 is allowed, and logical consistency alone determines the rule's admissibility.

4 Concluding Remarks

We have developed a notion of validity for default inference based on model proportions. A rule is δ-valid if the proportion of models in which the consequent of the rule is satisfied is greater than $1 - \delta$ in the relevant models picked out by the background knowledge, the evidence, and the applicability conditions of the default rule. Given a body of background knowledge K, we can systematically generate candidate default rules and determine which ones are δ-valid based on the statistical facts known in K. Conflicts between default rules stemming from multiple inheritance are resolved as a consequence of the validation process. The result is a set of δ-valid default rules which are "precompiled" for a given background knowledge base, and can be reused for different evidence sets without change.

This idea of evaluating the validity of a default rule using model proportions is extended to extensions generated by a combinations of rules. The compound effect is regulated by a sequential thresholding process, which blocks the rules whose associated model proportions with respect to the "current" (shrinking) set of models fall below a particular comfort threshold. This allows us to use a more reasonable "closure condition" for extension than the usual maximal logical strength: we can refrain from applying rules that would make the extension satisfiable in only a small set of models, even if the consequent of the rule is logically consistent with the extension.

Grounding the justification of default rules in model proportions provides a way to validate the rules empirically, and is a first step towards automating the learning of default rules from (statistical) data. One might ask why we need the default rules when we can reason with the statistics directly. Default rules provide a succinct and more understandable characterization of the import of the data, as well as a smooth articulation of the information that may exist in the knowledge base.

Acknowledgement

This work was supported by the National Science Foundation STS-9906128, IIS-0082928, and NASA NCC2-1239.

References

[Adams, 1966] Ernest W. Adams. Probability and the logic of conditionals. In Jaakko Hintikka and Patrick Suppes, editors, *Aspects of Inductive Logic*, pages 265–316. North Holland, Amsterdam, 1966.

[Adams, 1975] Ernest W. Adams. *The Logic of Conditionals.* Reidel, Dordrecht, 1975.

[Bacchus *et al.*, 1993] Fahiem Bacchus, Adam J. Grove, Joesph Y. Halpern, and Daphne Koller. Statistical foundations for default reasoning. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, pages 563–569, 1993.

[Brewka, 1989] Gerhard Brewka. Preferred subtheories—an extended logical framework for default reasoning. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, 1989.

[Brewka, 1994] Gerhard Brewka. Reasoning about priorities in default logic. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 940–945, 1994.

[Dewey, 1938] John Dewey. *Logic: the Theory of Inquiry.* Henry Holt, 1938.

[Dummett, 1978] Michael Dummett. The justification of deduction. In *Truth and Other Enigmas*, pages 290–318. Duckworth, London, 1978.

[Goldman, 1979] Alvin I. Goldman. What is justified belief? In George S. Pappas, editor, *Justification and Knowledge*, pages 1–24. D. Reidel, Dordrecht, 1979.

[Haack, 1976] Susan Haack. The justification of deduction. *Mind*, 85:112–119, 1976.

[Horty *et al.*, 1987] J. F. Horty, D. S. Touretzky, and R. H. Thomason. A clash of intuitions: The current state of nonmonotonic multiple inheritance systems. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, pages 476–482, 1987.

[James, 1948] William James. *Essays in Pragmatism.* Hafner, New York, 1948.

[Kyburg and Teng, 2001] Henry E. Kyburg, Jr. and Choh Man Teng. *Uncertain Inference.* Cambridge University Press, New York, 2001.

[Kyburg, 1958] Henry E. Jr. Kyburg. The justification of deduction. *Review of Metaphysics*, 12:19–25, 1958.

[Kyburg, 1961] Henry E. Kyburg, Jr. *Probability and the Logic of Rational Belief.* Wesleyan University Press, 1961.

[Kyburg, 2001] Henry E. Kyburg, Jr. Real logic is nonmonotonic. *Minds and Machines*, 11:577–595, 2001.

[Levi, 1967] Isaac Levi. *Gambling with Truth.* Knopf, New York, 1967.

[Levi, 1980] Isaac Levi. *The Enterprise of Knowledge,.* MIT Press, Cambridge, 1980.

[Levi, 1996] Isaac Levi. *For the Sake of the Argument.* Cambridge University Press, Cambridge, 1996.

[Łukaszewicz, 1988] Witold Łukaszewicz. Considerations on default logic: An alternative approach. *Computational Intelligence*, 4(1):1–16, 1988.

[Moore, 1985] Robert C. Moore. Semantical considerations on nonmonotonic logic. *Artificial Intelligence*, 25:75–94, 1985.

[Morgan, 1998] Charles Morgan. Non-monotonic logic is impossible. *Canadian Artificial Intelligence Magazine*, 42:18–25, 1998.

[Neufeld *et al.*, 1990] Eric Neufeld, David Poole, and Romas Aleliunas. Probabilistic semantics and defaults. In *Uncertainty in Artificial Intelligence*, volume 4, pages 121–131. North-Holland, 1990.

[Pearl and Geffner, 1990] Judea Pearl and Hector Geffner. A framework for reasoning with defaults. In Henry E. Kyburg, Jr., Ronald P. Loui, and Greg N. Carlson, editors, *Knowledge Representation and Defeasible Reasoning*, pages 69–87. Kluwer, 1990.

[Pearl, 1988] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems.* Morgan Kaufmann, 1988.

[Pearl, 1990] Judea Pearl. System Z: A natural ordering of defaults with tractable applications to default reasoning. In *Theoretical Aspects of Reasoning about Knowledge*, pages 121–135, 1990.

[Peirce, 1903] Charles S. Peirce. The fixation of belief. In J. Buchler, editor, *The Philosophy of Peirce.*, pages 5–22. Harcourt Brace and Company, New York, 1903.

[Poole, 1989] David Poole. What the lottery paradox tells us about default reasoning. In *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*, pages 333–340, 1989.

[Priest, 1989] Graham Priest. Reasoning about truth. *Artificial Intelligence*, 39:231–244, 1989.

[Reiter and Criscuolo, 1981] Raymond Reiter and Giovanni Criscuolo. On interacting defaults. In *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, pages 270–276, 1981.

[Reiter, 1980] R. Reiter. A logic for default reasoning. *Artificial Intelligence*, 13:81–132, 1980.

[Teng, 1997] Choh Man Teng. Sequential thresholds: Context sensitive default extensions. In *Proceedings of the Thirteen Conference of Uncertainty in Artificial Intelligence*, pages 437–444, 1997.

[Touretzky, 1984] David S. Touretzky. Implicit ordering of defaults in inheritance systems. In *Proceedings of the Fifth National Conference on Artificial Intelligence*, pages 322–325, 1984.

Nonmonotonic Probabilistic Logics between Model-Theoretic Probabilistic Logic and Probabilistic Logic under Coherence

Thomas Lukasiewicz*
Dipartimento di Informatica e Sistemistica,
Università di Roma "La Sapienza"
Via Salaria 113, 00198 Roma, Italy
lukasiewicz@dis.uniroma1.it

Abstract

Recently, it has been shown that probabilistic entailment under coherence is weaker than model-theoretic probabilistic entailment. Moreover, probabilistic entailment under coherence is a generalization of default entailment in System P. In this paper, we continue this line of research by presenting probabilistic generalizations of more sophisticated notions of classical default entailment that lie between model-theoretic probabilistic entailment and probabilistic entailment under coherence. That is, the new formalisms properly generalize their counterparts in classical default reasoning, they are weaker than model-theoretic probabilistic entailment, and they are stronger than probabilistic entailment under coherence. The new formalisms are useful especially for handling probabilistic inconsistencies related to conditioning on zero events. They can also be applied for probabilistic belief revision. More generally, in the same spirit as a similar previous paper, this paper sheds light on exciting new formalisms for probabilistic reasoning beyond the well-known standard ones.

1 Introduction

In the recent decades, probabilistic reasoning has become an important research topic in artificial intelligence. In particular, extensive research has been carried out on probabilistic reasoning with interval restrictions for conditional probabilities, also called *conditional constraints* [40].

One important approach for handling conditional constraints is model-theoretic probabilistic logic, which comes especially from philosophy and logic, and whose roots go back to Boole's book of 1854 "The Laws of Thought" [11]. There is a wide spectrum of formal languages that have been explored in model-theoretic probabilistic logic, ranging from constraints for unconditional and conditional events [16, 20, 2, 53, 19, 23, 33, 45, 39, 40, 42, 46] to rich languages that specify linear inequalities over events [22]. The main problems related to model-theoretic probabilistic logic are checking satisfiability, deciding logical consequence, and computing tight logically entailed intervals.

Another important approach to probabilistic reasoning with conditional constraints, which has been extensively explored in the area of statistics, is based on the coherence principle of de Finetti and suitable generalizations of it [8, 12, 13, 14, 15, 27, 28, 29, 50], or on similar principles that have been adopted for lower and upper probabilities [49, 54]. Two important aspects of dealing with uncertainty in this framework are (i) checking the consistency of a probabilistic assessment, and (ii) the propagation of a given assessment to further uncertain quantities.

Recently, the relationship between model-theoretic probabilistic logic and probabilistic logic under coherence has been thoroughly explored in [10]. In particular, it turned out that probabilistic entailment under coherence is weaker than model-theoretic probabilistic entailment. Moreover, it has been shown in [10] that probabilistic entailment under coherence is a generalization of classical default entailment in System P, while model-theoretic probabilistic entailment is well-known to be a generalization of model-theoretic entailment in classical propositional logics.

In this paper, we continue this line of research by presenting probabilistic generalizations of more sophisticated formalisms for default reasoning from conditional knowledge bases, which lie between probabilistic entailment under coherence and model-theoretic probabilistic entailment. That is, the new notions of entailment generalize their classical counterparts, they are stronger than entailment under coherence, and weaker than model-theoretic entailment.

*Alternate address: Institut für Informationssysteme, Technische Universität Wien, Favoritenstraße 9-11, 1040 Wien, Austria. E-mail: lukasiewicz@kr.tuwien.ac.at.

The literature contains several different proposals for default reasoning from conditional knowledge bases and extensive work on its desired properties. The core of these properties are the rationality postulates of System P proposed by Kraus et al. [34]. It turned out that these rationality postulates constitute a sound and complete axiom system for several classical model-theoretic entailment relations under uncertainty measures on worlds. In detail, they characterize classical model-theoretic entailment under preferential structures [51, 34], infinitesimal probabilities [1, 47], possibility measures [17], and world rankings [52, 31]. They also characterize an entailment relation based on conditional objects [18]. A survey of all these relationships is given in [6, 24].

Mainly to solve problems with irrelevant information, the notion of rational closure as a more adventurous notion of entailment was introduced by Lehmann [36, 38]. It is equivalent to entailment in System Z by Pearl [48], to the least specific possibility entailment by Benferhat et al. [5], and to a conditional (modal) logic-based entailment by Lamarre [35]. Finally, mainly to solve problems with property inheritance from classes to exceptional subclasses, the maximum entropy approach to default entailment was proposed by Goldszmidt et al. [30]; lexicographic entailment was introduced by Lehmann [37] and Benferhat et al. [4]; conditional entailment was proposed by Geffner [25, 26]; and an infinitesimal belief function approach was suggested by Benferhat et al. [7].

In this paper, we define probabilistic generalizations of entailment in Pearl's System Z and Lehmann's lexicographic entailment that lie between entailment in model-theoretic probabilistic logic and entailment in probabilistic logic under coherence. Roughly speaking, the main difference between model-theoretic probabilistic entailment and probabilistic entailment under coherence is that the former realizes an inheritance of logical knowledge, while the latter does not. Intuitively, the new formalisms now add a strategy for resolving inconsistencies to model-theoretic probabilistic logic, and a restricted form of inheritance of logical knowledge to probabilistic logic under coherence. This is why the new notions of entailment are weaker than entailment in model-theoretic probabilistic logic and stronger than entailment in probabilistic logic under coherence.

Hence, the new formalisms are refinements of both model-theoretic probabilistic entailment and probabilistic entailment under coherence. They can be used in place of model-theoretic probabilistic entailment, when we want to resolve probabilistic inconsistencies related to *conditioning on zero events*. Here, the new formalisms are especially well-suited as they coincide with model-theoretic probabilistic entailment as long as we condition on non-zero events. Furthermore, the new formalisms can be used in place of probabilistic entailment under coherence, when we also want to have an inheritance of logical knowledge. Finally, another important application is probabilistic belief revision.

A companion paper [44] presents similar probabilistic generalizations of entailment in Pearl's System Z and of Lehmann's lexicographic entailment. These formalisms, however, are quite different from the ones in this paper. In detail, the new formalisms in [44] add an inheritance of probabilistic knowledge and a strategy for resolving inconsistencies to entailment in model-theoretic probabilistic logic. For this reason, they are generally much stronger than entailment in model-theoretic probabilistic logic. Thus, they are especially useful where the classical notion of model-theoretic entailment is too weak, for example, in probabilistic logic programming [42, 41]. Other applications are deriving degrees of belief from statistical knowledge and degrees of belief, handling inconsistencies in probabilistic knowledge bases, and probabilistic belief revision.

The main contributions of this paper are as follows:

- We show that the notions of g-coherence and of g-coherent entailment in probabilistic logic under coherence can be expressed in terms of probability rankings.
- We present new probabilistic generalizations of entailment in Pearl's System Z and Lehmann's lexicographic entailment, which lie between model-theoretic probabilistic entailment and probabilistic entailment under coherence.
- We analyze and compare some general nonmonotonic properties of the new notions of entailment, of model-theoretic probabilistic entailment, and of probabilistic entailment under coherence. In particular, it turns out that the new formalisms satisfy the properties of Rational Monotonicity and Irrelevance, while probabilistic entailment under coherence is lacking these properties.
- We analyze the relationship between the new notion of entailment in System Z, the new notion of lexicographic entailment, model-theoretic probabilistic entailment, and probabilistic entailment under coherence.
- We show that the new notions of entailment in System Z and of lexicographic entailment are proper generalizations of their classical counterparts.

The rest of this paper is organized as follows. Section 2 gives some technical preliminaries. In Section 3, we show that probabilistic reasoning under coherence can be expressed in terms of probability rankings, and we introduce the new probabilistic notions of entailment in System Z and of lexicographic entailment. Section 4 concentrates on the properties of the new formalisms and of the standard ones. In Section 5, we summarize the main results and give

an outlook on future research. Note that detailed proofs of all results are given in the extended paper [43].

2 Preliminaries

In this section, we first define probabilistic knowledge bases. We then recall the notions of satisfiability and of logical entailment from model-theoretic probabilistic logic, and the notions of g-coherence and of g-coherent entailment from probabilistic logic under coherence.

2.1 Probabilistic Knowledge Bases

We assume a set of *basic events* $\Phi = \{p_1, \ldots, p_n\}$ with $n \geq 1$. We use $\bot$ and $\top$ to denote *false* and *true*, respectively. We define *events* by induction as follows. Every element of $\Phi \cup \{\bot, \top\}$ is an event. If ϕ and ψ are events, then also $\neg\phi$ and $(\phi \wedge \psi)$. A *logical constraint* is an event of the form $\psi \Leftarrow \phi$. A *conditional event* is an expression of the form $\psi|\phi$ with events ψ and ϕ. A *conditional constraint* is an expression of the form $(\psi|\phi)[l,u]$ with events ψ, ϕ, and real numbers $l, u \in [0,1]$. We define *probabilistic formulas* by induction as follows. Every conditional constraint is a probabilistic formula. If F and G are probabilistic formulas, then also $\neg F$ and $(F \wedge G)$. We use $(F \vee G)$ and $(F \Leftarrow G)$ to abbreviate $\neg(\neg F \wedge \neg G)$ and $\neg(\neg F \wedge G)$, respectively, where F and G are either two events or two probabilistic formulas. We adopt the usual conventions to eliminate parentheses.

A *world* I is a truth assignment to the basic events in Φ (that is, a mapping $I\colon \Phi \rightarrow \{\mathbf{true}, \mathbf{false}\}$), which is inductively extended to all events by $I(\bot) = \mathbf{false}$, $I(\top) = \mathbf{true}$, $I(\neg\phi) = \mathbf{true}$ iff $I(\phi) = \mathbf{false}$, and $I((\phi \wedge \psi)) = \mathbf{true}$ iff $I(\phi) = I(\psi) = \mathbf{true}$. We use $\mathcal{I}_\Phi$ to denote the set of all worlds for Φ. A world I *satisfies* an event ϕ, or I is a *model* of ϕ, denoted $I \models \phi$, iff $I(\phi) = \mathbf{true}$. We extend worlds I to conditional events $\psi|\phi$ by $I(\psi|\phi) = \mathbf{true}$ iff $I \models \psi \wedge \phi$, $I(\psi|\phi) = \mathbf{false}$ iff $I \models \neg\psi \wedge \phi$, and $I(\psi|\phi) = \mathbf{indeterminate}$ iff $I \models \neg\phi$. A *probabilistic interpretation* Pr is a probability function on $\mathcal{I}_\Phi$ (that is, a mapping $Pr\colon \mathcal{I}_\Phi \rightarrow [0,1]$ such that all $Pr(I)$ with $I \in \mathcal{I}_\Phi$ sum up to 1). The *probability* of an event ϕ in the probabilistic interpretation Pr, denoted $Pr(\phi)$, is the sum of all $Pr(I)$ such that $I \in \mathcal{I}_\Phi$ and $I \models \phi$. For events ϕ and ψ with $Pr(\phi) > 0$, we write $Pr(\psi|\phi)$ to abbreviate $Pr(\psi \wedge \phi) \,/\, Pr(\phi)$. The *truth* of logical constraints and probabilistic formulas F in a probabilistic interpretation Pr, denoted $Pr \models F$, is defined as follows:

- $Pr \models \psi \Leftarrow \phi$ iff $Pr(\psi \wedge \phi) = Pr(\phi)$.
- $Pr \models (\psi|\phi)[l,u]$ iff $Pr(\phi) = 0$ or $Pr(\psi|\phi) \in [l,u]$.
- $Pr \models \neg F$ iff not $Pr \models F$.
- $Pr \models (F \wedge G)$ iff $Pr \models F$ and $Pr \models G$.

We say Pr *satisfies* F, or Pr is a *model* of F, iff $Pr \models F$. We say Pr *satisfies* a set of logical and conditional constraints $\mathcal{F}$, or Pr is a *model* of $\mathcal{F}$, denoted $Pr \models \mathcal{F}$, iff Pr is a model of all $F \in \mathcal{F}$.

A *probabilistic knowledge base* $KB = (L, P)$ consists of a finite set of logical constraints L and a finite set of conditional constraints P such that (i) $l \leq u$ for all $(\psi|\phi)[l,u] \in P$, and (ii) $\psi_1|\phi_1 \neq \psi_2|\phi_2$ for any two distinct $(\psi_1|\phi_1)[l_1,u_1]$, $(\psi_2|\phi_2)[l_2,u_2] \in P$. Note that here $\psi_1|\phi_1 \neq \psi_2|\phi_2$ iff either $\psi_1 \wedge \phi_1$ is not logically equivalent to $\psi_2 \wedge \phi_2$, or ϕ_1 is not logically equivalent to ϕ_2.

Example 2.1 A sample probabilistic knowledge base $KB = (L, P)$ is given as follows:

$$L = \{\mathsf{bird} \Leftarrow \mathsf{penguin}\},$$
$$P = \{(\mathsf{have_legs} \mid \mathsf{bird})[1,1], (\mathsf{fly} \mid \mathsf{bird})[1,1]\}.$$

Here, L represents the logical knowledge "all penguins are birds". Moreover, in model-theoretic probabilistic logic, P expresses the *logical knowledge* "all birds have legs" and "all birds fly", while in probabilistic logic under coherence and in the new nonmonotonic logics, P expresses the *default logical knowledge* "typically, birds have legs" and "typically, birds fly".

Another probabilistic knowledge base $KB' = (L', P')$ is given as follows:

$$L' = \{\mathsf{bird} \Leftarrow \mathsf{penguin}\},$$
$$P' = \{(\mathsf{have_legs} \mid \mathsf{bird})[1,1], (\mathsf{fly} \mid \mathsf{bird})[1,1],$$
$$(\mathsf{fly} \mid \mathsf{penguin})[0, 0.05]\}.$$

In model-theoretic probabilistic logic, the additional conditional constraint $(\mathsf{fly} \mid \mathsf{penguin})[0, 0.05]$ represents the *probabilistic knowledge* "at most 5% of all penguins fly", while in probabilistic logic under coherence and in the new nonmonotonic logics, $(\mathsf{fly} \mid \mathsf{penguin})[0, 0.05]$ represents the *default probabilistic knowledge* "typically, penguins fly with a probability of at least 0.05". □

2.2 Model-Theoretic Probabilistic Logic

We now recall the model-theoretic notions of satisfiability and of logical entailment.

A set of logical and conditional constraints $\mathcal{F}$ is *satisfiable* iff a model of $\mathcal{F}$ exists. A probabilistic knowledge base $KB = (L, P)$ is *satisfiable* iff $L \cup P$ is satisfiable.

A conditional constraint C is a *logical consequence* of a set of logical and conditional constraints $\mathcal{F}$, denoted $\mathcal{F} \mid\!\models C$, iff every model of $\mathcal{F}$ is also a model of C. A conditional

constraint $(\psi|\phi)[l,u]$ is a *tight logical consequence* of $\mathcal{F}$, denoted $\mathcal{F}|\!\models_{tight}(\psi|\phi)[l,u]$, iff l (resp., u) is the infimum (resp., supremum) of $Pr(\psi|\phi)$ subject to all models Pr of $\mathcal{F}$ with $Pr(\phi)>0$. Note that we define $l=1$ and $u=0$, when $\mathcal{F}\not\models(\phi|\top)[0,0]$. A conditional constraint C is a *logical consequence* of a probabilistic knowledge base KB, denoted $KB\models C$, iff $L\cup P\models C$. It is a *tight logical consequence* of KB, denoted $KB|\!\models_{tight}C$, iff $L\cup P|\!\models_{tight}C$.

Example 2.2 The probabilistic knowledge base KB of Example 2.1 has the following logical consequences:

$$(\mathsf{fly}\,|\,\mathsf{bird})[1,1],\ (\mathsf{have_legs}\,|\,\mathsf{bird})[1,1],$$
$$(\mathsf{fly}\,|\,\mathsf{penguin})[1,1],\ (\mathsf{have_legs}\,|\,\mathsf{penguin})[1,1].$$

In fact, all these conditional constraints are even tight logical consequences of KB.

The probabilistic knowledge base $KB'=(L',P')$ of Example 2.1 has the following tight logical consequences:

$$(\mathsf{fly}\,|\,\mathsf{bird})[1,1],\ (\mathsf{have_legs}\,|\,\mathsf{bird})[1,1],$$
$$(\mathsf{fly}\,|\,\mathsf{penguin})[1,0],\ (\mathsf{have_legs}\,|\,\mathsf{penguin})[1,0].$$

Here, the empty interval "$[1,0]$" in the last two conditional constraints is due to the fact that the logical property of being able to fly is inherited from birds to penguins and is incompatible there with the knowledge that at most 5% of all penguins are able to fly. Hence, in this example, the notion of logical entailment is too strong, as the desirable tight conclusions from KB' are given by $(\mathsf{fly}\,|\,\mathsf{penguin})[0,0.05]$ and $(\mathsf{have_legs}\,|\,\mathsf{penguin})[1,1]$, respectively. □

2.3 Probabilistic Logic under Coherence

We now recall the notions of g-coherence and of g-coherent entailment. We define them by using some recent characterizations through concepts from default reasoning [10]. We first give some preparative definitions.

A probabilistic interpretation Pr *verifies* a conditional constraint $(\psi|\phi)[l,u]$ iff $Pr(\phi)>0$ and $Pr\models(\psi|\phi)[l,u]$. It *falsifies* a conditional constraint $(\psi|\phi)[l,u]$ iff $Pr(\phi)>0$ and $Pr\not\models(\psi|\phi)[l,u]$. A set of conditional constraints P *tolerates* a conditional constraint C *under* a set of logical constraints L iff $L\cup P$ has a model that verifies C. A set of conditional constraints P is *under* L *in conflict* with C iff no model of $L\cup P$ verifies C.

A *conditional constraint ranking* σ on a probabilistic knowledge base $KB=(L,P)$ maps each element of P to a nonnegative integer. It is *admissible* with KB iff every $P'\subseteq P$ that is under L in conflict with some $C\in P$ contains a conditional constraint C' such that $\sigma(C')<\sigma(C)$.

We are now ready to define the concept of g-coherence. A probabilistic knowledge base KB is *g-coherent* iff there exists a conditional constraint ranking on KB that is admissible with KB.

We next define the notion of g-coherent entailment. Let $KB=(L,P)$ be a g-coherent probabilistic knowledge base, and let $(\psi|\phi)[l,u]$ be a conditional constraint. Then, $(\psi|\phi)[l,u]$ is a *g-coherent consequence* of KB, denoted $KB\,|\!\sim^{g}(\psi|\phi)[l,u]$, iff $(L,P\cup\{(\psi|\phi)[p,p]\})$ is not g-coherent for all $p\in[0,l)\cup(u,1]$. We say $(\psi|\phi)[l,u]$ is a *tight g-coherent consequence* of KB, denoted $KB\,|\!\sim^{g}_{tight}(\psi|\phi)[l,u]$, iff l (resp., u) is the infimum (resp., supremum) of p subject to all g-coherent probabilistic knowledge bases $(L,P\cup\{(\psi|\phi)[p,p]\})$.

Example 2.3 The probabilistic knowledge base KB of Example 2.1 has the following tight g-coherent consequences:

$$(\mathsf{fly}\,|\,\mathsf{bird})[1,1],\ (\mathsf{have_legs}\,|\,\mathsf{bird})[1,1],$$
$$(\mathsf{fly}\,|\,\mathsf{penguin})[0,1],\ (\mathsf{have_legs}\,|\,\mathsf{penguin})[0,1].$$

Here, the interval "$[0,1]$" in the last two conditional constraints is due to the fact that the logical properties of being able to fly and of having legs are not inherited from birds to penguins. Thus, in this example, the notion of g-coherent entailment is too weak, as the desirable tight conclusions from KB are given by $(\mathsf{fly}\,|\,\mathsf{penguin})[1,1]$ and $(\mathsf{have_legs}\,|\,\mathsf{penguin})[1,1]$, respectively.

The probabilistic knowledge base KB' of Example 2.1 has the following tight g-coherent consequences:

$$(\mathsf{fly}\,|\,\mathsf{bird})[1,1],\ (\mathsf{have_legs}\,|\,\mathsf{bird})[1,1],$$
$$(\mathsf{fly}\,|\,\mathsf{penguin})[0,0.05],\ (\mathsf{have_legs}\,|\,\mathsf{penguin})[0,1].$$

Here, the last interval "$[0,1]$" is due to the fact that the logical property of having legs is not inherited from birds to penguins. Thus, also in this example, g-coherent entailment is too weak, as the desirable tight conclusion from KB' is given by $(\mathsf{have_legs}\,|\,\mathsf{penguin})[1,1]$. □

3 Nonmonotonic Probabilistic Logics

In this section, we first show that the notions of g-coherence and of g-coherent entailment can be expressed in terms of probability rankings. We then introduce new probabilistic generalizations of Pearl's entailment in System Z and of Lehmann's lexicographic entailment. Observe that the notion of g-coherent entailment is defined through a set of probability rankings, while the new notions of entailment in System Z and of lexicographic entailment are each defined through a unique single probability ranking.

In some sense, the new formalisms are second-order formalisms, as probability rankings are closely related to possibility measures over probabilities and can thus be seen as uncertainty measures over other uncertainty measures.

3.1 Probability Rankings

We now show that g-coherence and g-coherent entailment can be defined in terms of admissible probability rankings. We first give some preparative definitions.

In the sequel, we use $\alpha > 0$ to abbreviate the probabilistic formula $\neg(\alpha|\top)[0,0]$. A *probability ranking* κ maps each probabilistic interpretation on $\mathcal{I}_\Phi$ to a member of $\{0,1,\ldots\} \cup \{\infty\}$ such that $\kappa(Pr) = 0$ for at least one interpretation Pr. It is extended to all logical constraints and probabilistic formulas F as follows. If F is satisfiable, then $\kappa(F) = \min\{\kappa(Pr) \mid Pr \models F\}$; otherwise, $\kappa(F) = \infty$. A probability ranking κ is *admissible* with a probabilistic knowledge base $KB = (L,P)$ iff $\kappa(\neg F) = \infty$ for all $F \in L$ and $\kappa(\phi > 0) < \infty$ and $\kappa(\phi > 0 \wedge (\psi|\phi)[l,u]) < \kappa(\phi > 0 \wedge \neg(\psi|\phi)[l,u])$ for all $(\psi|\phi)[l,u] \in P$.

The following result shows that g-coherence is equivalent to the existence of an admissible probability ranking.

Theorem 3.1 *Let $KB = (L,P)$ be a probabilistic knowledge base. Then, KB is g-coherent iff there exists a probability ranking κ that is admissible with KB.*

The next theorem shows that g-coherent entailment can be expressed in terms of admissible probability rankings.

Theorem 3.2 *Let $KB = (L,P)$ be a g-coherent probabilistic knowledge base, and let $(\psi|\phi)[l,u]$ be a conditional constraint. Then, $KB \mid\!\sim^g (\psi|\phi)[l,u]$ iff $\kappa(\phi > 0) = \infty$ or $\kappa(\phi > 0 \wedge (\psi|\phi)[l,u]) < \kappa(\phi > 0 \wedge \neg(\psi|\phi)[l,u])$ for every probability ranking κ that is admissible with KB.*

3.2 System Z

We now extend Pearl's entailment in System Z [48, 32] to g-coherent probabilistic knowledge bases $KB = (L,P)$. The new notion of entailment in System Z is linked to an ordered partition of P, a conditional constraint ranking z on KB, and a probability ranking κ^z.

The *z-partition* of KB is the unique ordered partition $(P_0,\ldots,P_k)$ of P such that each P_i, $i \in \{0,\ldots,k\}$, is the set of all $C \in P - (P_0 \cup \cdots \cup P_{i-1})$ that is tolerated under L by $P - (P_0 \cup \cdots \cup P_{i-1})$.

Example 3.3 The z-partition of the probabilistic knowledge base $KB = (L,P)$ in Example 2.1 is given by:

$$(P_0) = (\{(\mathsf{have_legs} \mid \mathsf{bird})[1,1], (\mathsf{fly} \mid \mathsf{bird})[1,1]\}).$$

The z-partition of the probabilistic knowledge base $KB' = (L',P')$ in Example 2.1 is given as follows:

$$(P_0', P_1') = (\{(\mathsf{have_legs} \mid \mathsf{bird})[1,1], (\mathsf{fly} \mid \mathsf{bird})[1,1]\}, \{(\mathsf{fly} \mid \mathsf{penguin})[0,0.05]\}). \quad \square$$

We next define the conditional constraint ranking z and the probability ranking κ^z. For every $j \in \{0,\ldots,k\}$, each $C \in P_j$ is assigned the value j under z. The probability ranking κ^z on all probabilistic interpretations Pr is then defined as follows:

$$\kappa^z(Pr) = \begin{cases} \infty & \text{if } Pr \not\models L \\ 0 & \text{if } Pr \models L \cup P \\ 1 + \max\limits_{C \in P:\ Pr \not\models C} z(C) & \text{otherwise.} \end{cases}$$

The following lemma shows that, in fact, z is a conditional constraint ranking on KB that is admissible with KB, and κ^z is a probability ranking that is admissible with KB.

Lemma 3.4 *Let KB be a g-coherent probabilistic knowledge base. Then,*
(a) z is admissible with KB.
(b) κ^z is admissible with KB.

We next define a preference relation on probabilistic interpretations as follows. For probabilistic interpretations Pr and Pr', we say Pr is *z-preferable* to Pr' iff $\kappa^z(Pr) < \kappa^z(Pr')$. A model Pr of a set of logical and conditional constraints $\mathcal{F}$ is a *z-minimal model* of $\mathcal{F}$ iff no model of $\mathcal{F}$ is z-preferable to Pr.

We finally define the notion of *z-entailment* as follows. A conditional constraint C is a *z-consequence* of KB, denoted $KB \mid\!\sim^z C$, iff every z-minimal model of L satisfies C. A conditional constraint $(\psi|\phi)[l,u]$ is a *tight z-consequence* of KB, denoted $KB \mid\!\sim^z_{tight} (\psi|\phi)[l,u]$, iff l (resp., u) is the infimum (resp., supremum) of $Pr(\psi|\phi)$ subject to all z-minimal models Pr of L with $Pr(\phi) > 0$.

The following example shows that z-entailment realizes an inheritance of logical properties from classes to non-exceptional subclasses. However, logical properties are not inherited from classes to subclasses that are exceptional with respect to some other property (in classical default reasoning from conditional knowledge bases, we call this undesirable feature *inheritance blocking*).

Example 3.5 The probabilistic knowledge base KB of Example 2.1 has the following tight z-consequences:

$$(\mathsf{fly}|\mathsf{bird})[1,1],\ (\mathsf{have_legs}|\mathsf{bird})[1,1],$$
$$(\mathsf{fly}|\mathsf{penguin})[1,1],\ (\mathsf{have_legs}|\mathsf{penguin})[1,1].$$

The probabilistic knowledge base KB' of Example 2.1 has the following tight z-consequences:

$$(\mathsf{fly}|\mathsf{bird})[1,1],\ (\mathsf{have_legs}|\mathsf{bird})[1,1],$$
$$(\mathsf{fly}|\mathsf{penguin})[0,0.05],\ (\mathsf{have_legs}|\mathsf{penguin})[0,1].$$

Here, the last interval "[0, 1]" is due to the fact that the logical property of having legs is not inherited from birds to its exceptional subclass penguins. □

The following lemma describes an alternative way of defining the notion of z-consequence.

Lemma 3.6 *Let KB be a g-coherent probabilistic knowledge base, and let $C = (\psi|\phi)[l,u]$ be a conditional constraint. Then, $KB \mid\!\sim^z C$ iff $\kappa^z(\phi > 0) = \infty$ or $\kappa^z(\phi > 0 \wedge C) < \kappa^z(\phi > 0 \wedge \neg C)$.*

3.3 Lexicographic Entailment

We next extend Lehmann's lexicographic entailment [37] to g-coherent probabilistic knowledge bases $KB = (L, P)$. Note that even though we do not use probability rankings here, the new notion of entailment can be easily expressed by a unique single probability ranking.

We use the z-partition $(P_0, \ldots, P_k)$ of KB to define a lexicographic preference relation on probabilistic interpretations as follows. For probabilistic interpretations Pr and Pr', we say Pr is *lexicographically preferable* (or *lex-preferable*) to Pr' iff some $i \in \{0, \ldots, k\}$ exists such that $|\{C \in P_i \mid Pr \models C\}| > |\{C \in P_i \mid Pr' \models C\}|$ and $|\{C \in P_j \mid Pr \models C\}| = |\{C \in P_j \mid Pr' \models C\}|$ for all $i < j \leq k$. A model Pr of a set of logical and conditional constraints $\mathcal{F}$ is a *lexicographically minimal model* (or *lex-minimal model*) of $\mathcal{F}$ iff no model of $\mathcal{F}$ is lex-preferable to Pr.

A conditional constraint C is a *lex-consequence* of KB, denoted $KB \mid\!\sim^{lex} C$, iff every lex-minimal model of L satisfies C. A conditional constraint $(\psi|\phi)[l,u]$ is a *tight lex-consequence* of KB, denoted $KB \mid\!\sim^{lex}_{tight} (\psi|\phi)[l,u]$, iff l (resp., u) is the infimum (resp., supremum) of $Pr(\psi|\phi)$ subject to all lex-minimal models Pr of L with $Pr(\phi) > 0$.

The following example shows that lex-entailment realizes a correct inheritance of logical properties. In particular, it does not show the problem of inheritance blocking.

Example 3.7 The probabilistic knowledge base KB of Example 2.1 has the following tight lex-consequences:

$$(\mathsf{fly} \mid \mathsf{bird})[1,1],\ (\mathsf{have_legs} \mid \mathsf{bird})[1,1],$$
$$(\mathsf{fly} \mid \mathsf{penguin})[1,1],\ (\mathsf{have_legs} \mid \mathsf{penguin})[1,1].$$

The probabilistic knowledge base KB' of Example 2.1 has the following tight lex-consequences:

$$(\mathsf{fly} \mid \mathsf{bird})[1,1],\ (\mathsf{have_legs} \mid \mathsf{bird})[1,1],$$
$$(\mathsf{fly} \mid \mathsf{penguin})[0, 0.05],\ (\mathsf{have_legs} \mid \mathsf{penguin})[1,1].$$ □

4 Properties

In this section, we analyze some properties of logical entailment, g-coherent entailment, z-entailment, and lex-entailment. We first describe some general nonmonotonic properties. We then explore the relationship between the formalisms, and the one to their classical counterparts.

4.1 General Nonmonotonic Properties

We now analyze some general nonmonotonic properties of the probabilistic formalisms of this paper. It is important to point out that these properties are especially related to the inheritance of logical knowledge.

We first consider the postulates *Right Weakening (RW)*, *Reflexivity (Ref)*, *Left Logical Equivalence (LLE)*, *Cut*, *Cautious Monotonicity (CM)*, and *Or* by Kraus et al. [34], which are commonly regarded as being particularly desirable for any reasonable notion of nonmonotonic entailment. The following result shows that logical entailment, g-coherent entailment, z-entailment, and lex-entailment all satisfy (probabilistic versions of) these postulates.

Theorem 4.1 *$\models$, $\mid\!\sim^g$, $\mid\!\sim^z$, and $\mid\!\sim^{lex}$ satisfy the following properties for all probabilistic knowledge bases $KB = (L, P)$, all events ε, ε', ϕ, and ψ, and all $l, l', u, u' \in [0,1]$:*

RW. *If $(\phi|\top)[l,u] \Rightarrow (\psi|\top)[l',u']$ is logically valid and $KB \mid\!\sim (\phi|\varepsilon)[l,u]$, then $KB \mid\!\sim (\psi|\varepsilon)[l',u']$.*

Ref. *$KB \mid\!\sim (\varepsilon|\varepsilon)[1,1]$.*

LLE. *If $\varepsilon \Leftrightarrow \varepsilon'$ is logically valid, then $KB \mid\!\sim (\phi|\varepsilon)[l,u]$ iff $KB \mid\!\sim (\phi|\varepsilon')[l,u]$.*

Cut. *If $KB \mid\!\sim (\varepsilon|\varepsilon')[1,1]$ and $KB \mid\!\sim (\phi|\varepsilon \wedge \varepsilon')[l,u]$, then $KB \mid\!\sim (\phi|\varepsilon')[l,u]$.*

CM. *If $KB \mid\!\sim (\varepsilon|\varepsilon')[1,1]$ and $KB \mid\!\sim (\phi|\varepsilon')[l,u]$, then $KB \mid\!\sim (\phi|\varepsilon \wedge \varepsilon')[l,u]$.*

Or. *If $KB \mid\!\sim (\phi|\varepsilon)[1,1]$ and $KB \mid\!\sim (\phi|\varepsilon')[1,1]$, then $KB \mid\!\sim (\phi|\varepsilon \vee \varepsilon')[1,1]$.*

Another desirable property is *Rational Monotonicity (RM)* [34], which describes a restricted form of monotony, and allows to ignore certain kinds of irrelevant knowledge. The next theorem shows that logical entailment, z-entailment, and lex-entailment satisfy *RM*. Here, we use $KB \mid\!\not\sim C$ to denote that it is not the case that $KB \mid\!\sim C$.

Theorem 4.2 *$\models$, $\mid\!\sim^z$, and $\mid\!\sim^{lex}$ satisfy the following property for all probabilistic knowledge bases $KB = (L, P)$, all events ε, ε', and ψ, and all $l, u \in [0,1]$:*

Figure 1: Logical Implications between Consequence Relations

RM. *If* $KB \mid\!\sim (\psi|\varepsilon)[1,1]$ *and* $KB \not\mid\!\sim (\neg\varepsilon'|\varepsilon)[1,1]$, *then* $KB \mid\!\sim (\psi|\varepsilon \wedge \varepsilon')[1,1]$.

The following example shows that g-coherent entailment, however, generally does not satisfy *RM*.

Example 4.3 Consider the probabilistic knowledge base

$$KB = (L,P) = (\{\mathsf{bird} \Leftarrow \mathsf{eagle}\}, \{(\mathsf{fly} \mid \mathsf{bird})[1,1]\}).$$

Then, $(\mathsf{fly} \mid \mathsf{bird})[1,1]$ is a logical consequence (resp., g-coherent consequence, z-consequence, lex-consequence) of KB. Moreover, $(\neg\mathsf{eagle} \mid \mathsf{bird})[1,1]$ is not a logical consequence (resp., g-coherent consequence, z-consequence, lex-consequence) of KB.

Observe now that $(\mathsf{fly} \mid \mathsf{bird} \wedge \mathsf{eagle})[1,1]$ is a logical consequence (resp., z-consequence, lex-consequence) of KB, but $(\mathsf{fly} \mid \mathsf{bird} \wedge \mathsf{eagle})[1,1]$ is not a g-coherent consequence of KB. We remark that $(\mathsf{fly} \mid \mathsf{bird} \wedge \mathsf{eagle})[1,1]$ is the tight logical consequence (resp., z-consequence, lex-consequence) of KB, while $(\mathsf{fly} \mid \mathsf{bird} \wedge \mathsf{eagle})[0,1]$ is the tight g-coherent consequence of KB. □

We now consider the property *Irrelevance (Irr)* adapted from [7]. Informally, *Irr* says that ε' is irrelevant to a conclusion "$P \mid\!\sim (\psi|\varepsilon)[1,1]$" when they are defined over disjoint sets of atoms. The following result shows that logical entailment, z-entailment, and lex-entailment satisfy *Irr*.

Theorem 4.4 $\models$, $\mid\!\sim^z$, *and* $\mid\!\sim^{lex}$ *satisfy the following property for all probabilistic knowledge bases* $KB = (L,P)$ *and all events* ε, ε', *and* ψ:

Irr. *If* $KB \mid\!\sim (\psi|\varepsilon)[1,1]$, *and no atom of* KB *and* $(\psi|\varepsilon)[1,1]$ *occurs in* ε', *then* $KB \mid\!\sim (\psi|\varepsilon\wedge\varepsilon')[1,1]$.

The following example shows that g-coherent entailment, however, does not satisfy *Irr*.

Example 4.5 Consider the probabilistic knowledge base

$$KB = (L,P) = (\emptyset, \{(\mathsf{fly} \mid \mathsf{bird})[1,1]\}).$$

Clearly, $(\mathsf{fly} \mid \mathsf{bird})[1,1]$ is a logical consequence (resp., g-coherent consequence, z-consequence, lex-consequence) of KB. Observe now that $(\mathsf{fly} \mid \mathsf{red} \wedge \mathsf{bird})[1,1]$ is a logical consequence (resp., z-consequence, lex-consequence) of KB, but $(\mathsf{fly} \mid \mathsf{red} \wedge \mathsf{bird})[1,1]$ is not a g-coherent consequence of KB. We remark that $(\mathsf{fly} \mid \mathsf{red} \wedge \mathsf{bird})[1,1]$ is the tight logical consequence (resp., z-consequence, lex-consequence) of KB, while $(\mathsf{fly} \mid \mathsf{red} \wedge \mathsf{bird})[0,1]$ is the tight g-coherent consequence of KB. □

We finally consider the property *Direct Inference (DI)* adapted from [3]. Informally, *DI* expresses that P should entail all its own conditional constraints. The following theorem shows that logical entailment, g-coherent entailment, z-entailment, and lex-entailment all satisfy *DI*.

Theorem 4.6 $\models$, $\mid\!\sim^g$, $\mid\!\sim^z$, *and* $\mid\!\sim^{lex}$ *satisfy the following property for all probabilistic knowledge bases* $KB = (L,P)$, *all events* ε, ϕ, *and* ψ, *and all* $l, u \in [0,1]$:

DI. *If* $(\psi|\phi)[l,u] \in P$ *and* $\varepsilon \Leftrightarrow \phi$ *is logically valid, then* $KB \mid\!\sim (\psi|\varepsilon)[l,u]$.

4.2 Relationship between Probabilistic Formalisms

We now explore the logical implications between the probabilistic formalisms of this paper.

The following theorem shows that the logical implications illustrated in Fig. 1 hold between logical entailment, g-coherent entailment, z-entailment, and lex-entailment.

Theorem 4.7 *Let* $KB = (L,P)$ *be a g-coherent probabilistic knowledge base, and let* $C = (\psi|\phi)[l,u]$ *be a conditional constraint. Then,*

(a) $KB \mid\!\sim^g C$ *implies* $KB \mid\!\sim^z C$.

(b) $KB \mid\!\sim^z C$ *implies* $KB \mid\!\sim^{lex} C$.

(c) $KB \mid\!\sim^{lex} C$ *implies* $KB \models C$.

In general, none of the converse implications holds, as Examples 2.2, 2.3, 3.5, and 3.7 immediately show. But in the special case when $L \cup P$ has a model in which the conditioning event ϕ has a positive probability, the notions of logical entailment, z-entailment, and lex-entailment of $(\psi|\phi)[l,u]$ from KB all coincide. This important result is expressed by the following theorem.

Theorem 4.8 *Let* $KB = (L,P)$ *be a g-coherent probabilistic knowledge base, and let* $C = (\psi|\phi)[l,u]$ *be a*

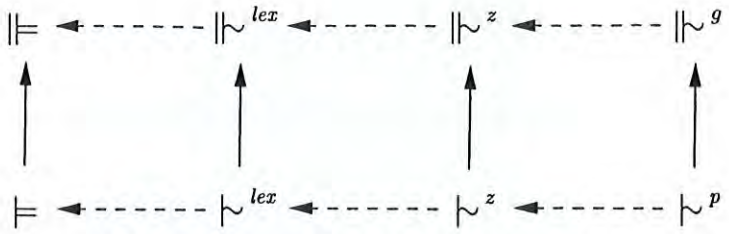


Figure 2: Relationship to Classical Formalisms

conditional constraint such that $L \cup P$ has a model Pr with $Pr(\phi) > 0$. Then,

$$KB \,|\!\models C \text{ iff } KB \,|\!\sim^{lex} C \text{ iff } KB \,|\!\sim^{z} C.$$

The following example shows that g-coherent entailment, however, generally does not coincide with logical entailment when $L \cup P$ has a model Pr with $Pr(\phi) > 0$.

Example 4.9 Consider again the probabilistic knowledge base $KB = (L, P) = (\{\text{bird} \Leftarrow \text{eagle}\}, \{(\text{fly} \mid \text{bird})[1,1]\})$. Then, $L \cup P$ has clearly a model Pr with $Pr(\text{eagle}) > 0$.

It is now easy to verify that $(\text{fly} \mid \text{eagle})[1,1]$ is a logical consequence (resp., z-consequence, lex-consequence) of KB, but $(\text{fly} \mid \text{eagle})[1,1]$ is not a g-coherent consequence of KB. We remark that $(\text{fly} \mid \text{eagle})[1,1]$ is the tight logical consequence (resp., z-consequence, lex-consequence) of KB, while $(\text{fly} \mid \text{eagle})[0,1]$ is the tight g-coherent consequence of KB. □

4.3 Relationship to Classical Formalisms

We now analyze the relationship between the probabilistic formalisms of this paper and their classical counterparts. We first give some technical preparation as follows.

The operator γ on conditional constraints, sets of conditional constraints, and conditional knowledge bases replaces each conditional constraint $(\psi|\phi)[1,1]$ by the classical default $\psi \leftarrow \phi$. We use the expressions $\mid\!\sim^{p}$, $\mid\!\sim^{z}$, $\mid\!\sim^{lex}$, and $\models$ to denote the classical notions of entailment in System P, Pearl's entailment in System Z, Lehmann's lexicographic entailment, and logical entailment, respectively.

The following well-known result says that logical entailment in model-theoretic probabilistic logic is a generalization of logical entailment in classical propositional logic.

Theorem 4.10 *Let $KB = (L, P)$ be a g-coherent probabilistic knowledge base, where $P = \{(\psi_1|\phi_1)[1,1], \ldots, (\psi_n|\phi_n)[1,1]\})$, and let $(\beta|\alpha)[1,1]$ be a conditional constraint. Then,*

$$KB \,|\!\models (\beta|\alpha)[1,1] \text{ iff } \gamma(KB) \models \beta \Leftarrow \alpha.$$

The next theorem recalls the recent result by Biazzo et al. [10] that g-coherent entailment is a generalization of default entailment in System P.

Theorem 4.11 (Biazzo et al. [10]) *Let $KB = (L, P)$ be a g-coherent probabilistic knowledge base, where $P = \{(\psi_1|\phi_1)[1,1], \ldots, (\psi_n|\phi_n)[1,1]\})$, and let $(\beta|\alpha)[1,1]$ be a conditional constraint. Then,*

$$KB \,|\!\sim^{g} (\beta|\alpha)[1,1] \text{ iff } \gamma(KB) \mid\!\sim^{p} \beta \leftarrow \alpha.$$

The following theorem shows that the new notions of z- and lex-entailment for g-coherent probabilistic knowledge bases generalize their classical counterparts for ε-consistent conditional knowledge bases.

Theorem 4.12 *Let $KB = (L, P)$ be a g-coherent probabilistic knowledge base, where $P = \{(\psi_1|\phi_1)[1,1], \ldots, (\psi_n|\phi_n)[1,1]\})$, and let $(\beta|\alpha)[1,1]$ be a conditional constraint. Then,*

(a) $KB \,|\!\sim^{z} (\beta|\alpha)[1,1]$ *iff* $\gamma(KB) \mid\!\sim^{z} \beta \leftarrow \alpha$.

(b) $KB \,|\!\sim^{lex} (\beta|\alpha)[1,1]$ *iff* $\gamma(KB) \mid\!\sim^{lex} \beta \leftarrow \alpha$.

5 Summary and Outlook

We have presented probabilistic generalizations of Pearl's entailment in System Z and of Lehmann's lexicographic entailment that lie between entailment in model-theoretic probabilistic logic and entailment in probabilistic logic under coherence. That is, the new formalisms properly generalize their counterparts in classical default reasoning, they are weaker than model-theoretic probabilistic entailment, and they are stronger than probabilistic entailment under coherence. The new formalisms are useful especially for handling probabilistic inconsistencies related to conditioning on zero events. Furthermore, they can be applied for probabilistic belief revision. More generally, in the same spirit as a similar previous paper, this paper sheds light on exciting new formalisms for probabilistic reasoning beyond the well-known standard ones.

An interesting topic of future research is to develop algorithms for the new formalisms and to analyze their computational complexity (along the lines of [44]).

Another very interesting topic of future research is to develop and explore further nonmonotonic formalisms for reasoning with conditional constraints. Besides extending classical formalisms for default reasoning from conditional knowledge bases, which may additionally contain a strength assignment to the defaults, one may also think about combining the new formalisms of this paper and of [44] with some probability selection technique (as e.g. maximum entropy or center of mass).

Acknowledgments

This work has been supported by a Marie Curie Individual Fellowship of the European Community and by the Austrian Science Fund Project Z29-INF. I am grateful to Angelo Gilio and Lluis Godo for their useful comments.

References

[1] E. W. Adams. *The Logic of Conditionals*, volume 86 of *Synthese Library*. D. Reidel, Dordrecht, Netherlands, 1975.

[2] S. Amarger, D. Dubois, and H. Prade. Constraint propagation with imprecise conditional probabilities. In *Proceedings UAI-91*, pages 26–34. Morgan Kaufmann, 1991.

[3] F. Bacchus, A. Grove, J. Halpern, and D. Koller. From statistical knowledge bases to degrees of belief. *Artif. Intell.*, 87:75–143, 1996.

[4] S. Benferhat, C. Cayrol, D. Dubois, J. Lang, and H. Prade. Inconsistency management and prioritized syntax-based entailment. In *Proceedings IJCAI-93*, pages 640–645. Morgan Kaufmann, 1993.

[5] S. Benferhat, D. Dubois, and H. Prade. Representing default rules in possibilistic logic. In *Proceedings KR-92*, pages 673–684. Morgan Kaufmann, 1992.

[6] S. Benferhat, D. Dubois, and H. Prade. Nonmonotonic reasoning, conditional objects and possibility theory. *Artif. Intell.*, 92(1–2):259–276, 1997.

[7] S. Benferhat, A. Saffiotti, and P. Smets. Belief functions and default reasoning. *Artif. Intell.*, 122(1–2):1–69, 2000.

[8] V. Biazzo and A. Gilio. A generalization of the fundamental theorem of de Finetti for imprecise conditional probability assessments. *Int. J. Approx. Reasoning*, 24(2–3):251–272, 2000.

[9] V. Biazzo, A. Gilio, T. Lukasiewicz, and G. Sanfilippo. Probabilistic logic under coherence: Complexity and algorithms. In *Proceedings ISIPTA-01*, pages 51–61, 2001.

[10] V. Biazzo, A. Gilio, T. Lukasiewicz, and G. Sanfilippo. Probabilistic logic under coherence, model-theoretic probabilistic logic, and default reasoning. In *Proceedings ECSQARU-01*, volume 2143 of *LNCS/LNAI*, pages 290–302. Springer, 2001.

[11] G. Boole. *An Investigation of the Laws of Thought, on which are Founded the Mathematical Theories of Logic and Probabilities*. Walton and Maberley, London, 1854. (reprint: Dover Publications, New York, 1958).

[12] G. Coletti. Coherent numerical and ordinal probabilistic assessments. *IEEE Trans. Syst. Man Cybern.*, 24(12):1747–1754, 1994.

[13] G. Coletti and R. Scozzafava. Characterization of coherent conditional probabilities as a tool for their assessment and extension. *Journal of Uncertainty, Fuzziness and Knowledge-based Systems*, 4(2):103–127, 1996.

[14] G. Coletti and R. Scozzafava. Coherent upper and lower Bayesian updating. In *Proceedings ISIPTA-99*, pages 101–110, 1999.

[15] G. Coletti and R. Scozzafava. Conditioning and inference in intelligent systems. *Soft Computing*, 3(3):118–130, 1999.

[16] D. Dubois and H. Prade. On fuzzy syllogisms. *Computational Intelligence*, 4(2):171–179, 1988.

[17] D. Dubois and H. Prade. Possibilistic logic, preferential models, non-monotonicity and related issues. In *Proceedings IJCAI-91*, pages 419–424. Morgan Kaufmann, 1991.

[18] D. Dubois and H. Prade. Conditional objects as nonmonotonic consequence relationships. *IEEE Trans. Syst. Man Cybern.*, 24(12):1724–1740, 1994.

[19] D. Dubois, H. Prade, L. Godo, and R. L. de Màntaras. Qualitative reasoning with imprecise probabilities. *Journal of Intelligent Information Systems*, 2:319–363, 1993.

[20] D. Dubois, H. Prade, and J.-M. Touscas. Inference with imprecise numerical quantifiers. In Z. W. Ras and M. Zemankova, editors, *Intelligent Systems*, chapter 3, pages 53–72. Ellis Horwood, 1990.

[21] T. Eiter and T. Lukasiewicz. Default reasoning from conditional knowledge bases: Complexity and tractable cases. *Artif. Intell.*, 124(2):169–241, 2000.

[22] R. Fagin, J. Y. Halpern, and N. Megiddo. A logic for reasoning about probabilities. *Inf. Comput.*, 87:78–128, 1990.

[23] A. M. Frisch and P. Haddawy. Anytime deduction for probabilistic logic. *Artif. Intell.*, 69:93–122, 1994.

[24] D. M. Gabbay and P. Smets, editors. *Handbook on Defeasible Reasoning and Uncertainty Management Systems*. Kluwer Academic, Dordrecht, Netherlands, 1998.

[25] H. Geffner. *Default Reasoning: Causal and Conditional Theories*. MIT Press, 1992.

[26] H. Geffner and J. Pearl. Conditional entailment: Bridging two approaches to default reasoning. *Artif. Intell.*, 53(2–3):209–244, 1992.

[27] A. Gilio. Probabilistic consistency of conditional probability bounds. In *Advances in Intelligent Computing*, volume 945 of *LNCS*, pages 200–209. Springer, 1995.

[28] A. Gilio. Probabilistic reasoning under coherence in System P. *Ann. Math. Artif. Intell.*, 34(1–3):5–34, 2002.

[29] A. Gilio and R. Scozzafava. Conditional events in probability assessment and revision. *IEEE Trans. Syst. Man Cybern.*, 24(12):1741–1746, 1994.

[30] M. Goldszmidt, P. Morris, and J. Pearl. A maximum entropy approach to nonmonotonic reasoning. *IEEE Trans. Pattern Anal. Mach. Intell.*, 15(3):220–232, 1993.

[31] M. Goldszmidt and J. Pearl. Rank-based systems: A simple approach to belief revision, belief update and reasoning about evidence and actions. In *Proceedings KR-92*, pages 661–672. Morgan Kaufmann, 1992.

[32] M. Goldszmidt and J. Pearl. Qualitative probabilities for default reasoning, belief revision, and causal modeling. *Artif. Intell.*, 84(1–2):57–112, 1996.

[33] J. Heinsohn. Probabilistic description logics. In *Proceedings UAI-94*, pages 311–318. Morgan Kaufmann, 1994.

[34] S. Kraus, D. Lehmann, and M. Magidor. Nonmonotonic reasoning, preferential models and cumulative logics. *Artif. Intell.*, 14(1):167–207, 1990.

[35] P. Lamarre. A promenade from monotonicity to nonmonotonicity following a theorem prover. In *Proceedings KR-92*, pages 572–580. Morgan Kaufmann, 1992.

[36] D. Lehmann. What does a conditional knowledge base entail? In *Proceedings KR-89*, pages 212–222. Morgan Kaufmann, 1989.

[37] D. Lehmann. Another perspective on default reasoning. *Ann. Math. Artif. Intell.*, 15(1):61–82, 1995.

[38] D. Lehmann and M. Magidor. What does a conditional knowledge base entail? *Artif. Intell.*, 55(1):1–60, 1992.

[39] T. Lukasiewicz. Local probabilistic deduction from taxonomic and probabilistic knowledge-bases over conjunctive events. *Int. J. Approx. Reasoning*, 21(1):23–61, 1999.

[40] T. Lukasiewicz. Probabilistic deduction with conditional constraints over basic events. *J. Artif. Intell. Res.*, 10:199–241, 1999.

[41] T. Lukasiewicz. Probabilistic logic programming under inheritance with overriding. In *Proceedings UAI-01*, pages 329–336. Morgan Kaufmann, 2001.

[42] T. Lukasiewicz. Probabilistic logic programming with conditional constraints. *ACM Trans. Computat. Logic*, 2(3):289–339, 2001.

[43] T. Lukasiewicz. Nonmonotonic probabilistic logics between model-theoretic probabilistic logic and probabilistic logic under coherence. Technical Report INFSYS RR-1843-02-02, Institut für Informationssysteme, TU Wien, 2002.

[44] T. Lukasiewicz. Probabilistic default reasoning with conditional constraints. *Ann. Math. Artif. Intell.*, 34(1–3):35–88, 2002.

[45] C. Luo, C. Yu, J. Lobo, G. Wang, and T. Pham. Computation of best bounds of probabilities from uncertain data. *Computational Intelligence*, 12(4):541–566, 1996.

[46] N. J. Nilsson. Probabilistic logic. *Artif. Intell.*, 28:71–88, 1986.

[47] J. Pearl. Probabilistic semantics for nonmonotonic reasoning: A survey. In *Proceedings KR-89*, pages 505–516. Morgan Kaufmann, 1989.

[48] J. Pearl. System Z: A natural ordering of defaults with tractable applications to default reasoning. In *Proceedings TARK-90*, pages 121–135. Morgan Kaufmann, 1990.

[49] R. Pelessoni and P. Vicig. A consistency problem for imprecise conditional probability assessments. In *Proceedings IPMU-98*, pages 1478–1485, 1998.

[50] R. Scozzafava. Subjective conditional probability and coherence principles for handling partial information. *Mathware Soft Comput.*, 3(1):183–192, 1996.

[51] Y. Shoham. A semantical approach to nonmonotonic logics. In *Proceedings of the 2nd IEEE Symposium on Logic in Computer Science*, pages 275–279, 1987.

[52] W. Spohn. Ordinal conditional functions: A dynamic theory of epistemic states. In W. Harper and B. Skyrms, editors, *Causation in Decision, Belief Change, and Statistics*, volume 2, pages 105–134. Reidel, Dordrecht, Netherlands, 1988.

[53] H. Thöne, U. Güntzer, and W. Kießling. Towards precision of probabilistic bounds propagation. In *Proceedings UAI-92*, pages 315–322. Morgan Kaufmann, 1992.

[54] P. Walley. *Statistical Reasoning with Imprecise Probabilities*. Chapman and Hall, 1991.

Default Logic in a Coherent Setting

Giulianella Coletti
Dipartimento
Matematica e Informatica
Università di Perugia
06100 Perugia (Italy)

Romano Scozzafava*
Dipartimento
Metodi e Modelli Matematici
Università La Sapienza
00161 Roma (Italy)

Barbara Vantaggi
Dipartimento
Metodi e Modelli Matematici
Università La Sapienza
00161 Roma (Italy)

Abstract

In this talk – based on the results of a forthcoming paper (Coletti, Scozzafava and Vantaggi 2002), presented also by one of us at the Conference on "Non Classical Logic, Approximate Reasoning and Soft-Computing" (Anacapri, Italy, 2001) – we discuss the problem of representing default rules by means of a suitable coherent conditional probability, defined on a family of conditional events. An event is singled-out (in our approach) by a *proposition*, that is a statement that can be either *true* or *false*; a conditional event is consequently defined by means of two propositions and is a 3–valued entity, the third value being (in this context) a conditional probability.

1 INTRODUCTION

The concept of conditional event (as dealt with in this paper) plays a central role for the probabilistic reasoning. We give up (or better, in a sense, we generalize) the idea of de Finetti of looking at a conditional event $E|H$, with $H \neq \emptyset$ (the *impossible* event), as a 3–valued logical entity looked on as "undetermined" when H is false: it is *true* when both E and H are true, *false* when H is true and E is false, while we let the third value *suitably depend on the given ordered pair* (E, H) and not being just an undetermined *common value* for all pairs. It turns out (as explained in detail in Coletti and Scozzafava 1999) that this function can be seen as a measure of the degree of belief in the conditional event $E|H$, which under "natural" conditions reduces to the conditional probability $P(E|H)$, in its most general sense related to the concept of *coherence*, and satisfying the classic axioms as given by de Finetti (1949), Rényi (1956), Krauss (1968), Dubins (1975): see Section 2. Notice that our concept of conditional event differs from that adopted, for example, by Adams (1975), Benferhat, Dubois and Prade (1997), Goodman and Nguyen (1988), Schay (1968).

Among the peculiarities (which entail a large flexibility in the management of any kind of uncertainty) of this concept of *coherent* conditional probability versus the usual one, we recall the following ones:

- due to its *direct* assignment as a whole, the knowledge (or the assessment) of the "joint" and "marginal" unconditional probabilities $P(E \wedge H)$ and $P(H)$ is not required;
- the *conditioning* event H (which *must* be a *possible* one) may have *zero probability*, but in the assignment of $P(E|H)$ we are driven by *coherence*, contrary to what is done in those treatments where the relevant conditional probability is given an *arbitrary* value in the case of a conditioning event of zero probability;
- a suitable interpretation of its extreme values 0 and 1 for situations which are different, respectively, from the trivial ones $E \wedge H = \emptyset$ and $H \subseteq E$, leads to a "natural" treatment of the *default reasoning*.

In this talk we deal with the latter aspect.

2 COHERENT CONDITIONAL PROBABILITY

The classic *axioms for a conditional probability* read as follows (given a set $\mathcal{C} = \mathcal{G} \times \mathcal{B}^o$ of conditional events $E|H$ such that $\mathcal{G}$ is a Boolean algebra and $\mathcal{B} \subseteq \mathcal{G}$ is closed with respect to (finite) logical sums, with $\mathcal{B}^o = \mathcal{B} \setminus \{\emptyset\}$):

* Corresponding author.

(i) $P(H|H) = 1$, for every $H \in \mathcal{B}^o$,

(ii) $P(\cdot|H)$ is a (finitely additive) probability on $\mathcal{G}$ for any given $H \in \mathcal{B}^o$,

(iii) $P(E \wedge A|H) = P(E|H)P(A|E \wedge H)$, for any $A, E \in \mathcal{G}$, $H, E \wedge H \in \mathcal{B}^o$.

Conditional probability P has been defined on $\mathcal{G} \times \mathcal{B}^o$; however it is possible, through the concept of *coherence*, to handle also those situations where we need to assess P on an *arbitrary* set $\mathcal{C}$ of conditional events.

Definition 1 - The assessment $P(\cdot|\cdot)$ on $\mathcal{C}$ is *coherent* if there exists $\mathcal{C}' \supset \mathcal{C}$, with $\mathcal{C}' = \mathcal{G} \times \mathcal{B}^o$, such that $P(\cdot|\cdot)$ can be extended from $\mathcal{C}$ to $\mathcal{C}'$ as a *conditional probability.*

A characterization of coherence is given (see, e.g., Coletti and Scozzafava 1996) by the following

Theorem 1 - Let $\mathcal{C}$ be an arbitrary finite family of conditional events $E_1|H_1, \ldots, E_n|H_n$ and $\mathcal{A}_o$ denote the set of atoms A_r generated by the (unconditional) events $E_1, H_1, \ldots, E_n, H_n$. For a real function P on $\mathcal{C}$ the following two statements are equivalent:

(i) P is a *coherent* conditional probability on $\mathcal{C}$;

(ii) there exists (at least) a *class* of probabilities $\{P_0, P_1, \ldots P_k\}$, each probability P_α being defined on a suitable subset $\mathcal{A}_\alpha \subseteq \mathcal{A}_o$, such that for any $E_i|H_i \in \mathcal{C}$ there is a unique P_α with

$$\sum_{\substack{r \\ A_r \subseteq H_i}} P_\alpha(A_r) > 0\,,$$

$$(1) \qquad P(E_i|H_i) = \frac{\sum_{\substack{r \\ A_r \subseteq E_i \wedge H_i}} P_\alpha(A_r)}{\sum_{\substack{r \\ A_r \subseteq H_i}} P_\alpha(A_r)}\,;$$

moreover $\mathcal{A}_{\alpha'} \subset \mathcal{A}_{\alpha''}$ for $\alpha' > \alpha''$ and $P_{\alpha''}(A_r) = 0$ if $A_r \in \mathcal{A}_{\alpha'}$.

According to Theorem 1, a coherent conditional probability gives rise to a suitable class $\{P_o, P_1, \ldots P_k\}$ of "unconditional" probabilities.

Where do the above classes of probabilities come from? Since P is coherent on $\mathcal{C}$, there exists an extension P^* on $\mathcal{G} \times \mathcal{B}^o$, where $\mathcal{G}$ is the algebra generated by the set $\mathcal{A}_o$ of atoms and $\mathcal{B}$ the additive class generated by $H_1, \ldots, H_n$: then, putting $\mathcal{F} = \{\Omega, \emptyset\}$, the restriction of P^* to $\mathcal{A}_o \times \mathcal{F}^o$ satisfies (1) with $\alpha = 0$ for any $E_i|H_i$ such that $P_o(H_i) > 0$. The subset $\mathcal{A}_1 \subset \mathcal{A}_o$ contains only the atoms $A_r \subseteq H_o^1$, the union of H_i's with $P_o(H_i) = 0$ (and so on): we proved (see, e.g., Coletti and Scozzafava 1996, 1999) that, starting from a coherent assessment $P(E_i|H_i)$ on $\mathcal{C}$, a relevant family $\mathcal{P} = \{P_\alpha\}$ can be suitably defined that allows a representation such as (1). Every value $P(E_i|H_i)$ constitutes a constraint in the construction of the probabilities P_α ($\alpha = 0, 1, \ldots$); in fact, given the set $\mathcal{A}_o$ of atoms generated by $E_1, \ldots, E_n, H_1, \ldots, H_n$, and its subsets $\mathcal{A}_\alpha$ (such that $P_\beta(A_r) = 0$ for any $\beta < \alpha$, with $A_r \in \mathcal{A}_\alpha$), each P_α must satisfy the following system (S_α) with unknowns $P_\alpha(A_r) \geq 0$, $A_r \in \mathcal{A}_\alpha$,

$$(S_\alpha) \begin{cases} \sum_{\substack{r \\ A_r \subseteq E_i H_i}} P_\alpha(A_r) = P(E_i|H_i) \sum_{\substack{r \\ A_r \subseteq H_i}} P_\alpha(A_r)\,, \\ [\text{if } P_{\alpha-1}(H_i) = 0] \\ \sum_{\substack{r \\ A_r \subseteq H_0^\alpha}} P_\alpha(A_r) = 1 \end{cases}$$

where $P_{-1}(H_i) = 0$ for all H_i's, and H_o^α denotes, for $\alpha \geq 0$, the union of the H_i's such that $P_{\alpha-1}(H_i) = 0$; so, in particular, $H_o^o = H_o = H_1 \vee \ldots \vee H_n$.

Any class $\{P_\alpha\}$ singled-out by the condition *(ii)* is said *to agree* with the conditional probability P. Notice that in general there are infinite classes of probabilities $\{P_\alpha\}$; in particular we have *only one agreeing class* in the case that $\mathcal{C}$ is a product of Boolean algebras.

A coherent assessment P, defined on a set $\mathcal{C}$ of conditional events, can be extended in a natural way to all the conditional events $E|H$ such that $E \wedge H$ is an element of the algebra $\mathcal{G}$ spanned by the (unconditional) events E_i, H_i, $i = 1, 2, \ldots, n$ taken from the elements of $\mathcal{C}$, and H is an element of the additive class spanned by the H_i's. Obviously, this extension is not unique, since there is no uniqueness in the choice of the class $\{P_\alpha\}$ related to condition *(ii)* of Theorem 1.

In general, we have the following result (see, e.g., Coletti and Scozzafava 1996):

Theorem 2 - If $\mathcal{C}$ is a given family of conditional events and P a corresponding assessment, then there exists a (possibly not unique) coherent extension of P to an arbitrary family $\mathcal{K}$ of conditional events, with $\mathcal{K} \supseteq \mathcal{C}$, *if and only if* P is coherent on $\mathcal{C}$.

Notice that if P is coherent on a family $\mathcal{C}$, it is coherent also on $\mathcal{E} \subseteq \mathcal{C}$.

3 ZERO-LAYERS

Given a class $\mathcal{P} = \{P_\alpha\}_{\alpha=0,1,\ldots,k}$, agreeing with a conditional probability on $\mathcal{C}$, it *naturally induces* the *zero-layer* $\circ(H)$ of an event H, defined as

$$\circ(H) = \beta \quad \text{if } P_\beta(H) > 0\,;$$

if $P_\alpha(H) = 0$ for every $\alpha = 0, 1, \ldots, k$ (obviously, we necessarily have $H \neq H_i$ for every $i = 1, 2, \ldots, n$), then $\circ(H) = k + 1$.

The zero-layer of a conditional event $E|H$ is defined as

$$\circ(E|H) = \circ(E \wedge H) - \circ(H).$$

Obviously, for the certain event Ω and for any event E with positive probability, we have $\circ(\Omega) = \circ(E) = 0$ (so that, if the class contains only an *everywhere positive* probability P_o, there is only one (trivial) zero-layer, *i.e.* $\alpha = 0$), while we put $\circ(\emptyset) = +\infty$. Clearly,

$$\circ(A \vee B) = \min\{\circ(A), \circ(B)\}.$$

Moreover, notice that $P(E|H) > 0$ if and only if $\circ(EH) = \circ(H)$, *i.e.* $\circ(E|H) = 0$.

On the other hand, Spohn (see, for example, Spohn 1994, 1999) considers degrees of plausibility defined via a *ranking* function, that is a map κ that assigns to each *possible* proposition a natural number (its *rank*) such that

(a) either $\kappa(A) = 0$ or $\kappa(A^c) = 0$, or both;

(b) $\kappa(A \vee B) = \min\{\kappa(A), \kappa(B)\}$;

(c) for all $A \wedge B \neq \emptyset$, the conditional rank of B given A is $\kappa(B|A) = \kappa(A \wedge B) - \kappa(A)$.

Ranks represent degrees of "disbelief". For example, A is *not* disbelieved iff $\kappa(A) = 0$, and it is disbelieved iff $\kappa(A) > 0$.

Remark 1 - *Ranking functions are seen by Spohn as a tool to manage* plain belief *and* belief revision, *since he maintains that probability is inadequate for this purpose. In our framework this claim can be challenged (see* Coletti, Scozzafava and Vantaggi 2001*), since our tools for belief revision are* coherent conditional probabilities *and the* **ensuing** *concept of* zero-layers*: it is easy to check that zero-layers have the same formal properties of ranking functions.*

4 COHERENT PROBABILITY AND DEFAULT LOGIC

We recall that in Coletti, Scozzafava and Vantaggi (2001) we showed that a sensible use of events whose probability is 0 (or 1) can be a more general tool in revising beliefs when new information comes to the fore, so that we have been able to challenge the claim contained in Shenoy (1991) that probability is inadequate for revising plain belief. Moreover, as recalled in Section 1, we may deal with the extreme value $P(E|H) = 1$ also for situations which are different from the trivial one $H \subseteq E$.

The aim of this Section is to handle, by means of a *coherent* conditional probability, some aspects of *default reasoning* (see, e.g., Reiter 1980, Russel and Norvig 1995): as it is well-known, a default rule is a sort of weak implication.

First of all, we discuss briefly some aspects of the classic example of Tweety.

The usual *logical implication* (denoted by $\subseteq$) can be anyway useful to express that a penguin (π) is *certainly* a bird (β), i.e.

$$\pi \subseteq \beta,$$

so that

$$P(\beta|\pi) = 1;$$

moreover we know that Tweety (τ) is a penguin (that is, $\tau \subseteq \pi$), and so also this fact can be represented by a conditional probability equal to 1, that is

$$P(\pi|\tau) = 1.$$

But we can express as well the statement "a penguin *usually* does not fly" (we denote by φ^c the contrary of φ, the latter symbol denoting "flying") by writing

$$P(\varphi^c|\pi) = 1.$$

(For simplicity, we have avoided to write down explicit a *proposition* – that is, an event – such as "a given animal is a penguin", using the short-cut "penguin" and the symbol π to denote this event; similar considerations apply to β, τ and φ).

The question "can Tweety fly?" can be faced through an assessment of the conditional probability $P(\varphi|\tau)$, which must be coherent with the already assessed ones: by Theorem 1, it can be shown that *any value* $p \in [0,1]$ is a coherent value for $P(\varphi|\tau)$, so that no conclusion can be reached – *from the given premises* – on Tweety's ability of flying.

In other words, interpreting an equality such as $P(E|H) = 1$ like a *default rule* (denoted by $\longmapsto$), which in particular (when $H \subseteq E$) reduces to the usual implication, we have shown its *nontransitivity*: in fact we have

$$\tau \longmapsto \pi \quad \text{and} \quad \pi \longmapsto \varphi^c,$$

but it *does not* necessarily follow the further default rule $\tau \longmapsto \varphi^c$ (even if we *might* have that $P(\varphi^c|\tau) = 1$, *i.e.* that "Tweety usually does not fly").

Definition 2 - Given a *coherent* conditional probability P on a family $\mathcal{C}$ of conditional events, a *default rule*, denoted by $H \longmapsto E$, is any conditional event $E|H \in \mathcal{C}$ such that $P(E|H) = 1$.

Clearly, any logical implication $A \subseteq B$ (and so also any equality $A = B$) between events can be seen as a (trivial) default rule.

Remark 2 - *By resorting to the systems (S_α) to check the coherence of the assessment $P(E|H) = 1$ (which implies, for the relevant zero-layer, $\circ(E|H) = 0$), a simple computation gives $P_o(E^c \wedge H) = 0$ (notice that the class $\{P_\alpha\}$ has in this case only one element P_o). It follows $\circ(E^c|H) = 1$, so that*

$$\circ(E^c|H) > \circ(E|H)\,.$$

In terms of Spohn's ranking functions (we recall – and underline – *that our zero-layers are – so to say – "incorporated" into a coherent conditional probability, so that* **we do not need** *an "autonomous" definition of ranking!) we could say, when $P(E|H) = 1$, that the disbelief in $E^c|H$ is greater than that in $E|H$. This conclusion* **must not** *be read as $P(E|H) > P(E^c|H)$!*

Given a *set $\Delta \subseteq \mathcal{C}$ of default rules $H_i \longmapsto E_i$*, with $i = 1, ..., n$, we need to check its *consistency*, that is the coherence of the "global" assessment P on $\mathcal{C}$ such that $P(E_i|H_i) = 1$, $i = 1, ..., n$.

We stress that, even if our definition involves a conditional probability, the condition given in the following theorem refers *only* to logical (in the sense of *Boolean* logic) relations.

Theorem 3 - Given a *coherent* conditional probability P on a family $\mathcal{C}$ of conditional events, the following two statements are equivalent:

(i) the set $\Delta \subseteq \mathcal{C}$ of default rules

$$H_i \longmapsto E_i\,,\ i = 1, 2, ..., n\,,$$

represented by the assessment

$$P(E_i|H_i) = 1\,,\ i = 1, 2, ..., n\,,$$

is consistent;

(ii) for every subset

$$\{H_{i_1} \longmapsto E_{i_1}, \ldots, H_{i_s} \longmapsto E_{i_s}\}$$

of Δ, with $s = 1, 2, ..., n$, we have

$$(2) \qquad \bigvee_{k=1}^{s} (E_{i_k} \wedge H_{i_k}) \not\subseteq \bigvee_{k=1}^{s} (E_{i_k}^c \wedge H_{i_k}).$$

Proof - We prove that, assuming the above logical relations (2), coherence of P is compatible with the assessment $P(E_i|H_i) = 1$ $(i = 1, 2, ..., n)$, on Δ.

We resort to the characterization Theorem 1: to begin with, put $P(E_i|H_i) = 1$ $(i = 1, 2, ..., n)$, in the system (S_o). The unconditional probability P_o can be obtained by putting $P_o(A_r) = 0$ for all atoms $A_r \subseteq \bigvee_{j=1}^{n} \left(E_j^c \wedge H_j\right)$, so for any atom $A_k \subseteq E_i \wedge H_i$ which is not contained in $\bigvee_{j=1}^{n} \left(E_j^c \wedge H_j\right)$ – notice that condition (2) ensures that there is such an atom A_k, since $\bigvee_{j=1}^{n}(E_j \wedge H_j) \not\subseteq \bigvee_{j=1}^{n}(E_j^c \wedge H_j)$ – we may put $P_o(A_k) > 0$ in such a way that these numbers sum up to 1, and we put $P_o(A_r) = 0$ for all remaining atoms.

This clearly gives a solution of the first system (S_o). If, for some i, $E_i \wedge H_i \subseteq \bigvee_{j=1}^{n} \left(E_j^c \wedge H_j\right)$, then $P_o(E_i \wedge H_i) = 0$. So we consider the second system (which refers to all H_i such that $P_o(H_i) = 0$), proceeding as above to construct the probability P_1; and so on. Condition (2) ensures that at each step we can give positive probability P_α to (at least) one of the remaining atoms.

Conversely, consider the (coherent) assignment $P(E_i|H_i) = 1$ (for $i = 1, ..., n$). Then, for any index $j \in \{1, 2, \ldots, n\}$ there exists a probability P_α such that $P_\alpha(E_j \wedge H_j) > 0$ and $P_\alpha(E_j^c \wedge H_j) = 0$. Notice that the restriction of P to some conditional events $E_{i_1}|H_{i_1}, ..., E_{i_s}|H_{i_s}$ of Δ is coherent as well.

Let P_o be the first element of an agreeing class, and i_k an index such that $P_o(H_{i_k}) > 0$: then we have $P_o(E_{i_k} \wedge H_{i_k}) > 0$ and $P_o(E_{i_k}^c \wedge H_{i_k}) = 0$. Suppose that $E_{i_k} \wedge H_{i_k} \subseteq \bigvee_{k=1}^{s}(E_{i_k}^c \wedge H_{i_k})$: then $P_o(E_{ik} \wedge H_{ik}) = 0$. This contradiction shows that condition (2) holds.

Definition 3 - A set Δ of default rules *entails* the default rule $H \longmapsto E$ if the *only* coherent value for $P(E|H)$ is 1.

In other words, the rule $H \longmapsto E$ is entailed by Δ (or by a subset of Δ) if every possible extension (cf. Theorem 2) of the probability assessment $P(E_{i_r}|H_{i_r}) = 1$, $r = 1 \ldots s$, assigns the value 1 also to $P(E|H)$.

Going back to the previous example of Tweety, its possible ability (or inability) of flying can be expressed by saying that the default rule $\tau \longmapsto \varphi$ (or $\tau \longmapsto \varphi^c$) *is not entailed* by the premises (the given set Δ).

5 INFERENCE

Several formalisms for default logic have been studied in the relevant literature with the aim of discussing the minimal conditions that an entailment should satisfy. In our framework this "inferential" process is ruled by the following

Theorem 4 - Given a set Δ of consistent default rules, we have:

(Reflexivity)
Δ entails $A \longmapsto A$ for any $A \neq \emptyset$;

(Left Logical Equivalence)
$(A = B), (A \longmapsto C) \in \Delta$ entails $B \longmapsto C$;

(Right Weakening)
$(A \subseteq B), (C \longmapsto A) \in \Delta$ entails $C \longmapsto B$;

(Cut)
$(A \wedge B \longmapsto C), (A \longmapsto B) \in \Delta$ entails $A \longmapsto C$;

(Cautious Monotonicity)
$(A \longmapsto B), (A \longmapsto C) \in \Delta$ entails $A \wedge B \longmapsto C$;

(Equivalence)
$(A \longmapsto B), (B \longmapsto A), (A \longmapsto C) \in \Delta$ entails $B \longmapsto C$;

(And)
$(A \longmapsto B), (A \longmapsto C) \in \Delta$ entails $A \longmapsto B \wedge C$;

(Or)
$(A \longmapsto C), (B \longmapsto C) \in \Delta$ entails $A \vee B \longmapsto C$.

Proof - Reflexivity amounts to $P(A|A) = 1$ for every possible event.

Left Logical Equivalence and *Right weakening* trivially follow from elementary properties of conditional probability.

Cut: from $P(C|A \wedge B) = P(B|A) = 1$ it follows that

$$P(C|A) = P(C|A \wedge B)P(B|A) + P(C|A \wedge B^c)P(B^c|A) =$$
$$= P(C|A \wedge B)P(B|A) = 1.$$

Cautious Monotonicity: since $P(B|A) = P(C|A) = 1$, we have that

$$1 = P(C|A \wedge B)P(B|A) + P(C|A \wedge B^c)P(B^c|A) =$$
$$= P(C|A \wedge B)P(B|A),$$

hence $P(C|A \wedge B) = 1$.

Equivalence: since at least one conditioning event must have positive probability, it follows that A, B, C have positive probability; moreover,

$$P(A \wedge C) = P(A) = P(A \wedge B) = P(B),$$

which implies $P(A \wedge B \wedge C) = P(A) = P(B)$, so $P(C|B) = 1$.

And: since

$$1 \geq P(B \vee C|A) = P(B|A) + P(C|A) - P(B \wedge C|A) =$$
$$= 2 - P(B \wedge C|A),$$

it follows $P(B \wedge C|A) = 1$.

Or: since

$$P(C|A \vee B) =$$
$$= P(C|A)P(A|A \vee B) + P(C|B)P(B|A \vee B) -$$
$$- P(C|A \wedge B)P(A \wedge B|A \vee B) =$$
$$= P(A|A \vee B) + P(B|A \vee B) -$$
$$- P(C|A \wedge B)P(A \wedge B|A \vee B) \geq 1,$$

we get $P(C|A \vee B) = 1$.

We consider now some "unpleasant" properties (cf., e.g., Lehmann and Magidor, 1992), that in fact do not necessarily hold also in our framework:

(Monotonicity)
$(A \subseteq B), (B \longmapsto C) \in \Delta$ entails $A \longmapsto C$

(Transitivity)
$(A \longmapsto B), (B \longmapsto C) \in \Delta$ entails $A \longmapsto C$

(Contraposition)
$(A \longmapsto B) \in \Delta$ entails $B^c \longmapsto A^c$

The previous example about Tweety shows that *Transitivity* can fail.

In the same example, if we add the evaluation $P(\varphi|\beta) = 1$ (that is, a bird *usually* flies) to the initial ones, the assessment is still coherent (even if $P(\varphi|\pi) = 0$ and $\pi \subseteq \beta$), but *Monotonicity* can fail.

Now, consider the conditional probability P defined as follows:

$$P(B|A) = 1, \; P(A^c|B^c) = \frac{1}{4};$$

it is easy to check that it is coherent, and so *Contraposition* can fail.

Many authors (cf., e.g., again Lehmann and Magidor, 1992) claim (and we agree) that the previous unpleasant properties should be replaced by others, that we express below in our own notation and interpretation: we show that these properties hold in our framework. Since a widespread consensus among their "right" formulation is lacking, we will denote them as cs–(Negation Rationality), cs–(Disjunctive Rationality), cs–(Rational Monotonicity), where "cs" stands

for "in a coherent setting". Notice that, given a default rule $H \longmapsto E$, to say $(H \longmapsto E) \notin \Delta$ means that the conditional event $E|H$ belongs to the set $\mathcal{C} \setminus \Delta$.

cs–(Negation Rationality)

If $(A \wedge C \longmapsto B)$, $(A \wedge C^c \longmapsto B) \notin \Delta$
then Δ does not entail $(A \longmapsto B)$

Proof - If $(A \wedge C \longmapsto B)$ and $(A \wedge C^c \longmapsto B)$ do not belong to Δ, i.e. $P(B|A \wedge C) < 1$ and $P(B|A \wedge C^c) < 1$ imply

$$P(B|A) = P(B|A \wedge C)P(C|A) + P(B|A \wedge C^c)P(C^c|A) < \\ < P(C|A) + P(C^c|A) = 1\,.$$

cs–(Disjunctive Rationality)

If $(A \longmapsto C)$, $(B \longmapsto C) \notin \Delta$
then Δ does not entail $(A \vee B \longmapsto C)$

Proof - Starting from the equalities

$$P(C|A \vee B) =$$

$$= P(C|A)P(A|A \vee B) + P(C|A^c \wedge B)P(A^c \wedge B|A \vee B)$$

and

$$P(C|A \vee B) =$$

$$= P(C|B)P(B|A \vee B) + P(C|A \wedge B^c)P(A \wedge B^c|A \vee B),$$

since we have $P(C|A) < 1$ and $P(C|B) < 1$, then $P(C|A \vee B) = 1$ would imply (by the first equality) $P(A|A \vee B) = 0$ and (by the second one) $P(B|A \vee B) = 0$ (contradiction).

cs–(Rational Monotonicity)

If $(A \wedge B \longmapsto C)$, $(A \longmapsto B^c) \notin \Delta$
then Δ does not entail $(A \longmapsto C)$

Proof - If it were $P(C|A) = 1$, i.e.

$$1 = P(C|A \wedge B)P(B|A) + P(C|A \wedge B^c)P(B^c|A)\,,$$

we would get either

$$P(C|A \wedge B) = P(C|A \wedge B^c) = 1$$

or one of the following

$$P(C|A \wedge B) = P(B|A) = 1\,,$$

$$P(C|A \wedge B^c) = P(B^c|A) = 1$$

(contradiction).

In conclusion, let us notice the simplicity of our approach (Occam's razor...!), with respect to other well-known methodologies, such as, e.g. those given by Adams (1975), Benferhat, Dubois and Prade (1997), Goldszmidt and Pearl (1996), Lehmann and Magidor (1992), Schaub (1998).

6 DISCUSSION

Thought-provoking comments of two anonymous reviewers suggested to us to add this further section.

Among coherence–based approaches to default reasoning (in the framework of imprecise probability propagation), that of Gilio (2000) deserves to be mentioned, even if we claim (besides the utmost simplicity of our definitions and results) many important semantic and syntactic differences.

First of all, our framework (see the very beginning of our Introduction) is clearly and rigorously settled: conditional events $E|H$ are **not** 3-valued entities whose third value is looked on as "undetermined" when H is false, but they have been defined instead in a way which entails "automatically" (so-to-say) the axioms of conditional probability, which are those *ruling coherence* (the details, as already recalled in the Introduction, are in Coletti and Scozzafava, 1999).

In other words (**french** words, since we are in France), "tout se tient", while in the aforementioned paper by Gilio a concept such as $E|H$ is interpreted sometimes as a 3-valued entity looked on as "undetermined" when H is false, sometimes as an ordered pair of events, sometimes as a conditional assertion $H \,|\!\sim\, E$ (in the knowledge base).

Moreover, our notions of *consistency* and *entailment* are both different from his: in fact he gives a theorem (without proof) connecting the notion of consistency to that of Adams (1975).

The problem is that we do not understand Adams' framework: in fact he requires probability to be *proper* (i.e., positive) on the given events, but (since the domain of a probability P is an algebra) we need to extend P from the given events to other events (by the way, coherence is nothing but complying with this need). In particular, these "new" events may have zero probability: it follows, according to Adams' definition of *conditional* probability in the case of a conditioning event of zero probability, that we can easily get *incoherent* assessments (see the example below). By the way, in the section **"Some preliminaries"**, Gilio claims "We can frame our approach to the problem of propagating imprecise conditional probability assessments from the *probabilistic logic* point of view, see, e.g., Frisch and Haddawy ...": unfortunately, Frisch and Haddawy definition of conditional probability coincides (for conditioning events which are null) with that of Adams, and so it violates coherence as well!

Not to mention that both Gilio and Adams (and many others: some of them are mentioned at the end of the

previous section) base the concept of consistency on that of *quasi conjunction*, which is a particular conditional event (and our concept of conditional event is different from theirs); moreover we deem that the notion they give of *verifiability* of a conditional event $E|H$, that is $E \wedge H \neq \emptyset$, is too weak – except in the case $H = \Omega$ – to express properly the relevant semantics.

Our discussion can be better illustrated by the following (very simple) example:

Example - Consider two (logically independent) events H_1 and H_2, and put

$$E_1 = H_1 \wedge H_2 \ , \ E_2 = H_1^c \wedge H_2 \, ,$$

$$E_3 = H_1^c \wedge H_2^c \ , \ E = E_2 \ , \ H = H_3 = \Omega \, .$$

Given α, with $0 < \alpha < 1$, the assessment

$$P(E_1|H_1) = P(E_2|H_2) = 1 \ , \ P(E_3|H_3) = \alpha$$

on $\mathcal{C} = \{E_1|H_1, E_2|H_2, E_3|H_3\}$ is coherent; the relevant probabilities of the atoms are

$$P(H_1 \wedge H_2) = P(H_1 \wedge H_2^c) = 0 \, ,$$

$$P(H_1^c \wedge H_2^c) = \alpha \, , \, P(H_1^c \wedge H_2) = 1 - \alpha \, ,$$

so that the set Δ of default rules corresponding to $\{E_1|H_1, E_2|H_2\}$ is consistent.

Does Δ entail $E|H$? A simple check shows that the only coherent assessment for this conditional event is $P(E|H) = 1 - \alpha$. Then the answer is NO, since we require (in the definition of entailment) that 1 is (the only) coherent extension.

On the contrary, according to Gilio characterization of entailment – that is: Δ (our notation) entails $E|H$ iff $P(E^c|H) = 1$ is not coherent – the answer to the previous question is YES, since the only coherent value of this conditional probability is $P(E^c|H) = \alpha$ (see the above computation).

For any $\epsilon > 0$, consider now the assessment

$$P(E_1|H_1) = 1 \, , \, P(E_2|H_2) = 1 - \epsilon \, ,$$

so that $\{E_1|H_1, E_2|H_2\}$ is consistent according to Adams, as can be easily checked giving the atoms the probabilities

$$P(H_1 \wedge H_2) = \epsilon \, , \, P(H_1 \wedge H_2^c) = 0 \, ,$$

$$P(H_1^c \wedge H_2^c) = 0 \, , \, P(H_1^c \wedge H_2) = 1 - \epsilon \, ,$$

(notice that the assessment is proper). But for any event $A \subset H_1 \wedge H_2^c$ we can extend P, according to his definition of conditional probability, as

$$P(A|H_1 \wedge H_2^c) = P(A^c|H_1 \wedge H_2^c) = 1 \, ,$$

which is **not** coherent!

Finally, there is no mention in Gilio's paper of Negation Rationality, Disjunctive Rationality, and Rational Monotonicity (and, according to one of the reviewers, these properties do not hold "in default reasoning under coherent probabilities", while in our setting they have been proved at the end of Section 5).

Acknowledgements

We thank an anonymous referee for signaling us a slight mistake in the proof of Disjunctive Rationality.

References

E. Adams (1975). *The Logic of Conditionals*, Dordrecht: Reidel.

S. Benferhat, D. Dubois, and H. Prade (1997). Nonmonotonic Reasoning, Conditional Objects and Possibility Theory. *Artificial Intelligence* **92**:259–276

G. Coletti and R. Scozzafava (1996). Characterization of Coherent Conditional Probabilities as a Tool for their Assessment and Extension. *International Journal of Uncertainty, Fuzziness and Knowledge-Based System* **4**:103–127.

G. Coletti and R. Scozzafava (1999). Conditioning and Inference in Intelligent Systems. *Soft Computing* **3**:118–130.

G. Coletti, R. Scozzafava, and B. Vantaggi (2001). Probabilistic Reasoning as a General Unifying Tool. In S.Benferhat and P. Besnard (eds.), *Lectures Notes in Computer Science* LNAI 2143 (ECSQUARU 2001), 120–131.

G. Coletti, R. Scozzafava, and B. Vantaggi (2002). Coherent Conditional Probability as a Tool for Default Reasoning. *Proc. IPMU 2002*, Annecy (France), to appear.

B. de Finetti (1949). Sull'impostazione assiomatica del calcolo delle probabilità. *Annali Univ. Trieste* **19**:3–55. (Engl. transl.: Ch.5 In *Probability, Induction, Statistics*, London: Wiley, 1972).

L.E. Dubins (1975). Finitely Additive Conditional Probabilities, Conglomerability and Disintegration. *Annals of Probability* **3**:89–99.

A. Gilio (2000). Precise propagation of upper and lower probability bounds in system P. In *Proc. 8th Int. Workshop on Non-monotonic Reasoning*, "Uncertainty Frameworks in Non-Monotonic Reasoning", Breckenridge (USA).

M. Goldszmidt and J. Pearl (1996) Qualitative proba-

bility for default reasoning, belief revision and causal modeling. *Artificial Intelligence* **84**:57–112.

I.R. Goodman and H.T. Nguyen (1988). Conditional objects and the modeling of uncertainties. In M. Gupta and T. Yamakawa (eds.), *Fuzzy Computing*, 119-138. Amsterdam: North Holland.

P.H. Krauss (1968). Representation of Conditional Probability Measures on Boolean Algebras. *Acta Math. Acad. Scient. Hungar* **19**:229–241.

D. Lehmann and M. Magidor (1992). What does a conditional knowledge base entail? *Artificial Intelligence* **55**:1–60.

R. Reiter (1980). A Logic for Default Reasoning. *Artificial Intelligence* **13**(1-2):81–132.

A. Rényi (1956). On Conditional Probability Spaces Generated by a Dimensionally Ordered Set of Measures. *Theory of Probability and its Applications* **1**:61–71.

S.J. Russel and P. Norvig (1995). *Artificial Intelligence. A Modern Approach.* New Jersey: Prentice-Hall.

T. Schaub (1998). The Family of Default Logics. In D.M. Gabbay and P. Smets (eds.), *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, Vol.2, 77–133, Dordrecht: Kluwer.

G. Schay (1968). An Algebra of Conditional Events. *Journal of Mathematical Analysis and Applications* **24**:334-344.

P.P. Shenoy (1991). On Spohn's Rule for Revision of Beliefs. *International Journal of Approximate Reasoning* **5**:149–181.

W. Spohn (1994). On the Properties of Conditional Independence. In P. Humphreys, P. Suppes (eds), *Scientific Philosopher* **1**, Probability and Probabilistic Causality, 173–194. Dordrecht: Kluwer.

W. Spohn (1999). Ranking Functions, AGM Style. *Research Group "Logic in Philosophy"*, Preprint 28.

Default Epistemology
Rankings, Reasoning and Revision

A very preliminary account

Emil Weydert
Max-Planck-Institute for Computer Science
emil@mpi-sb.mpg.de

Abstract

There are well-known links between default reasoning and belief revision. However, previous work has been mainly concerned with the relation between default conditional logics and single-step revision. But this is only the tip of the iceberg. In this paper, we want to go beyond these initial considerations and explore how to derive iterated revision strategies from nonmonotonic default inference. We first take a general perspective before we focus on some instances of cross-fertilization within extensions of Spohn's ranking paradigm.

1 Introduction

It is well-known that there are different formal links between default reasoning and belief revision. However, previous work has been mainly concerned with the connection between monotonic default conditional logics (object-level), or abstract defeasible consequence relations (meta-level), and single-step revision in the tradition of AGM, e.g. exploiting epistemic entrenchment orderings. But this is only the tip of the iceberg. What is much less well understood – characterized by many competing approaches and a lack of substantial general principles – are default inference notions (nonmonotonic reasoning exploiting defaults), iterated revision strategies, and especially their mutual relationships. In this paper, we want to explore some of these links to develop new revision methods, especially in the context of a quasi-probabilistic ranking semantics for default knowledge.

The idea is to use well-behaved powerful default formalisms – either directly or by adapting their semantic mechanisms – to specify dynamic epistemic models. Starting from a general perspective on belief revision and default reasoning, we try to achieve this in particular for default inference relations based on the ranking construction paradigm, like system JZ [Wey 98] or JLX-entailment [Wey 01]. Here we are primarily interested in extending Spohn-style ranking revision [Spohn 88] for single propositions to multiple default conditional evidence. Roughly speaking, the basic approach is to characterize epistemic states by default knowledge bases – using a suitable default formalism, e.g. JZ or JLX, to determine the beliefs – and to realize revision by simply adding the new default evidence restricted to some generic proposition X representing the raw unspecified input. For suitable default inference notions, in particular those based on minimal canonical ranking constructions, the resulting default revision strategies are very powerful. An alternative approach is to generalize the existing ranking construction procedures from default reasoning to non-uniform priors. These more sophisticated semantic accounts allow us to overcome problems of standard default revision.

2 Ranking framework

Let $\mathcal{L}$ be a language of propositional logic with a countably infinite set of variables Var, $\mathcal{W} = 2^{Var}$ the corresponding set of valuations, or $\mathcal{L}$-structures, and $\models \subseteq \mathcal{W} \times \mathcal{L}$ the standard satisfaction relation. $(\mathcal{L}, \models)$ will be our background logic. For $\Sigma \cup \{\varphi\} \subseteq \mathcal{L}$, we set $[\varphi] = \{w \in \mathcal{W} \mid w \models \varphi\}$, $[\Sigma] = \bigcap\{[\psi] \mid \psi \in \Sigma\}$ and $\Sigma \vdash \varphi$ iff $[\Sigma] \subseteq [\varphi]$. This definition allows us to write $\neg[\varphi]$ for $[\neg\varphi]$ and $[\varphi] \wedge [\psi]$ for $[\varphi \wedge \psi]$. Let $\mathcal{B} = \{[\varphi] \mid \varphi \in \mathcal{L}\}$ be the corresponding boolean propositional system and $\mathcal{L}(\Rightarrow) = \{\varphi \Rightarrow \psi \mid \varphi, \psi \in \mathcal{L}\}$ the set of default conditionals over $\mathcal{L}$.

In this paper, we are going to exploit the ranking measure framework [Weydert 94], which offers a powerful and intuitively appealing semantics for default reasoning and belief revision. Ranking measures are coarse-

grained, quasi-probabilistic valuations which quantify the (relative) degree of surprise or implausibility of propositions. More specifically, their ranking scales carry an additive structure supporting a suitable notion of independency (e.g. for plausibility networks). They generalize Spohn's discrete-valued κ-functions, which he used to model iterated revision of graded full belief [Spohn 88, 90] and Pearl to handle default statements [Pearl 90], as well as Dubois and Prade's dense-valued possibility functions [BCDLP 93].

Definition 2.1 (Ranking measures)
$R : \mathcal{B} \to \mathcal{V}$ is called a ranking measure iff for $A, B \in \mathcal{B}$,

- *$\mathcal{V} = (V, 0, \infty, +, <)$ is the positive half of an ordered commutative group, extended by a maximal and absorptive element ∞ (i.e. for all $x \in G$, $x \leq x + \infty = \infty + x = \infty$),*
- *$R(\mathcal{W}) = 0$ and $R(\emptyset) = \infty$,*
- *$R(A \cup B) = \min\{R(A), R(B)\}$,*
- *$R(B) = \infty$ if $B \subseteq \bigcup\{A \in \mathcal{B} \mid R(A) = \infty\}$.*

The conditional measure $R(.|.) : \mathcal{B} \times \mathcal{B} \to \mathcal{V}$ associated with R is given by

- *$R(B|A) = R(B \cap A) - R(A)$ if $R(A) \neq \infty$ and $R(B|A) = \infty$ otherwise.*

R is called standard iff $\mathcal{V} = ([0, \infty], \infty, 0, +, <)$ is the additive structure of the positive reals with ∞. R_0 with $R_0(A) = 0$ for all $A \neq \emptyset$ is the uniform ranking measure.

Ranking measures are meant to bridge the gap between the powerful probabilistic and the practical preferential perspective. Their strength and usefulness may be explained by the order of magnitude interpretation, which links each (standard) rank $R(A) = r$ to a non-standard probability $P(A) = \varepsilon^r$, where $0 < \varepsilon$ is a fixed infinitesimal.

Definition 2.2 (Ranking measure semantics)
The satisfaction relation $\models_{rk}$ between ranking measures and $\mathcal{L}(\Rightarrow)$ is given by $R \models_{rk} \varphi \Rightarrow \psi$ iff

- *$R([\varphi \wedge \psi]) + 1 \leq R([\varphi \wedge \neg\psi])$ or $R([\varphi]) = \infty$.*

Let $\mathcal{KP}(\Delta) = \{R \in \mathcal{KP} \mid R \models_{rk} \Delta$ be the set of standard $\kappa\pi$-models of $\Delta \subseteq \mathcal{L}(\Rightarrow)$, e.g. $\mathcal{KP} = \mathcal{KP}(\emptyset)$.

For our purposes, Spohn's discrete ranking scale is not fine-grained enough.

Definition 2.3 ($\kappa\pi$-measures)
A ranking measure $R : \mathcal{B} \to \mathcal{V}$ is a $\kappa\pi$-measure iff

- *$\mathcal{V}$ is divisible, i.e. for all $x \in V$ and $0 < n$, there is an $y \in V$ with $x = y + \ldots + y$ (n times).*

Let $\mathcal{KP}$ be the set of all standard $\kappa\pi$-measures.

Because each $\mathcal{V}$ admits a divisible extension and for our purposes, we only have to deal with finite constraints, w.l.o.g., we may focus on standard $\kappa\pi$-measures.

3 Revision framework

We interpret belief revision as a process by which an intelligent agent accommodates its beliefs about the actual history of the world – not only about the current state – in the light of new evidence. Understood in this way, belief revision subsumes belief update in a changing world. On the formal level, we assume that a dynamic epistemic framework should include the following ingredients:

- a collection of epistemic states $\mathcal{E}$,
- a set of generic prior states $\mathcal{E}_0 \subseteq \mathcal{E}$,
- a boolean propositional system $\mathcal{B}$, with $\emptyset, \mathcal{W} \in \mathcal{B} \subseteq 2^{\mathcal{W}}$,
- a set of epistemic specification functions, e.g. a belief set function $Bel : \mathcal{E} \to 2^{\mathcal{B}}$, where $Bel[e]$ is a filter on $\mathcal{B}$,
- a set of admissible epistemic inputs $\mathcal{I}$,
- an input evaluation function $ev : \mathcal{E} \times \mathcal{I} \to 2^{\mathcal{E}}$,
- a revision function $\star : \mathcal{E} \times \mathcal{I} \to \mathcal{E}$, with $e \star i \in ev(e, i)$.

This models a two-phase process which first interprets the cognitive input as a constraint $ev(e, i)$ over epistemic states before, in the context of e, it picks up the most suitable revised state $e \star i$. We illustrate these concepts by describing some ranking revision accounts going back to Spohn's approach.

- $\mathcal{E} = \mathcal{KP}$,
- $\mathcal{E}_0 = \{R_0\}$,
- $\mathcal{B} = \{[\varphi] \mid \varphi \in \mathcal{L}\}$,
- $\mathcal{I} = \mathcal{L}$,

- $Bel[R] = \{A \in \mathcal{B} \mid R(\neg A) > 0\}$,
  $Know[R] = \{A \in \mathcal{B} \mid R(\neg A) = \infty\}$,
- $ev(R, \varphi) = \{R^* \in \mathcal{E} \mid R^*([\varphi]) \geq 1\}$ if $\neq \emptyset$,
  $ev(R, \varphi) = \{R\}$ if $\{R^* \in \mathcal{E} \mid R^*([\varphi]) \geq 1\} = \emptyset$.

Here, the epistemic states are the standard $\kappa\pi$-measures, the generic prior is the uniform $\kappa\pi$-measure, the propositional system is the ranking domain, the inputs are the $\mathcal{L}$-formulas, belief/knowledge collects those propositions whose complement is surprising/impossible, and the minimal default belief strength attributed to incoming evidence is at least 1. Each evidence formula defines a ranking constraint to be satisfied – if possible – by the revised state $R\star\varphi$. To specify the revision function for evidence not strictly known to be false, we use the shifting function, a ranking transformation inspired from Spohn's conditionalization procedure. We define shifting over $\mathcal{KP}^-$, the set of "non-normalized $\kappa\pi$-measures" $S : \mathcal{B} \to [0, \infty]$, i.e. of maps which verify all the ranking conditions except $S(\mathcal{W}) = 0$.

Definition 3.1 (Shifting)
The shifting function is the unique map $+ : \mathcal{KP}^- \times]-\infty, \infty] \times \mathcal{B} \to \mathcal{KP}^-$ *s.t. for all* $A, B \in \mathcal{B}$,

- $(S + \infty A)(B) = S(B|\neg A)$, *and if* $r \neq \infty$,
- $(S + rA)(B|A) = S(B|A)$,
- $(S + rA)(B|\neg A) = S(B|\neg A)$,
- $(S + rA)(A) = max\{S(A) + r, 0\}$,
- $(S + rA)(\neg A) = S(\neg A)$.

In other words, we obtain $S + rA$ by uniformly shifting the A-worlds by r – upwards if $r \geq 0$, downwards if $r \leq 0$ – but at most until one of them touches the bottom 0. We may now specify three major revision strategies in the tradition of Spohn. If $R([\varphi]) = \infty$, we reject the input and stipulate $R\star\varphi = R$. So, let us assume $R([\varphi]) \neq \infty$.

Classical Spohn revision:

- $R \star_{sp} \varphi = R - R([\varphi])[\varphi] + 1[\neg\varphi]$

Rigid Spohn revision:

- $R \star_{rg} \varphi = R - R([\varphi])[\varphi] + (R([\neg\varphi]) + 1)[\neg\varphi]$

Minimal Spohn revision:

- $R \star_{jr} \varphi = R - R([\varphi])[\varphi] + max\{R([\neg\varphi]), 1\}[\neg\varphi]$

If $R([\neg\varphi]) \leq 1$, then we have $(R \star_{sp} \varphi)([\neg\varphi]) = 1$, $(R\star_{rg}\varphi)([\neg\varphi]) = 1 + R([\neg\varphi])$, and $(R\star_{jr}\varphi)([\neg\varphi]) = 1$. If $R([\neg\varphi]) > 1$, we still get $(R \star_{sp} \varphi)([\neg\varphi]) = 1$ as well as $(R \star_{rg} \varphi)([\neg\varphi]) = 1 + R([\neg\varphi])$, but also $(R \star_{jr} \varphi)([\neg\varphi]) = R([\neg\varphi])$. That is, only $\star_{jr}$ preserves the ranking state if the input constraint already holds.

Classical Spohn revision, also known as $A,1$-conditionalization, ignores potentially useful information, namely the original belief strength of A. Rigid Spohn revision, the approach discussed in [DP 97], which is also popular in the possibilistic literature, blindly adds belief strength to A. Minimal Spohn revision, which we have introduced in the context of default reasoning [Wey 98], requires the least shifting efforts and seems to be the most sensible implementation of the minimal change principle in this context. In addition, it fits the ranking instantiation of cross-entropy minimization, a well-behaved probabilistic revision procedure minimizing information loss [SJ 80]. Accordingly, $\star_{jr}$ is of particular interest and will be our reference formalism when looking for more general approaches.

4 Default reasoning

The role of standard default inference relations $\mid\sim$ is to derive plausible consequences from specific facts, e.g. $\Sigma \subseteq \mathcal{L}$, and defaults expressing generic plausibility statements, e.g. $\Delta \subseteq \mathcal{L}(\Rightarrow)$. we set $\Sigma \mid\sim_\Delta \psi$ iff $\Sigma \cup \Delta \mid\sim \psi$. In the following, we assume that $\mid\sim$ verifies some very reasonable basic principles. The first one is based on left logical equivalence, right weakening, and supraclassicality [KLM 90].

Definition 4.1 (Logicality)
A default inference relation $\mid\sim$ *verifies logicality iff for all finite* $\Sigma \subseteq \mathcal{L}, \Delta = \{\varphi_i \Rightarrow \psi_i \mid i \leq n\}, \Delta' = \{\varphi'_i \Rightarrow \psi'_i \mid i \leq n\} \subseteq \mathcal{L}(\Rightarrow)$,

- $\{\psi \mid \Sigma \cup \Delta \mid\sim \psi\}$ *is closed under classical logic,*
- $\Sigma \vdash \psi$ *implies* $\Sigma \cup \Delta \mid\sim \psi$,
- $\Sigma \dashv\vdash \Sigma'$, $\varphi_i \dashv\vdash \psi_i$ *and* $\varphi'_i \dashv\vdash \psi'_i$ $(i \leq n)$ *implies* $\Sigma \cup \Delta \mid\sim \psi$ *iff* $\Sigma' \cup \Delta' \mid\sim \psi'$.

The second one imposes a fundamental link between the default conditional and the default inference notion.

Definition 4.2 (Defeasible modus ponens)
Let $\varphi, \psi \in \mathcal{L}$ *and* $\Delta \subseteq \mathcal{L}(\Rightarrow)$.

- $\{\varphi\} \cup \{\varphi \Rightarrow \psi\} \cup \Delta \mid\sim \psi$.

The third one tells us that only the semantical or logical structure of the propositional components should be relevant.

Definition 4.3 (Boolean invariance)
Let $\Sigma \cup \{\psi\} \subseteq \mathcal{L}$, $\Delta \subseteq \mathcal{L}(\Rightarrow)$ *be finite,* $\pi' : \mathcal{B} \to \mathcal{B}$ *a boolean isomorphism, and* $\pi : \mathcal{L} \to \mathcal{L}$ *a map with* $\pi'([\varphi]) = [\pi(\varphi)]$. *Furthermore, let* $\Delta^\pi = \{\pi(\varphi) \Rightarrow \pi(\psi) \mid \varphi \Rightarrow \psi \in \Delta\}$ *and* $\Sigma^\pi = \{\pi(\varphi) \mid \varphi \in \Sigma\}$. *Then the default inference* $\mid\sim$ *verifies boolean invariance iff*

- $\Sigma \cup \Delta \mid\sim \psi$ *iff* $\Sigma^\pi \cup \Delta^\pi \mid\sim \pi(\psi)$.

Within the ranking measure framework, we may use ranking choice functions to specify default formalisms. These maps associate with each finite $\Delta \subseteq \mathcal{L}(\Rightarrow)$ a set of distinguished $\kappa\pi$-models $\mathcal{F}(\Delta) \subseteq \mathcal{KP}(\Delta)$, the only condition being consistency preservation (CP) and local semanticality (LS).

Definition 4.4 (Ranking choice)
$\mathcal{F}$ *is called a ranking choice function iff for each finite* $\Delta \subseteq \mathcal{L}(\Rightarrow)$,

1. $\mathcal{F}(\Delta) \subseteq \mathcal{KP}(\Delta)$, *(Choice)*
2. $\mathcal{KP}(\Delta) \neq \emptyset$ *implies* $\mathcal{F}(\Delta) \neq \emptyset$ *(CP),*
3. $\mathcal{F}(\Delta) = \mathcal{F}'(\{\mathcal{KP}(\{\delta\}) \mid \delta \in \Delta\}$ *(LS).*

LS enforces representation independence on the level of defaults, which is a conditio sine qua non for a semantic framework. In particular, it entails

$$\mathcal{F}(\{\varphi \Rightarrow \psi\}) = \mathcal{F}(\{\varphi \Rightarrow \varphi \wedge \psi\}),$$

whereas it does not support

$$\mathcal{F}(\{\mathbf{T} \Rightarrow \varphi, \mathbf{T} \Rightarrow \psi\}) = \mathcal{F}(\{\mathbf{T} \Rightarrow \varphi \wedge \psi\}).$$

A reasonable and natural strengthening is strong semanticality (SS), which asks for invariance under boolean isomorphisms of $\mathcal{B}$. Local semanticality, on the other hand, just implements invariance w.r.t. the identity on $\mathcal{B}$.

Definition 4.5 (Strong semanticality)
A ranking choice function $\mathcal{F}$ *is strongly semantical iff for each boolean isomorphism* π *over* $\mathcal{B}$, *if* $\{([\varphi \wedge \neg\psi], [\varphi \wedge \psi]) \mid \varphi \Rightarrow \psi \in \Delta'\} = \{(\pi([\varphi \wedge \neg\psi]), \pi([\varphi \wedge \psi])) \mid \varphi \Rightarrow \psi \in \Delta\}$, *then*

$$R \in \mathcal{F}(\Delta) \text{ iff } R \circ \pi^{-1} \in \mathcal{F}(\Delta').$$

Because we may want to exploit the internal structure of a default knowledge base, which is a necessary prerequisite for some inheritance desiderata, we do not require full conditional logical invariance. That is, $\mathcal{KP}(\Delta) = \mathcal{KP}(\Delta')$ does not always entail $\mathcal{F}(\Delta) = \mathcal{F}(\Delta')$. Each ranking choice $\mathcal{F}$ now defines a semantic-based default inference notion $\mid\sim^{\mathcal{F}}$.

Definition 4.6 (Default entailment)
Let $\mathcal{F}$ *be a ranking choice function and* $\sigma = \bigwedge \Sigma$.

$$\Sigma \mid\sim^{\mathcal{F}}_{\Delta} \psi \text{ iff for all } R \in \mathcal{F}(\Delta),\ R([\neg\psi]|[\sigma]) > 0.$$

This approach is very general. For instance, an inference notion $\mid\sim$ is preferential (for arbitrary finite Δ) [KLM 90] iff there is a ranking choice function $\mathcal{F}$ with $\mid\sim = \mid\sim^{\mathcal{F}}$. The weakest instantiation, where $\mathcal{F} = \mathcal{KP}$, corresponds to system P. It follows from the definition that logicality and DMP are always valid for $\mid\sim^{\mathcal{F}}$. With strong semanticality, boolean invariance holds as well. To get further nice properties, we need additional constraints for $\mathcal{F}$. One of the simplest and most powerful proposals is the ranking construction paradigm [Wey 95, 96, 98]. It is based on the idea that the preferred models of $\Delta = \{\varphi_i \Rightarrow \psi_i \mid i \leq n\}$ should be constructible over Δ, that is accessible by iterated parametrized Spohn revision (i.e. A, α-conditioning) with the material implications $\varphi_i \to \psi_i$.

Definition 4.7 (Ranking constructibility)
$R \in \mathcal{KP}$ *is constructible over* $\Delta = \{\varphi_i \Rightarrow \psi_i \mid i \leq n\}$ *iff there are* $0 \leq a_i$ *with*

$$R = R_0 + a_0[\varphi_0 \wedge \neg\psi_0] + \ldots + a_n[\varphi_n \wedge \neg\psi_n].$$

Let $Constr(\Delta)$ *be the set of* R *constructible over* Δ.

In a finite context, this amounts to impose a – possibly vanishing – penalty for the violation of each default and to construct a ranking measure by adding the penalties for each world and considering the minima for each proposition. This gives us a whole class of well-behaved default inference relations.

Definition 4.8 (J-admissible inference)
A default entailment $\mid\sim$ *is called J-admissible iff there is a strongly semantical ranking choice function* $\mathcal{F}$ *with* $\mid\sim = \mid\sim^{\mathcal{F}}$ *and* $\mathcal{F}(\Delta) \subseteq \mathcal{KP}(\Delta) \cap Constr(\Delta)$.

J-admissible default reasoning satisfies most inheritance desiderata as well as the representation independence and irrelevance principles for nonmonotonic inference. Of particular interest are those inference notions based on a minimal canonical ranking construction.

Definition 4.9 (JMR-admissible inference)
A J-admissible inference relation $\mid\sim^{\mathcal{F}}$ *is called JMR-admissible iff for each finite consistent* Δ, *there is*

$R_\Delta \in Min_\leq(\mathcal{KP}(\Delta))$ *with* $\mathcal{F}(\Delta) = \{R^\mathcal{F}_\Delta\}$, *with* $\leq$ *being pointwise comparison.*

JMR-admissible inference notions $\mid\!\sim^\mathcal{F}$ combine normality maximization, e.g. as implemented in system Z, with constructibility, i.e. J-admissibility. Obviously, each $\mid\!\sim^\mathcal{F}_\Delta$ is rational and, for consistent $\varphi \wedge \psi$, we have $R^\mathcal{F}_{\{\varphi \Rightarrow \psi\}} = R_0 + 1[\varphi \wedge \neg\psi]$. Examples are system JZ [Wey 98] and JLX/JLZ-entailment [Wey 01], which admit probabilistic justifications and are of particular interest.

5 Default revision

We are now going to investigate how and under which conditions we may exploit a suitable default inference relation $\mid\!\sim$ to specify a revision strategy $\star$ for default evidence, i.e. epistemic inputs consisting of finite default sets Δ. We start with a general discussion of default revision frameworks $(\mathcal{E}, \mathcal{E}_0, \mathcal{B}, Bel, \mathcal{I}, ev, \star)$ in the context of a some default entailment notion $\mid\!\sim$, which will not be limited to ranking-based formalisms. One of the basic ideas is to use $\mid\!\sim$ to determine the belief set $Bel[e]$ of e. This assumes that we associate with each epistemic state e a default knowledge base $\Sigma_e \cup \Delta_e$ from which we then defeasibly infer the beliefs in e. As usual, the boolean propositional system is $\mathcal{B}$.

- $Bel[\Sigma_e \cup \Delta_e] = \{[\varphi] \in \mathcal{B} \mid \Sigma_e \cup \Delta_e \mid\!\sim \varphi\}$.

Our definition of epistemic frameworks then requires that $\mid\!\sim_\Delta$ should verify right weakening and right conjunction. Observe that the same state may support different belief set notions because the default consequences depend on $\mid\!\sim$. In the simplest scenario, we may identify e with $\Sigma_e \cup \Delta_e$ and characterize epistemic states by default knowledge bases.

- $\mathcal{E} = \{\Sigma \cup \Delta \mid \Sigma \cup \Delta \subseteq \mathcal{L} \cup \mathcal{L}(\Rightarrow)$ *finite*$\}$.

The maximally ignorant prior state is therefore

- $\mathcal{E}_0 = \{\emptyset\}$.

In default revision, the finite default bases constitute the admissible evidence.

- $\mathcal{I} = \{\Delta \mid \Delta \subseteq \mathcal{L}(\Rightarrow)$ *finite*$\}$.

The inputs Δ themselves are best seen as initial compilations of evidence, translating e.g. the external – i.e. usually defeasible – simultaneous observations of some $\varphi_0, \ldots, \varphi_n$ into the default set $\{\mathbf{T} \Rightarrow \varphi_i \mid i \leq n\}$, or the internal – i.e. usually unquestionable – parallel insights that $\psi_0, \ldots, \psi_n$ into the strict default set $\{\neg\psi_i \Rightarrow \mathbf{F} \mid i \leq n\}$. To specify the proper input evaluation function, we have to clarify the intended impact, i.e. the meaning of the default evidence. Let $\Sigma \cup \Delta$ be the prior state and Δ_0 be some incoming evidence. We assume that Δ_0 is coherent with $\Sigma \cup \Delta$.

Definition 5.1 (Coherence)
$\Sigma' \cup \Delta'$ *is said to be coherent with* $\Sigma \cup \Delta$ *iff there is* $R \in \mathcal{KP}(\Delta)$ *with* $R([\eta]|[\sigma \wedge \sigma']) \neq \infty$, *where* $\eta = \bigwedge \Delta^\rightarrow$ *with* $\Delta^\rightarrow = \{\varphi \rightarrow \psi \mid \varphi \Rightarrow \psi \in \Delta\}$, $\sigma = \bigwedge \Sigma$, *and* $\sigma' = \bigwedge \Sigma'$.

Then, following the tradition of the qualified AGM success postulate (which respects epistemic necessities), we want to ensure that the inference behaviour of the revised state $\Sigma^* \cup \Delta^*$ reflects the defaults in Δ_0, at least as long as Δ_0 is not rejected because of incoherence.

Definition 5.2 (Default success)
If $\Sigma, \Delta, \Delta_0, \Sigma^*, \Delta^*$ *are as above and* Δ_0 *is coherent with* $\Sigma \cup \Delta$, *then*

- $\varphi \Rightarrow \psi \in \Delta_0$ *implies* $\{\varphi\} \cup \Sigma^* \cup \Delta^* \mid\!\sim \psi$.

In particular, if we have $\mathbf{T} \Rightarrow \psi \in \Delta_0$ and coherence holds, we expect $\psi \in Bel[\Sigma^* \cup \Delta^*]$. We may consider default success to be the minimal requirement for revision candidates $\Sigma^* \cup \Delta^*$. The straightforward choice for the input evaluation function is therefore

- $ev(\Sigma \cup \Delta, \Delta) = \{\Sigma^* \cup \Delta^* \subseteq \mathcal{L} \cup \mathcal{L}(\Rightarrow) \mid$
  $\varphi \Rightarrow \psi \in \Delta$ implies $\{\varphi\} \cup \Sigma^* \cup \Delta^* \mid\!\sim \psi\}$.

How may we now specify $\Sigma^* \cup \Delta^* = (\Sigma \cup \Delta) \star \Delta_0$? Obviously, the naive approach of setting $\Delta^* = \Delta \cup \Delta_0$ doesn't work because it runs into the same problems as standard conditionalization, namely by trivializing any iterated revision sequence including conflicting evidence. However, the idea to generate $\Sigma^* \cup \Delta^*$ by simply expanding $\Sigma \cup \Delta$ with a suitable supplement $\Sigma^+ \cup \Delta^+$ exploiting Δ_0 looks rather attractive. In the tradition of Rott's perspective [Rott 96], this amounts to implement a vertical revision through expansion philosophy. So, given Σ, Δ and Δ_0, what would be an appropriate choice for $\Sigma^+ \cup \Delta^+$?

The expansion procedure only makes sense if Σ^+ consists of new hard or unrevisable knowledge. The natural candidate here would be the bare fact that a raw epistemic input is received at a given moment. For practical purposes, we may split Var into concrete world-description variables encoding objective

information and abstract input-description variables $\{X_i \mid i \in Nat\}$ representing – a priori – the unreflected basic data experience at some moment i, whatever its exact nature. Accordingly, we may distinguish $\mathcal{L}_{obj}$ and $\mathcal{L}_{bas}$. The X_i start as virtual statements, logically independent from all the boolean combinations of sentences appearing in $\Sigma \cup \Delta \cup \Delta_0$, and provide a reference context for the new defaults in Δ^+, which describe the content of the incoming information mainly through the relation of X_i to objective variables. Our first step is therefore to set $\Sigma^+ = \{X_i\}$ which we add to the old virtual input knowledge and possibly some absolute prior constraints in Σ.

To enforce default success within this framework, the most promising strategy is to exploit the defeasible modus ponens (DMP). The idea is to relativize the conditional input Δ_0 to $\Sigma^* = \Sigma \cup \Sigma^+ = \Sigma \cup \{X_i\}$, where X_i is the new virtual input statement. Let $\Phi^\alpha = \{\varphi \wedge \alpha \Rightarrow \psi \mid \varphi \Rightarrow \psi \in \Phi\}$ and $\sigma^* = \bigwedge \Sigma \wedge X_i$. If $\Sigma^* \cup \Delta_0^{\sigma^*} \cup \Delta$ is consistent, i.e. $\not\vdash \mathbf{F}$, we set $\Delta^+ = \Delta_0^{\sigma^*}$. If not, we stipulate $\Delta^+ = \emptyset$. Note that it is not enough to relativize Δ_0 to X_i because this would conflict with iterative success. Together with DMP, this constellation guarantees default success. Using this default projection strategy, we may now specify the revision function. Let $X_{new} \in \mathcal{L}_{bas}$ be the variable with the smallest index not occurring in $\Sigma \cup \Delta \cup \Delta_0$.

Definition 5.3 (Default revision)

- $(\Sigma \cup \Delta) \star \Delta_0 = \Sigma \cup \{X_{new}\} \cup \Delta \cup \Delta_0^{\sigma \wedge X_{new}}$.

In this way, we can associate with each default inference notion $\mid\!\sim$ a corresponding default revision framework. Whether it is reasonable depends of course on the properties of $\mid\!\sim$. In fact, it turns out that the requirements for $\mid\!\sim$ are pretty high. For instance, our strong form of the defeasible modus ponens corresponds to an instance of the specificity principle. In particular, it doesn't hold for Reiter's default logic. But fortunately, we also have many default formalisms verifying specificity. But there is another restriction which has an even bigger impact. Consider the preservation principle, which we may express by simulating propositional revision with φ by default revision with $\mathbf{T} \Rightarrow \varphi$.

- $\neg\varphi \notin Bel[\Sigma \cup \Delta]$ implies
  $Bel[\Sigma \cup \Delta] \subseteq Bel[(\Sigma \cup \Delta) \star \{\mathbf{T} \Rightarrow \varphi\}]$.

Within the default revision framework, this gives us

- $\Sigma \not\mid\!\sim_\Delta \neg\varphi$ and $\Sigma \mid\!\sim_\Delta \psi$ implies

$$\Sigma \cup \{X_{new}\} \cup \Delta \cup \{\sigma \wedge X_{new} \Rightarrow \varphi\} \mid\!\sim \psi.$$

However, in general, this only holds if $\mid\!\sim_\Delta$ satisfies rational monotony, i.e. if $\Sigma \not\mid\!\sim_\Delta \neg\varphi$ and $\Sigma \mid\!\sim_\Delta \psi$ entails $\Sigma \cup \{\varphi\} \mid\!\sim_\Delta \psi$, which is a rather strong demand, but difficult to avoid. These considerations suggest that our discussion of default revision should focus on strong default entailment notions, like JMR-admissible default inference. To summarize, we have here a very general approach for generating new revision formalisms, but the subclass of those meeting all the standard desiderata is relatively small.

6 JZ-revision

We are now going to investigate how we may exploit suitable default inference notions $\mid\!\sim$ for the specification of ranking revision strategies extending minimal Spohn revision to finite default sets Δ. Taking into account the requiremnts for default revision, we are particularly interested in well-motivated JMR-admissible approaches. For instance, we may consider system JZ [Wey 98], which is based on the canonical construction of a $\kappa\pi$-model $JZ[\Delta]$ of a finite consistent Δ. The first, obvious step is to stipulate $R_0 \star \Delta = JZ[\Delta]$. The question is how to deal with arbitrary R, or at least those accessible by an iterated revision process starting at R_0, i.e. of the form $R_0 \star \Delta_1 \star \ldots \star \Delta_n$. We may distinguish two general procedures. One is to extend a given ranking construction strategy to a sufficiently large class of non-uniform priors. The other one is to apply the default projection paradigm and to reformulate the revision problem in a way which allows us to directly apply the original ranking-based default formalism. Let us start with the second proposal.

To exploit system JZ for ranking revision, we may want to represent each possible prior R by a finite default set Δ_R, e.g. verifying $R = JZ[\Delta_R]$. However, to gain more flexibility, we prefer a conditional characterization

$$R = JZ[\Delta_R](.|[\Sigma_R]),$$

where $\Sigma_R \subseteq \mathcal{L}$ is finite and may depend on Δ_R. Σ_R is meant to refer to hard, possibly indirectly specified raw propositional input knowledge. So we may characterize R by $\Sigma_R \cup \Delta_R$. If $R = R_0 \star \Delta_1 \star \ldots \star \Delta_n$, we may consider the characteristic knowledge bases $\Sigma_{R_i} \cup \Delta_{R_i}$ corresponding to its predecessors $R_i = R_0 \star \Delta_1 \star \ldots \star \Delta_i$, with $\Sigma_{R_0} = \Delta_{R_0} = \emptyset$. We can now try to reformulate our default revision problem for $\kappa\pi$-measures into one for finite knowledge bases $\Sigma \cup \Delta \subseteq \mathcal{L} \cup \mathcal{L}(\Rightarrow)$. For instance, in the context of system JZ, we are looking

for a revision function $\star'$ such that for consistent Δ_0, if $\Sigma^* \cup \Delta^* = (\Sigma \cup \Delta) \star' \Delta_0$,

$$JZ[\Delta](.|[\Sigma]) \star \Delta_0 = JZ[\Delta^*](.|[\Sigma^*]).$$

The obvious candidate for $\star'$ is of course the default projection strategy. There are several invariance results supporting this approach. The first one guarantees invariance w.r.t. the choice of X. It follows from the strong semanticality of JZ.

Theorem 6.1 (Projection invariance)
Let $X, Y \in \mathcal{L}$ be logically independent from a sublanguage $\mathcal{L}' \subseteq \mathcal{L}$ with $\Sigma, \Delta, \Delta_0 \subseteq \mathcal{L}'$. Then, for $\varphi \in \mathcal{L}'$,

- $JZ[\Delta \cup \Delta_0^{\sigma \wedge X}]([\varphi]|[\Sigma \cup \{X\}]) =$
  $JZ[\Delta \cup \Delta_0^{\sigma \wedge Y}]([\varphi]|[\Sigma \cup \{Y\}])$.

Of particular importance is of course the well-definedness of this approach, the independence of the revision result from the choice of a representation $\Sigma_R \cup \Delta_R$ for the prior $\kappa\pi$-measure R, assuming that there exists at least one[1].

Theorem 6.2 (Representation invariance)
Let $R = JZ[\Delta_R](.|[\Sigma_R]) = JZ[\Psi_R](.|[\Phi_R])$ be definable over $\mathcal{B}_{obj}$. If $\Sigma^* \cup \Delta^* = (\Sigma_R \cup \Delta_R) \star' \Delta_0$ and $\Phi^* \cup \Psi^* = (\Phi_R \cup \Psi_R) \star' \Delta_0$, then for $\varphi \in \mathcal{L}'$,

- $JZ[\Delta^*]([\varphi]|[\Sigma^*]) = JZ[\Psi^*]([\varphi]|[\Phi^*])$.

That is, if a $\kappa\pi$-measure R admits a representation $\Sigma_R \cup \Delta_R$, default projection followed by JZ-closure and conditionalization always produces the same ranking measure over the relevant part $\mathcal{L}_{obj}$ of the language. The proof exploits the logical independence of X_{new} and the fact that the ranking construction within X_{new} is independent from the overall JZ-construction given R. With these results, we may now define a suitable revision strategy $\star_{jz}$ based on system JZ and default projection.

Definition 6.3 (JZ-revision)
Let $R = JZ[\Delta_R](.|[\Sigma_R])$ be definable over $\mathcal{B}_{obj}$ and $\Sigma^* \cup \Delta^* = (\Sigma_R \cup \Delta_R) \star' \Delta_0$. Then we define over $\mathcal{B}_{obj}$,

- $R \star_{jz} \Delta_0 = JZ[\Delta^*](.|[\Sigma^*])$.

What also makes the default revision perspective appealing is the possibility to pack together a whole sequence of revision steps in a single ranking construction and conditionalization step.

[1] If we consider defaults $\Rightarrow^a$ of arbitrary strength $a \in [0, \infty]$, each $\kappa\pi$-measure which is finitary in the sense of being definable over a finite subsystem $\mathcal{B}' \subseteq \mathcal{B}$ is also representable.

Theorem 6.4 (Aggregation)
Let $\Delta_1, \ldots, \Delta_n$ be a sequence of default sets. Then, we have over $\mathcal{B}_{obj}$,

- $R_0 \star_{jz} \Delta_1 \ldots \star_{jz} \Delta_n =$
  $JZ[\Delta_1 \star' \ldots \star' \Delta_n](.|[X_1 \wedge \ldots \wedge X_n])$

JZ-revision may be seen as a natural extension of minimal Spohn revision to multiple conditional evidence.

Theorem 6.5 (Spohn extension)
Let R be a finitary $\kappa\pi$-measure. Then

$$R \star_{jr} \varphi = R \star_{jz} \{\mathbf{T} \Rightarrow \varphi\}.$$

In fact, this result holds for all $\star_{jz}$ derived from a JMR-admissible ranking choice function $\mathcal{F}^{JX}$. Because our intuitions about general conditional revision are quite rudimentary, we prefer to illustrate the mechanism of JZ-revision by multiple propositional revision, i.e. normal default sets $\{\mathbf{T} \Rightarrow \varphi_i \mid i \leq n\}$. Consider for instance $\Delta_1 = \{\mathbf{T} \Rightarrow \neg\varphi, \mathbf{T} \Rightarrow \neg\psi\}$ and $\Delta_2 = \{\mathbf{T} \Rightarrow \varphi, \mathbf{T} \Rightarrow \psi\}$ with $\varphi, \psi \in \mathcal{L}_{obj}$.

$R_0 \star_{jz} \Delta_1 = JZ[\Delta_1 \cup \{X_0 \Rightarrow \neg\varphi, X_0 \Rightarrow \neg\psi\}](.|[X_0]) =$
$(R_0 + 1[X_0 \wedge \varphi] + 1[X_0 \wedge \psi])(.|[X_0]) = R_1,$

$R_1 \star_{jz} \Delta_2 = JZ[\Delta_1 \cup \{X_0 \Rightarrow \varphi, X_0 \Rightarrow \psi\}](.|[X_0]) =$
$R_0 + 1[\varphi] + 1[\psi] + 1[X_0 \wedge \neg\varphi] + 1[X_0 \wedge \neg\psi])(.|[X_0]) =$
$R_0 + 1[\neg\varphi] + 1[\neg\psi] = R_2,$

$R_0 \star_{jz} \Delta_1 \star_{jz} \Delta_2 = JZ[\Delta_1 \cup \{X_0 \Rightarrow \neg\varphi, X_0 \Rightarrow \neg\psi\}$
$\cup \{X_0 \wedge X_1 \Rightarrow \varphi, X_0 \wedge X_1 \Rightarrow \psi\}](.|[X_0 \wedge X_1]) =$
$R_0 + 1[X_0 \wedge \varphi] + 1[X_0 \wedge \psi] + 1[X_0 \wedge X_1 \wedge \neg\varphi] +$
$1[X_0 \wedge X_1 \wedge \neg\psi])(.|[X_0 \wedge X_1]) = R_2.$

7 Beyond JZ-revision

JZ-revision verifies major desiderata for iterated belief change, including Darwiche and Pearl's rationality postulates $C1 - C4$ for revised belief [DP 97] and, where applicable, the ranking versions of the basic principles of probabilistic revision [SJ 80]. The correct handling of standard examples, the inherent plausibility of the default projection procedure and the existing justifications of system JZ provide further support. However, JZ-revision is also haunted by some major problems, notably a lack of semantic invariance and counterintuitive results in conflict with the information minimization philosophy.

If we ask for reasonable inheritance or independence features, it is well known – and at least implicitly accepted – that $\mathcal{KP}(\Delta_1) = \mathcal{KP}(\Delta_2)$ may fail to entail

$R \star_{jz} \Delta_1 = R \star_{jz} \Delta_2$. The reason is that the basic ranking semantics is not fine-grained enough to grasp relevant information encoded in the structure of the default base. For logically independent $\varphi, \psi \in \mathcal{L}$, we have e.g.,

- $R_0 \star_{jz} \{\mathbf{T} \Rightarrow \varphi, \mathbf{T} \Rightarrow \psi\} \star_{jz} \{\mathbf{T} \Rightarrow \neg\varphi\} = R_0 + 1[\varphi] + 1[\neg\psi]$, but
- $R_0 \star_{jz} \{\mathbf{T} \Rightarrow \varphi \wedge \psi\} \star_{jz} \{\mathbf{T} \Rightarrow \neg\varphi\} = R_0 + 1[\varphi] + 1[\varphi \wedge \neg\psi]$,

although $\mathcal{KP}(\{\mathbf{T} \Rightarrow \varphi, \mathbf{T} \Rightarrow \psi\}) = \mathcal{KP}(\{\mathbf{T} \Rightarrow \varphi \wedge \psi\})$. However, the ranking construction semantics, whose semantic units are formal shifting sums $\Sigma_{i \leq n} a_i A_i$ ($a_i \in [0, \infty]$, $A_i \in \mathcal{B}$) with $\mathcal{KP}^+(\Delta) = \{\Sigma_{i \leq n} a_i[\varphi_i \wedge \neg\psi_i] \mid R_0 + \Sigma_{i \leq n} a_i[\varphi_i \wedge \neg\psi_i] \in \mathcal{KP}(\Delta)\}$, offers an interesting semantic refinement [Wey 01]. Here we have

- $\mathcal{KP}^+(\{\mathbf{T} \Rightarrow \varphi, \psi\}) = \{x[\neg\varphi] + y[\neg\psi] \mid 1 \leq x, y\} \neq$
- $\mathcal{KP}^+(\{\mathbf{T} \Rightarrow \varphi \wedge \psi\}) = \{x[\neg\varphi \vee \neg\psi] \mid 1 \leq x\}$.

The problem is now that $\star_{jz}$ also violates invariance w.r.t. the ranking construction semantics. In fact, for independent φ, ψ, ψ', let

- $\Delta_1 = \{\mathbf{T} \Rightarrow \neg\varphi, \varphi \vee \psi \Rightarrow \varphi, \varphi \vee \psi' \Rightarrow \varphi, \mathbf{T} \Rightarrow (\psi \vee \psi') \rightarrow \varphi\}$,
- $\Delta_2 = \{\mathbf{T} \Rightarrow \neg\varphi, \varphi \vee \psi \vee \psi' \Rightarrow \varphi, \mathbf{T} \Rightarrow (\psi \rightarrow \varphi), \mathbf{T} \Rightarrow (\psi' \rightarrow \varphi)\}$.

Then $\mathcal{KP}^+(\Delta_1) = \mathcal{KP}^+(\Delta_2) = \{x[\neg\varphi \wedge (\psi \vee \psi')] + y[\neg\varphi \wedge \psi] + y'[\neg\varphi \wedge \psi'] + z[\varphi] \mid 1 \leq z, z+1 \leq x+y, x+y'\}$, but the corresponding canonical JZ-constructions are different.

- $R_0 \star_{jz} \Delta_1 = R_0 + 2[\neg\varphi \wedge \psi] + 2[\neg\varphi \wedge \psi'] + 1[\varphi] \neq$
- $R_0 \star_{jz} \Delta_2 = R_0 + 2[\neg\varphi \wedge (\psi \vee \psi')] + 1[\varphi]$.

From this it immediately follows that we cannot expect to characterize $\star_{jz}$ by some pseudo-distance relation on ranking constructions. The question is whether we should accept such a stronger dependence from the syntactic fine-structure of default sets. Fortunately, this weakness inherited from system JZ is not a necessary ingredient of JMR-admissible default inference. In particular, it doesn't hold for JLX-entailment, a default formalism with similar qualities than JZ which was first described in [Wey 01]. JLX is based on a partial order $\prec_{jlx}$ over formal shifting sums $a_0 A_0 + \ldots + a_n A_n$ comparing the inherent construction efforts or construction complexity. $\prec_{jlx}$ tries to combine the normality maximization (minimizing ranks) and the minimal construction (minimizing shifting efforts) philosophy in a transparent, well-motivated way. Roughly speaking, the idea is to proceed lexicographically and bottom-up, simultaneously minimizing at each rank the set of shiftable propositions A_i arriving there as well as their shifting lengths. The important fact is that for each finite Δ, there is a unique $\prec_{jlx}$-minimum, i.e.

$$Min_{\prec_{jlx}}(\mathcal{KP}^+(\Delta)) = \{\sigma_\Delta^{jlx}\}.$$

This gives us a canonical ranking choice function $\mathcal{F}^{jlx}$ with

$$\mathcal{F}^{jlx}(\Delta) = \{JLX[\Delta]\},$$

where $JLX[\Delta] = R_0 + \sigma_\Delta^{jlx}$. A relatively simple algorithm allows us to compute $JLX[\Delta]$. The semantic invariance of JLX w.r.t. the ranking construction semantics entails $JLX \neq JZ$. Let us now consider the corresponding ranking revision function $\star_{jlx}$. The above example illustrates the JLX-approach quite well.

$$R_0 \star_{jlx} \Delta_1 = R_0 \star_{jlx} \Delta_2 =$$
$$R_0 + 1[\neg\varphi \wedge (\psi \vee \psi')] + 1[\neg\varphi \wedge \psi] + 1[\neg\varphi \wedge \psi'] + 1[\varphi].$$

$\star_{jlx}$ verifies the usual basic revision desiderata. For the reasons given above, it appears to be preferable to $\star_{jz}$. However, both approaches coincide for propositional evidence.

Theorem 7.1 (Normal default bases)
Let $\Delta = \{\mathbf{T} \Rightarrow \varphi_i \mid i \leq n\}$. *Then* $R \star_{jlx} \Delta = R \star_{jz} \Delta$.

On the other hand, $\star_{jlx}$ and $\star_{jz}$ also share a common drawback. Let $\kappa\pi$-MCE be the ranking-based counterpart of cross-entropy-minimization (information minimization), which translates the ranking conditions into nonstandard probability constraints, determines the minimum-cross-entropy model and transfers the result back into the ranking framework [SJ 80, Wey 95]. $\kappa\pi$-MCE is known to be even more representation-dependent than JZ-revision. Nonetheless, there are some revision instances where it seems to be more in line with our intuitions than JZ- or JLX-revision. Their counterintuitive behaviour already shows up for multiple propositional evidence, i.e. as soon as we go beyond standard Spohn revision $\star_{jr}$. Let us take a look at a simple example.

- $R_1 = R_0 \star_{jlx} \{\mathbf{T} \Rightarrow (\varphi \vee \psi)\} = R_0 + 1[\neg\varphi \wedge \neg\psi]$,

- $R_2 = R_1 \star_{jlx} \{\mathbf{T} \Rightarrow \neg\varphi, \mathbf{T} \Rightarrow \neg\psi\} =$
  $(R_1 + 2[X \wedge \varphi] + 2[X \wedge \psi])(.|X) =$
  $R_0 + 2[\varphi] + 2[\psi] - 1[\mathbf{T}]$, i.e. $R_2([\varphi \wedge \psi]) = 3$.

But the most natural solution here – which is also sanctioned by $\kappa\pi$-MCE – seems to be

- $R_2 = R_1 \star \{\mathbf{T} \Rightarrow \neg\varphi, \mathbf{T} \Rightarrow \neg\psi\} =$
  $R_0 + 1[\varphi] + 1[\psi]$, i.e. $R_2([\varphi \wedge \psi]) = 2$.

Unfortunately, this problem arises for all default revision strategies based on JMR-admissible default inference.

8 A glimpse at JLR-revision

So, what's wrong with the default projection strategy, what are the reasons for the mishandling of the above examples and how can we address them? The answer turns out to be surprisingly simple and may be summarized as follows.

- *In front of conflicting evidence, we need a contraction step to prepare the default expansion process.*

In fact, suppose that $\Delta = \{\varphi_i \Rightarrow \psi_i \mid i \leq n\}$ conflicts with R and $\delta = (\varphi_0 \rightarrow \psi_0) \wedge \ldots \wedge (\varphi_n \rightarrow \psi_n)$. Then $[\delta]$ includes the most plausible worlds according to Δ, but $R([\delta]) > 0$. What blocks the application of the standard upwards shifting strategies, aimed at making the $[\varphi_i \wedge \neg\psi_i]$ less plausible, is the fact that this would push $R([\neg\delta]) = R([\vee\{\varphi_i \wedge \neg\psi_i \mid i \leq n\}]$ above 0, which is of course incompatible with $R([\delta]) > 0$. That is, before our construction can proceed in the usual way, we first have to retract the support for $\neg\delta$, in other words, we have to ensure that $[\delta]$ gets rank 0. Note that this is only possible if $R([\delta]) \neq \infty$, otherwise we cannot revise with Δ and may want to keep R.

To implement these ideas we are now going to sketch a new revision strategy $\star_{jlr}$ exploiting contraction and the JLX-philosophy. This allows us to extend the original JLX-construction to non-uniform ranking priors. Although it is possible to formulate this approach within an extended default revision framework, to simplify the presentation we prefer to specify the revision procedures directly for R and Δ. We proceed in two steps.

1. We use the minimal shifting philosophy to transform R into a suitable R^- with $R^-([\delta]) = 0$.
2. We apply standard JLX-revision to R^-, setting $R \star_{jlr} \Delta = R^- \star_{jlx} \Delta$.

It follows that for non-conflicting R, Δ, $\star_{jlr}$ coincides with $\star_{jlx}$. So, how should we enforce $R^-([\delta]) = 0$? The idea is to consider first those $\varphi_i \Rightarrow \psi_i$ with $R([\varphi_i \wedge \neg\psi_i]) = 0$. To realize the corresponding – not yet verified – ranking constraints $R([\varphi_i \wedge \psi_i]) + 1 \leq R([\varphi_i \wedge \neg\psi_i])$, it seems reasonable to start by enforcing rank 0 for $[\varphi_i \wedge \psi_i]$. A natural way to achieve this is the uniform downwards shifting of the $[\varphi_i \wedge \psi_i]$ until they touch the bottom 0. We may then reapply this procedure to arrive at bigger and bigger rank-0-areas, until there are no more $[\varphi_i \wedge \psi_i]$ to be shifted downwards. Let $R' = R - \Sigma_{j \leq m} a_j[\varphi_{i_j} \wedge \psi_{i_j}]$ $(a_j \geq 0)$ be the resulting $\kappa\pi$-measure. If $R'([\delta]) = 0$, we are done and we may set $R^- = R'$. If not, the apparently most rational move is to stipulate $R^- = R' - R'([\delta])[\delta]$.

Definition 8.1 (JLR-revision)
Let $R([\delta]) \neq \infty$ and R' as above. Then

- $R \star_{jlr} \Delta = (R' - R'([\delta])[\delta]) \star_{jlx} \Delta$.

We leave it as an easy exercise to the reader to apply the JLR-procedure to the example causing difficulties to $\star_{jlx}$. Looking at the usual desiderata, $\star_{jlr}$ turns out be very well-behaved. For instance, it verifies the postulates CR0–CR7 for conditional revision, although not $CR8$ [KI 99]. The Shore/Johnson axioms for MCE over discrete probability measures are – mutans mutandum – also valid for $\star_{jlr}$. However, JLR-revision differs from $\kappa\pi$-MCE because it satisfies $\mathcal{KP}^+$-invariance. We may also point to the fact that most approaches from the literature, e.g. [Wil 94, Leh 95, DPr 97], are dealing with a much more restricted kind of evidence.

Note. *For more complete versions of this paper, including in particular more details, examples and comparisons, the reader is invited to have a look at* www.mpi-sb.mpg.de/~weydert/pub/defepi.pdf.

REFERENCES

BCDLP 93 S. Benferhat, C. Cayrol, D. Dubois, J. Lang, H. Prade. Inconsistency management and prioritized syntax-based entailment. In *Proceedings of IJCAI 93*, Morgan Kaufmann, 1993.

DP 97 A. Darwiche and J. Pearl. On the logic of iterated belief revision. In *Artificial Intelligence*, 89 : 1-29, 1997.

DP 88 D. Dubois, H. Prade. *Possibility Theory.* Plenum Press, New York 1988.

DPr 97 D. Dubois, H. Prade. A synthetic view of belief revision with uncertain inputs in the framework of possibility theory. In *International Journal of Approximate Reasoning*, 17:295-324, 1997.

GP 96 M. Goldszmidt, J. Pearl. Qualitative probabilities for default reasoning, belief revision, and causal modeling. In *Artificial Intelligence*, 84 : 57-112, 1996.

KI 99 G. Kern-Isberner. Postulates for conditional belief revision. In *Proc. of IJCAI 99*, Morgan Kaufmann, 1999.

KLM 90 S. Kraus, D. Lehmann, M. Magidor. Nonmonotonic reasoning, preferential models and cumulative logics. In *Artificial Intelligence*, 44: 167-207, 1990.

Leh 95 D. Lehmann. Belief revision, revised. In *Proc. of IJCAI 93.* Montreal, Canada, 1995.

Pearl 90 J. Pearl. System Z: a natural ordering of defaults with tractable applications to nonmonotonic reasoning. In *Proc. of the Third Conference on Theoretical Aspects of Reasoning about Knowledge.* Morgan Kaufmann, 1990.

Rott 96 H.Rott. *Making up one's mind: foundations, coherence, nonmonotonicity.* Habilitationsschrift, Konstanz, 1996.

SJ 80 J.E. Shore and R.W. Johnson. Axiomatic derivation of the principle of cross-entropy minimization. In *IEEE Transactions on Information Theory*, IT-26(1):26-37, 1980.

Spohn 88 W. Spohn. Ordinal conditional functions: a dynamic theory of epistemic states. In *Causation in Decision, Belief Change, and Statistics*, W.L.Harper, B.Skyrms (eds.), Kluwer, 1988.

Spohn 90 W. Spohn. A general non-probabilistic theory of inductive reasoning. In R.D.Shachter et al. (eds.) *Uncertainty in Artificial Intelligence 4*, North-Holland, Amsterdam 1990.

Wey 94 E. Weydert. General belief measures. In *Proc. of UAI 94.* Morgan Kaufmann, 1994.

Wey 95 E. Weydert. Defaults and infinitesimals. Defeasible inference by non-archimdean entropy maximization. In *Proc. of UAI 95.* Morgan Kaufmann, 1995.

Wey 96 E. Weydert. System J - Revision entailment. In *Proc. of FAPR 96*, Springer, 1996.

Wey 98 E. Weydert. System JZ - How to build a canonical ranking model of a default knowledge base. In *Proc. of KR 98.* Morgan Kaufmann, 1998.

Wey 01 Rankings we prefer: a rational minimal construction semantics for default reasoning. In *Proceedings of DGNMR 01.* Potsdam, 2001.

Wil 94 M. Williams. Transmutations of knowledge systems. In *Proc. of KR94.* Morgan Kaufmann, 1994.

Decision Making by Intelligent Agents: Logical Argument, Probabilistic Inference and the Maintenance of Beliefs and Acts

John Fox
Advanced Computation Laboratory
Cancer Research UK
(Previously ICRF)
London WC2A 3PX U.K.

Peter McBurney
Department of Computer Science
University of Liverpool
Liverpool L69 7ZF U.K.

Abstract

The PRO*forma* is a language and a technology for designing, implementing, testing and delivering software agents that can operate in dynamic and uncertain environments. The agent specification language is based on R^2L, a logic language that formalises reasoning, decision-making and plan execution using a combination of classical and non-classical logics. PRO*forma* can be viewed as an object-oriented layer on top of R^2L that reifies the logic into a small set of "tasks", notably decisions, plans and I/O tasks. The heart of decision-making in both PRO*forma* and R^2L is a logical decision model based on the creation and evaluation of arguments for and against alternative beliefs and actions. In previous papers we have provided proof-theoretic and model-theoretic semantics for a Logic of Argument and suggested that argumentation is a general framework which subsumes many different forms of uncertain reasoning as special cases, including qualitative and non-monotonic logics; quantitative representations like probability and possibility, and "linguistic" representations of belief. Current implementations support a simple monotonic interpretation of argument based decision-making but do not address requirements for defeasibility or probabilistic belief revision. In this paper we discuss how argumentation can provide a framework for integrating these approaches into a unified agent.

1 INTRODUCTION

PRO*forma* is an executable specification language for modelling decisions, plans and other tasks that an agent needs to have in its repertoire in order to achieve its goals (Fox and Das 2000). The language is a versatile language for designing agent applications and has been successfully used in a variety of medical applications e.g., (Fox and Thomson 1998). Software is available for authoring, verifying and testing applications in the language and enacting the specifications as autonomous processes, or as agents supporting human users.

The PRO*forma* language is defined around four classes of task:

- An *enquiry* is any process that acquires information during the enactment of a plan, whether by a dialogue with another agent, or through a sensor or other device.
- An *action* is any act carried out on the agent's environment, whether a direct call to an effector or a request for action to another agent.
- A *decision* is a general procedure for making choices where there is uncertainty about the options, reliability of evidence, soundness of conclusions etc.
- *A plan* is a collection of tasks for achieving agent's goals. Plan tasks can be enacted in parallel or sequentially over time.

The PRO*forma* language is defined recursively over plans so that an agent can include hierarchical task structures of any complexity. The language combines features of a *formal specification language* as developed in software engineering with features of *knowledge representation languages* as developed in AI. Alternatively it can be viewed as a hybrid of a *logic programming* language (it supports inference in propositional and predicate logics, together with certain non-classical logics) an *object-oriented* language, in

which the objects are tasks, and an agent-oriented language which represents and maintains a complex mental state over time (Fox 2002).

A PRO*forma* task's behaviour is specified by defining properties associated with each class of task. All tasks include the following properties[1]:

Task-name:	<unique ID symbol>
Goal:	(<verb><object>)
Trigger:	<event>
Preconditions:	<condition>
Postconditions:	<assertions>
Autonomy:	'automatic' or 'authorise'

The meanings of these attributes are:

Goal. The situation that the task is intended to bring about e.g. change the state of current knowledge, as in a medical diagnosis decision; change the state of the world, as in a plan or action.

Preconditions. Situational conditions that must be true before a task may be initiated e.g. a patient must be male and of a certain age.

Postconditions. Tasks may have consequences in the world, such as the side effects of drugs.

Triggers. Situations or events that can evoke a task which has not been explicitly scheduled for enactment (see plans below).

Autonomy. A task can be enacted autonomously or require external supervision/authorisation.

Decisions and plans have further attributes that distinguish them from the "generic" task and from each other. Every plan, for instance, has the following attributes as well as the general ones described above:

Plan components: the actions, decisions, sub-plans etc. which make up the plan

Scheduling constraints: ordering constraints on components, if any

Abort conditions: if true plan aborts (fails)

Terminate conditions: if true plan terminates (succeeds)

In contrast a decision has the following distinguishing attributes:

Candidates: Set of decision options (choices)

Sources: Data required for decision to be taken

Arguments: Reasons for/against candidates

Commitments: Criteria/constraints for taking decision.

An agent's plans and decisions are interfaced to the outside world via *actions* and *enquiries*. These tasks inherit all the standard properties (above) but their behaviour does not have a specific impact on the present discussion and are not considered further here (see Fox and Das 2000 for discussion of details).

A PRO*forma* process is specified by composing tasks into collections of prepared plans that are to be carried out under particular contingencies. Tasks can be enacted sequentially (explicitly scheduled), in parallel, or reactively in response to events, or a mixture of these modes. A simple example is ERA (Early Referrals Application) which is being used in the UK National Health Service to assist doctors in deciding whether patients with suspected cancer should be referred for urgent investigation. ERA can be found at www.infermed.com/wap/era.[2] A graphical representation of the ERA task structure is shown in figure 1.

This ERA referral guideline consists of a single plan containing a sequence of component tasks: an enquiry that acquires patient data, a decision that follows the data collection, and two alternative actions that follow the decision. ERA is based on a policy document "Referral Guidelines for Suspected Cancer published by the UK Department of Health[3]. In this document the decision rules are set out in informal English as in the following conditions for *urgent referral for suspected breast cancer:*

- Patients with a discrete lump in the appropriate age group (e.g. age > 30).
- Signs which are highly suggestive of cancer such as:
  - Ulceration
  - Skin nodule
  - Skin distortion
  - Nipple eczema
  - Recent nipple retraction or distortion (<3 months)

[1] There are also other attributes that are not relevant to the current discussion.

[2] To run one of the examples select a specific cancer by clicking on "referral form" for one of the cancers indicated. A patient data form will be displayed; once this has been completed click on the OK button to submit the data for processing by a PRO*forma* server on the ICRF web site. ERA will return its recommendations on the basis of the data provided. Please note this is a demonstration and is not appropriate for practical clinical use without a professional clinical training.

[3] http://www.doh.gov.uk/cancer/referral.htm

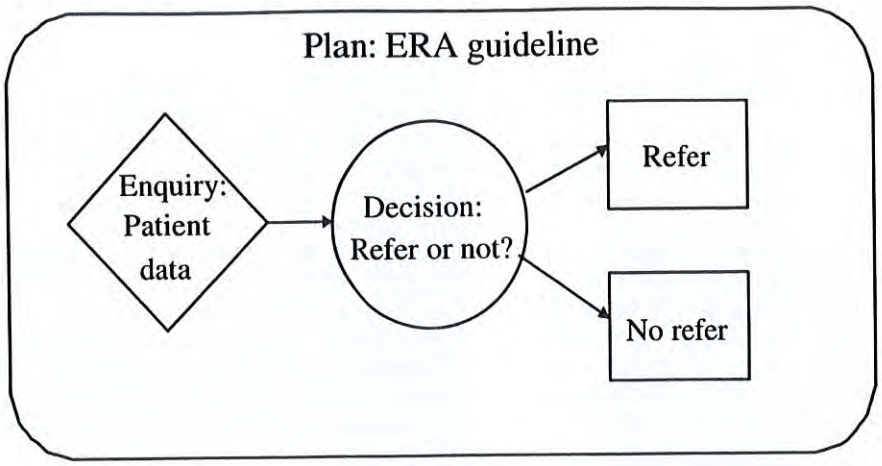


Figure 1: Task Structure of an ERA Referral Guideline

These criteria involve uncertainty (e.g. the risk associated with individual signs and symptoms) but do not explicitly quantify it (only using phrases like "highly suggestive"). This is common in medical guidelines and, despite the lack of precision and vagueness of meaning, qualitative terms are natural for doctors, easy to understand and easy to formalise.

2 DECISION MAKING IN PRO*forma*

It is straightforward to capture such simple criteria as the premises of rules in logical languages. In the PRO*forma* decision procedure rules are viewed as "arguments" that may support or oppose decision options (e.g. the patient does or does not require urgent referral) but are not necessarily conclusive. Arguments have a slightly different syntax as well as a weaker interpretation than *if ...then...* rules in traditional logic programs. An example of an argument "*for*" urgent referral in ERA is:

argument :: *for*, [1]

(nipple_changes includes Discharge and nipple_disc_features include 'Large volume' and nipple_disc_features includes 'Bilateral');

If arguments can be shown to be true then they support can support one or more decision option (in this case the option is "the patient should be urgently seen by a cancer specialist") but a single argument may not be decisive. Expert system rules are often associated with measures of confidence (e.g. certainty factors or other quantitative coefficients) but this is not the case here. Each argument merely represents a qualitative assertion that there is a *reason to believe* or a *reason to act* (Fox and Das 2000, chapter 4).

In this argumentation approach arguments can be "aggregated" to yield an overall measure of confidence in a decision option, and then a preference ordering on the set of options can be determined. In the simplest scheme each argument is treated as having equal weight (for or against a candidate) and the ordering is a direct function of the ratio of supporting to opposing arguments. Despite this simplicity argumentation schemes have been found to be effective and robust for many medical decisions (see Fox et al 2001 for a recent view). Argument systems are straightforward to set up and maintain, and easy for non-technical users to understand.

The argumentation model has emerged as a solution to a set of practical problems in the design of agent applications and decision support systems Applications are important because they provide settings in which to evaluate theoretical proposals empirically, but they are also important because they challenge theory. Medicine is arguably one of the most difficult areas of application for computer science and AI, not least because of the many different forms and sources of uncertainty that must be addressed. Argumentation can be given a clear and sound logical semantics providing some comfort that it can be used without incurring the pathological behaviour for which "*ad hoc*" decision procedures have been criticised (e.g. in some expert systems).

Argumentation is also a framework for uncertainty management that has theoretical and philosophical interest. In our view most, if not all, approaches to handling uncertainty can be viewed as special cases of a general argumentation model. In the next section we present a formalisation of the general model and then return to PRO*forma* as a platform for discussing some technical and practical requirements.

3 A FORMAL ACCOUNT OF ARGUMENT AND UNCERTAINTY

Argumentation is viewed as a general framework for logical inference and as an abstraction of a number of different kinds of uncertainty calculus, e.g., (Krause et al 1995; Fox and Das 2000). In this framework an argument has the *form* of a logical rule, but not its *force*. For example, a doctor may have a number of reasons to believe that a patient has had chicken pox as a child but cannot be certain. The difference between a logical procedure based on argumentation and a conventional proof procedure is that the conclusion of an argument cannot be guaranteed to be correct.

A logic of argument (LA) is summarised with the following consequence relation:

$$\text{Knowledge base} \cup \text{Data} \vdash_{LA} (\text{Claim} : \text{Grounds} : \text{Qualifier}) \quad [2]$$

Here the *knowledge base* is a collection of facts and rules about a domain (e.g. knowledge of diseases and their symptoms and treatments) together with a set of *data* such as a set of facts about a particular patient.

The turnstyle $\vdash$ is the *meta-symbol* representing a logic (here LA) under which we may construct valid arguments about alternative claims. There are multiple options for these axioms but, whatever the choice, the argument term has the same general form: it is a triple of elements consisting of a *Claim* which the argument addresses; the *Grounds* that justify the argument (a subset of facts and rules instantiated with data) and a *Qualifier* that represents the confidence warranted by the argument in the claim.

The qualifier may take a number of forms. It can be qualitative, as when an argument "supports" a claim or "opposes" it (ditto *for, against*), meaning that the argument increases or decreases our confidence in a claim but without indicating by how much. Alternatively the qualifier can be numeric, drawn from the [0,1] interval say, so probabilistic and other conventional measures of confidence may be used to quantify the force of arguments and the confidence we have in a claim.

The axioms of the base logic may vary. They could be those of classical propositional or predicate calculus for example or, as is often appropriate in medicine, Intuitionistic Logic (IL). The main difference here is that classical logic includes the axiom of the "excluded middle" (EM: $P \vee \neg P$) i.e. exactly one of P or not-P is true but not both. Since doctors may not know all the possible diseases or drugs that are relevant to a particular case dropping this axiom is often useful in modelling clinical reasoning. Furthermore there may be arguments for both P and not-P which are simultaneously valid, which cannot be accommodated in classical logic.

In their simplest form arguments are constructed by a mechanical proof procedure that generates a sequence of proof steps ending in a conclusion (e.g. "because the patient is elderly and has lost weight then the diagnosis is cancer", "because the patient may have cancer and cancer is a life-threatening and progressive disease we should refer him/her for urgent investigation"). However, although the form of the argument is similar to a proof, it does not have the conclusive force of a proof; in short a claim may be deductively valid but is *not necessarily true in the real world.* The premises for the argument may be unreliable for example (e.g. a laboratory test is unreliable) or the claim may later prove to be false as further data and arguments are considered.

We resolve this issue in decision making in two main ways. Firstly, arguments can have different precedences; conclusive argument may take precedence over merely indicative arguments, for example. If we know that a patient has breast cancer then we can argue, conclusively, that she has cancer, but if a patient is elderly and has lost weight we can argue that this supports a diagnosis of cancer but is not by itself sufficient to conclude this as a diagnosis.

Secondly, although arguments are logical in nature claims are not only true or false, but generally are ordered in terms of their relative merit. The ordering is determined by the aggregation of arguments for and against each claim. As data are acquired new arguments for or against each claim may become valid, resulting in changes to the aggregation and potentially changing the ordering.

We call the PROforma decision procedure "semi-quantitative", in that the mechanisms of argumentation are entirely logical, while the mechanism of aggregation is based on relative and/or absolute force of the arguments. Despite its simplicity the semi-quantitative approach has proved to be highly effective in many practical applications.

4 NON-MONOTONICITY AND ARGUMENTATION

From the point of view of non-monotonic logic argument aggregation is a kind of hybrid of classical monotonic reasoning and non-classical non-monotonic logic. As more data are acquired (e.g. a patient's symptoms) arguments are constructed from the data that may represent *reasons to believe* in different interpretations, or *reasons to act* in particular ways.

These arguments simply "pile up" monotonically. Observing a patient is elderly and has lost weight may support a diagnosis of cancer while a negative biopsy opposes this diagnosis, but both arguments are simultaneously valid, and will remain valid even as other arguments are formulated. The arguments remain

valid even if conclusive arguments against cancer are subsequently made; this apparent contradiction is resolved by the aggregation process not in the production of arguments themselves.

Issues of non-monotonicity only enter in when we take a decision, when we *commit* to a belief (e.g. the patient actually does have cancer rather than any of the alternatives) or to an act (e.g. to begin chemotherapy). Only if the agent has reason to change to change its mind about the diagnosis or about the treatment does it have to consider the possible downstream consequences (e.g. another decision about follow-up tests may no longer be valid).

We have found that for many medical applications it is quite practical to ignore issues of non-monotonicity. Typically we can arrange that all relevant data about a patient have been acquired before any commitment is made, so that at the point of decision all arguments have been derived and aggregated and the most preferred decision option selected. Under these circumstances it will not normally be necessary to "decommit" from the option.

This is not, however, guaranteed in all possible situations. We cannot for example guarantee that any data we have acquired will prove to be correct – a hospital laboratory may inform us that its equipment is faulty some days after telling us that a test was normal for example. Or alternatively it may not be possible to arrange that all data are available before starting an urgent treatment. We may be forced to act on a tentative diagnosis, while we wait several days for the definitive test results to become available which could show the tentative diagnosis to be wrong.

The argumentation model can be extended to accommodate some of these problems by means of a suitably adapted belief-maintenance scheme. Let Δ be a database of beliefs associated with a decision; K the set of claims between which we are choosing, and A the set of arguments that support these claims $\{(K_i:A_j: +)\}$. Then the conditions under which an argument A_j for a claim K_i is valid are as follows

$$\text{Valid}(K_i:A_j) \textit{ iff } \exists\ \delta \in \Delta \text{ and Provable}(K_i: \delta:+)$$

$$\text{Valid}(K_i:A_j) \textit{ iff } \exists\ K_j \in K,\ \delta \in \Delta$$

$$\text{and Provable}(K_i: K_j \& \delta: +) \qquad [3]$$

Which is to say $(K_i:A_j)$ is a valid argument (under some logical consequence relation such as LA) if and only if there is a provable line of reasoning from a ground belief which supports a claim K_i or an indirect argument which depends upon other Valid arguments . It is a relatively simple programming task to create an argument maintenance system that will automa tically add and delete argument terms as data are added or deleted from database Δ, as when we discover that the laboratory equipment is faulty or that a default assumption about the patient is false.

A claim remains *acceptable* while the following condition holds:

$$\text{Acceptable } (K_i) \textit{ iff } \exists\ A_j \in A \text{ and Valid}(K_i:A_j) \qquad [4]$$

Again it is a simple programming task to maintain a preference ordering over the acceptable claims.

Argument and claim maintenance are not currently supported in the PRO*forma* technology though ideally it would be part of the functionality of the decision task manager. This is a subject of continuing research. For example, an interesting refinement of the maintenance scheme outlined is based on the observation that the set of "acceptable" claims based on the existence of at least one valid argument is only one criterion of acceptability. We may, for example, require a more stringent criterion, such as "there must not only be a supporting argument but also there should be no excluding counter-arguments" for a claim to be acceptable.

In everyday language a particular disease may only be a "possible" diagnosis if there are arguments in support of it being present *and* no arguments that rule it out. Another possible acceptability condition is that a claim is only acceptable if it is "plausible" meaning the aggregated supporting arguments have greater force than the aggregated opposing arguments. If arguments are also weighted with probabilities (see next section) we could have the acceptability condition that the probability of a claim must be greater than some value, such as 0.5, for it to be considered as an option in the decision process.

Argumentation also appears to offer a promising approach to providing interpretations for common "linguistic" uncertainty terms. Linguistic representations of uncertainty have considerable value in building practical medical systems as clinical users find terms like "possible", "probable" useful and natural. (Elvang-Gøransson et al 1995) have formally explored logical interpretations of such terms in analysing the idea of acceptability of arguments. (Glasspool et al 1999) have shown empirically that it can provide at least as good an account of people's use of such terms as more traditional probabilistic explanations. The relationship between argument patterns and linguistic uncertainty terms and the relationship to modal and default logics is also discussed by (Fox and Das 2000, chapter 4).

For practical agent technologies, such as PRO*forma*, argument maintenance is only part of the problem however, because decision-making does not only entail decisions among alternative *beliefs* but also commitments to *actions* in the world. Suppose a PRO*forma* agent decides that a patient is suffering from

cancer, and that the required treatment is chemotherapy, a substantial medical process involving extensive planning, ordering of services and commitment of resources. If the agent later finds that the patient data on which these decisions were made were incorrect (a common enough event) then chemotherapy may no longer be the most preferred or an acceptable treatment.

The difficulty is not that an agent's beliefs about the diagnosis become invalid, the argument maintenance and aggregation systems will take care of that, but rather that commitments will have been made by the agent that are no longer under its control (e.g. making an appointment for the patient to be seen urgently by a cancer specialist). The agent must not only revise its beliefs but also take any action that is necessary to *reverse its commitments*.

In general an arbitrary number of commitments can arise as a result of decision-making and we require a general solution to this problem. This is not simply a matter of issuing "cancellation notices" for all commitments erroneously made, for by the time the error is identified any number of consequences may already be irreversible. However, simply cancelling every incomplete order is not always sufficient either. For example a cancellation could have greater costs than leaving the commitment in place so the challenge is to seek a minimum-cost solution.[4]

5 ARGUMENTS AND PROBABILITIES

On this decision model the decision process is a hybrid of classical monotonic reasoning (for argumentation) and non-classical non-monotonic logic (for commitment and de-commitment). It can also be viewed as a hybrid of logical and quantitative methods, in that in "weighing the pros and cons" of a claim we are effectively applying a linear decision rule with uniform weights on the arguments. It is an established result in the decision analysis literature that such rules are frequently as effective as more sophisticated decision procedures that use complex weighting schemes (e.g. Bayesian and expected utility decision models) (Fox, Barber, and Bardhan 1980).

In medicine it appears that precise weighting of arguments is frequently ignored in practice (the NHS cancer referral criteria above leave uncertainty implicit) and attempting to be precise about the uncertainty involved frequently produces little or no improvement in medical outcome (Fox et al 2001).

Despite this observation there are surely domains and applications where decision-making requires precise weighting of evidence, as when some symptoms are more strongly diagnostic of some diseases than others. PRO*forma* supports a simple integer weighting scheme for this purpose, which allows an application designer to place differential weights on arguments, which can be taken into account by the built-in aggregation functions in establishing the preference orderings over the decision options. While this scheme is practical it is subject to many of the standard criticisms of *ad hoc* decision procedures (e.g. the argument that belief revision methods which do not satisfy the probability axioms will be "incoherent" (Lindley 1985).

The use of an argumentation framework *per se* is not the primary weakness since the scheme in [2] is sufficiently general to accommodate different logics (including classical truth-functional logic) and other representations of uncertainty (including probability) while remaining within the argumentation meta-theory. This implies that if an application requires a probabilistic decision procedure it ought to be possible to capture it within the argumentation framework.

The way to achieve this in PRO*forma* is by defining a new class of argument-based decision called a *Bayesian decision* which respects the probability axioms and constraints (Figure 2).

Bayesian decisions inherit the usual properties of the standard decision task outlined earlier, though we rename them with more familiar terminology from the Bayesian Network literature).

- Input nodes = Data sources for a decision (e.g. patient data)
- Output nodes = Decision options or candidates (e.g. diagnoses)
- External nodes = Facts in a knowledge base (e.g. medical theories)
- Paths =Arguments.

The paths in Bayesian networks are equivalent to arguments in standard PRO*forma* decisions. Paths from input to output nodes are equivalent to implications from premises to conclusions. Paths may include internal nodes representing Boolean combinations of data, possibly drawing on external facts in a knowledge base. The paths in this graph are (IN1&EN1-> ON1); (IN2->ON2) and (IN2 & IN3 & EN2) -> ON2). The external nodes can provide support for the argument path, in the same way that warrants support the derivation of claims from data in Toulmin's influential model of argument (Toulmin 1958).

[4] As health services and airlines know patients and passengers often adopt the least-cost solution of doing nothing, and so providers have to allow for a substantial proportion of "no shows". We do not think that doing nothing is always a reasonable solution to this problem, but await a general solution to this difficult challenge, where "non-monotonicity meets the real world".

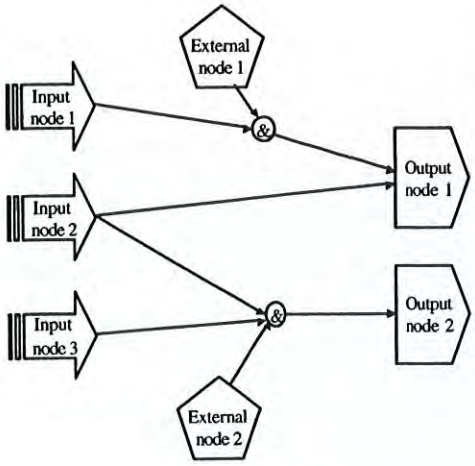


Figure 2: Arguments as Networks

PRO*forma* arguments are associated with simple qualifiers like "for" and "against", or simple integer weights to indicate the force of an argument with respect to particular decision options (claims). The equivalent to these in a Bayesian decision is a quantitative weight drawn from [0,1].

Path weight: The weight represents the conditional probability of a candidate being the most preferred given that all input nodes and nodes external to the path are true.

Prior belief: Each candidate will have a coefficient in [0,1] which represents the prior belief in the candidate.

Updating function: The revision function that is to be applied to the current belief in each candidate to yield a posterior probability for the candidate for each true argument. The updating function is a special class of aggregation function.

The practicality of imp lementing a Bayesian decision has been demonstrated by (Vicens 2001) who has implemented an application for differential diagnosis of abdominal pain based on a classical Bayesian decision support system originally developed by (de Dombal et al 1972). (Vicens 2001) contains further details of how probabilistic argumentation is incorporated into PRO*forma* in this model.

However, the model remains unsatisfactory in certain respects. If we opt for a standard Bayesian belief updating function then this places strict constraints on the structuring of the network. For example, there must be a complete set of dependency paths from all input nodes to all output nodes. Vicens encoded all possible paths in the abdominal pain diagnosis system mentioned above. This was possible because she had available a complete conditional probability table for symptoms and diseases which had been compiled from objective clinical data by de Dombal (*op cit*).

In practice it will be more typical that objective information of this kind is not available, so we will need to draw upon subjective clinical opinion in modelling a medical decision. Using this method, however, the decision graph is likely to be incomplete. As we saw with the ERA rules an expert diagnostician will typically assert statements like "the presence of swollen glands suggests glandular fever" while making no comment on (or possibly even denying "the absence of swollen glands suggests an absence of glandular fever (and therefore the presence of something else)". Indeed it is a standard result from the psychology of reasoning and decision making that people generally focus on things that are true and pay insufficient attention to information and relationships that are false (e.g. Legrenzi et al 1993). In other words if we rely entirely on knowledge acquisition from human experts there may well be paths missing from the graphs.

A standard Bayesian updating function cannot accept an incomplete topology. One possible solution is to associate a "graph normalisation function" with Bayesian decisions that autonomously infers all the links and estimates associated probabilities missing from the graph. However, this approach may lead to problems when the results are presented to the medical experts as the automatically-added paths may not have any evidence to support them. An alternative may be to use a non-probabilistic representation of uncertainty. For example, we may view those edges for which there is no evidence as being constrained by the assignment of uncertainty measures given to those edges for which there is evidence. This view means that the uncertainty values appropriate for the former edges (i.e., those with no evidence) may be seen as lying inside some range of values, between a minimum value and a maximum value. Such an approach leads naturally to Belief Function (Dempster-Shafer) Theory, where the minimum and maximum values are given by Belief and Plausibility values respectively (Shafer 1976). Belief Function Theory is a generalization of probability theory, with a weaker set of axioms.

6 CONCLUSIONS

In this paper we have presented an agent representation language and logical argumentation system that have been used successfully to represent uncertain beliefs and guide decision-making in medical domains. Although this application domain is very specific, it has proven to be a rich source of general theoretical problems. The most interesting conclusion is that argumentation appears to be a general framework

within which many other uncertain reasoning techniques can be modeled as special cases, since it can accommodate different base logics (e.g. classical and intuitionistic logics) and various representations of uncertainty (quantitative, qualitative and linguistic). As examples we have discussed non-monotonic reasoning and probabilistic belief revision. The medical domain continues to raise new challenges. For example, it reveals the apparent absence of a well-developed theory of defeasible practical reasoning, a theory of decision-making that formalises commitments and de-commitments to actions that have real-world consequences, which would be the action analogue of non-monotonic logics of beliefs.

References

Ambler S "A categorical approach to the semantics of argumentation" *Mathematical Structures in Computer Science*, 6 (2), 167-188, 1995.

Das S K, Fox J, Hammond P, Elsdon D "A flexible architecture for autonomous agents" *J. Exper. Theoretical Artificial Intelligence*, 9 (4), 407-440, 1997

De Dombal, T "Computer Aided Diagnosis of Abdominal Pain", *British Medical Journal*, 2(1), 9-13, 1972

Elvang-Goransson M, Krause P J, Fox J "Acceptability of arguments as logical uncertainty" in M Clarke, R Kruse, S Moral (eds) *Symbolic and Quantitative Approaches to Reasoning und Uncertainty, ECSQARU93*, Lecture notes in computer science 747, Berlin: Springer, 1993.

Fox, J, Barber, D and Bardhan, KD, "Alternative to Bayes? A quantitative comparison with rule-based diagnostic inference" *Methods of Information in Medicine*, 19, 210-215, 1980.

Fox J, Krause P and Elvang-Goransson M, "Argumentation as a general framework for uncertain reasoning" *Proc. 9th Conference on Uncertainty in AI*, Washington, 1993.

Fox, J. and Parsons, S. "Arguments about beliefs and Actions". in A. Hunter and S. Parsons (eds.) *Applications of Uncertainty Formalisms*, Springer Verlag, 1998.

Fox J and Thomson R "Decision support and disease management: a logic engineering approach" *IEEE Transactions in Biomedicine*, 2 (4), 217-228, 1998.

Fox J and Das S *Safe and Sound: Artificial Intelligence in Hazardous Applications*, AAAI and MIT Press, 2000

Fox J, Glasspool D, Bury J "Quantitative and qualitative approaches to reasoning under uncertainty in medical decision making" in S Quaglini, P Barahona, S Andreasen (eds) *Artificial Intelligence in Medicine, Lecture Notes in Artificial Intelligence 2101*, Berlin: Springer, 2001.

Glasspool, D and Fox, J "Understanding probabilistic words by constructing concrete mental models" In: M. Hahn and S. C. Stones (eds): *Proc. 21st. Conf. Cognitive Science Society*. Lawrence Erlbaum Associates: New Jersey, USA, pp. 185-190, 1999.

Krause P, Ambler S, Elvang-Goransson M, Fox J "A logic of argumentation for reasoning under uncertainty" *Computational Intelligence*, 11 (1), 113-131, 1995.

Legrenzi, P, Girotto, V and Johnson-Laird, PN Focussing in Reasoning and Decision Making. *Cognition, 49*, 37-66, 1993.

Lindley, DV *Making Decisions*, Second Edition. London: John Wiley and Sons, 1985.

Shafer, G *A Mathematical Theory of Evidence*, Princeton University Press: Princeton NJ, 1976.

Toulmin, SE *The Uses of Argument*, Cambridge University Press: Cambridge, UK, 1958.

Vicens, C Dissertation for Masters degree in Decision Analysis, London School of Economics, 2001.

Appendix

```
/** PROforma Guideline: Suspected breast cancer **/
/** 25/10/2000, (simplified by JF for presentation, 1/5/2001 **/

plan :: Breast ;
  caption :: 'Breast' ;
  component :: Clinical_information ;
  component :: Referral_decision ;
    schedule_constraint :: completed(Clinical_information) ;
  component :: No_two_week_referral ;
    schedule_constraint :: completed(Referral_decision) ;
  component :: Two_week_referral ;
    schedule_constraint :: completed(Referral_decision) ;
end plan .

decision :: Referral_decision ;
  caption :: 'Referral decision' ;
  choice_mode :: single ;
  support_mode :: symbolic ;
  candidate :: Two_week_referral ;
    argument :: for, ( tissue_changes includes 'Discrete lump' and age >= 30 ) ;
    argument :: for, ( skin_changes includes Ulceration ) ;
    argument :: for, ( skin_changes includes Nodule ) ;
    argument :: for, ( skin_changes includes Distortion ) ;
    argument :: for, ( nipple_changes includes Eczema ) ;
    argument :: for, ( nipple_changes includes 'Retraction_or_distortion' ) ;
    recommendation :: Netsupport( Referral_decision, Two_week_referral ) >= 1 ;
  candidate :: No_referral ;
    recommendation :: Netsupport( Referral_decision, Two_week_referral ) < 1 ;
end decision .

action :: No_two_week_referral ;
  caption :: 'No two week referral' ;
  precondition :: result_of( Referral_decision ) = No_referral ;
  procedure :: 'No referral' ;
end action .
action :: Two_week_referral;
  caption :: 'Two week referral' ;
  precondition :: result_of( Referral_decision ) = Two_week_referral;
  procedure :: 'Referral, though not necessarily urgent' ;
end action .
```

PR*Oforma* specification of the ERA breast cancer referral guideline (excluding data definitions and with a reduced number of decision options to improve clarity). The specification covers the tasks in figure 1, consisting of a plan that contains sequence of atomic tasks: an enquiry to collect patient data, a decision to determine urgency, and two alternative actions, one of which will depend on the result of the referral decision.

Embedding Default Logic in Propositional Argumentation Systems

Dritan Berzati
Department of Informatics
University of Fribourg
rue de Faucigny 2
1700 Fribourg, Switzerland
dritan.berzati@unifr.ch

Bernhard Anrig
Department of Informatics
University of Fribourg
rue de Faucigny 2
1700 Fribourg, Switzerland
bernhard.anrig@unifr.ch

Jürg Kohlas
Department of Informatics
University of Fribourg
rue de Faucigny 2
1700 Fribourg, Switzerland
juerg.kohlas@unifr.ch

Abstract

In this paper we present a transformation of finite propositional default theories into so-called propositional argumentation systems. This transformation allows to characterize all notions of Reiter's default logic in the framework of argumentation systems. As a consequence, computing extensions, or determining wether a given formula belongs to one extension or all extensions can be answered without leaving the field of classical propositional logic. The transformation proposed is linear in the number of defaults.

Keywords: Default Logic, Argumentation Systems, Embedding, Propositional Logic.

1 INTRODUCTION

1.1 MOTIVATION, CONTRIBUTION AND OUTLOOK

Reiter's *default logic* (Reiter 1980) is at once the most popular and the most controversial non-monotonic formalism dealing with uncertain information. Popular, because knowledge can be represented in a natural way in default logic. Controversial, because the concept of an extension as presented by Reiter is quite complex. Since default logic has been presented (Reiter 1980) a lot of effort took place to establish relationships between default logic and other non-monotonic formalisms, such as *autoepistemic logic* (Konolige 1988) and *circumscription* (Lifschitz 1990).

Poole (Poole 1988) (*Theorist*) was — to our knowledge — the first one who tried to embed Reiter's default logic in an *assumption-based framework* (De Kleer 1986a; De Kleer 1986b; Forbus and de Kleer 1993; Kean and Tsiknis 1992; Kean and Tsiknis 1993). It turned out that his proposition derives more extensions than the original default theory allows. The transformation proposed by Ben-Eliyahu & Dechter (Ben-Eliyahu and Dechter 1996) (*meta-interpretations*) results in a one-to-one correspondence of the original default theory and the their constructed propositional theory, but their transformation is in general is NP-complete. Another quasi-transformation proposed by Bondarenko et al. (Bondarenko et al. 1997) (*abstract argumentation-theoretic approach*) lacks a procedure to compute the extensions of the specified default theory. In conclusion, the transformations mentioned so far have their weaknesses. That is the main reason, why we searched for a translation of propositional default theories into an assumption-based framework, called *argumentation systems* (Kohlas and Monney 1993; Kohlas and Monney 1995; Kohlas et al. 1998; Haenni 1998; Anrig et al. 1999), to overcome the weaknesses encountered so far.

In argumentation systems all inference is done without leaving the field of classical propositional logic. Informally speaking, argumentation systems allow to judge open questions (hypotheses) about the unknown or future world in the light of the given knowledge. The problem is to determine a set of possible assumptions that allows to deduce the hypothesis from the given knowledge. Note that the concept of argumentation systems has many different meanings and we refer to (Chesñevar et al. 1998) for an overview. For instance, the *argumentation framework* proposed by Prakken (Prakken 1993), which uses the same terminology as in Bondarenko et al. (Bondarenko et al. 1997), is a derivation of Nute's (Nute 1994) *defeasible logic*. So, Prakken's proposal has only the term *argumentation* with our approach in common and it is not an assumption-based framework. Therefore, we do not make a comparison between Prakken's (Prakken 1993) and our approach. But it

is worth noting that Prakken's proposal is close to Lukasiewicz's (Lukasiewicz 1998) approach which extends still another derivation of Nute's defeasible logic with probabilities. Lukasiewicz proposal in turn has similar properties as the framework proposed by Benferhat et al. (Benferhat et al. 2000). However, in this paper the notion of an argumentation system is defined in Section 4.

In this paper, we present a transformation of Reiter's default logic into the framework of argumentation systems. This transformation is linear in the number of defaults and allows us to establish a bijection between extensions of the default theory and some subsets of assumptions of the argumentation system. Moreover, we provide procedures to answer any query on the specified default theory.

The rest of this paper is organized as follows: Section 1.2 repeats the necessary basic notions of propositional logic as a remainder. In Section 2 we briefly present Reiter's default logic and introduce the transformation of propositional default theories into associated propositional argumentation systems. Section 3 discusses the basic elements of propositional argumentation systems. In Section 4 we present the new results obtained using the transformation mentioned. We conclude with an outlook in Sections 5. Proofs of the new theorems can be found in (Berzati and Anrig 2001).

1.2 PRELIMINARIES

We assume that the reader is familiar with the basic concepts of propositional logic. We work with the usual propositional language $\mathscr{L}_V$. That is, $\mathscr{L}_V$ denotes set of all formulas which can be formed using the finite set of *propositions* $V = v_1, ..., v_n$, where as usual, $\wedge$ denotes *conjunction*, $\vee$ *disjunction*, $\neg$ *negation* and $\rightarrow$ *implication*. A *literal* $\ell \in V^{\pm}$ is a proposition or the negation of a proposition. $\top$ denotes *verum* and $\bot$ *falsum*. Lower-case Greek letters $\alpha, \beta, \ldots$ denote formulas and upper-case Greek letters $\Sigma, \Xi, \ldots$ denote sets of formulas called *theories*. For a set of formulas *satisfiability* and *consistency* is defined as usual. Propositional derivability is denoted by $\vdash$ and the corresponding consequence operator by Th.

For a given a formula $\phi \in \mathscr{L}_V$ and a subset of propositions $Q \subseteq V$, we are often interested in computing a formula $\psi \in \mathscr{L}_Q$ such that

(i) $\phi \vdash \psi$ and
(ii) $\varphi \in \mathscr{L}_Q$ and $\phi \vdash \varphi$ imply $\psi \vdash \varphi$.

This is the *marginalization-problem* (Kohlas et al. 1999; Haenni et al. 2000; Haenni and Lehmann 2000; Kohlas et al. 1998) which is a special case of *literal forgetting* (Lin and Reiter 1994) and equivalent to computing *prime implicates* (Raiman and De Kleer 1992). A formula ψ which satisfies the two conditions above is called a *marginal* of ϕ with respect to $Q \subseteq V$, denoted by $\phi^{\downarrow Q}$. Note that $Th_{\mathscr{L}_Q}(\phi^{\downarrow Q}) = Th_{\mathscr{L}_V}(\phi) \cap \mathscr{L}_Q$. We will not enter into the details how such a marginal can be computed. The interested reader is referred for example to Marquis (Marquis 2000).

Finally, for any set S we denote with $|S|$ the *cardinal number* of S, i.e. the number of elements contained in S, and with 2^S the *power set* of S, i.e. the set of all subsets of S.

2 DEFAULT LOGIC AND TRANSLATION

2.1 DEFAULT THEORIES AND EXTENSIONS

A Reiter's *default theory* (Reiter 1980) is a pair $\langle \Sigma, \Delta \rangle$, where Σ consists of formulas over a propositional language $\mathscr{L}$ and Δ is a set of defaults. A *default* δ is a construct of the form[1]

$$\frac{\alpha : \beta_1, \ldots, \beta_k}{\gamma}, \text{ where}$$

(i) $\alpha, \beta_1, \ldots, \beta_k, \gamma$ are formulas in $\mathscr{L}$,

(ii) $\mathtt{pre}(\delta) := \{\alpha\}$ is called the *prerequisite* of the default δ,

(iii) $\mathtt{jus}(\delta) := \{\beta_1, \ldots, \beta_k\}$ is called the *justification* of the default δ,

(iv) $\mathtt{con}(\delta) := \{\gamma\}$ is called the *consequence* of the default δ.

The selectors pre, jus and con are naturally generalized to subsets $D \subseteq \Delta$ of defaults:

$$\mathtt{pre}(D) := \bigcup_{\delta \in D} \mathtt{pre}(\delta),$$
$$\mathtt{jus}(D) := \bigcup_{\delta \in D} \mathtt{jus}(\delta),$$
$$\mathtt{con}(D) := \bigcup_{\delta \in D} \mathtt{con}(\delta).$$

Observe that the selectors pre, jus and con, respectively, return sets of formulas of $\mathscr{L}$. It is sometimes

[1]Note that different defaults have in general a different number of justifications; yet for the sake of simplicity, we do not index the k's in the sequel.

necessary to select the formulas themselves. Thus for any default δ with k justifications, we define following selectors:

$$\mathtt{p}(\delta) := \alpha, \quad \mathtt{j}_i(\delta) := \beta_i, \quad \mathtt{c}(\delta) := \gamma.$$

$\langle \Sigma, \Delta \rangle$ is called *finite* if both Σ and Δ are finite. In this paper we focus on finite default theories. Therefore let $P = \mathrm{Var}(\langle \Sigma, \Delta \rangle)$ denote the set of all propositions occurring in the formulas of $\langle \Sigma, \Delta \rangle$ and $\mathscr{L}_P$ the propositional language of $\langle \Sigma, \Delta \rangle$.

The fundamental notion in default logic is the *extension*. Intuitively, it denotes the deductive closure of a maximal set of formulas of $\langle \Sigma, \Delta \rangle$ containing Σ and "consistently" adding the consequences of some defaults.

Definition 1 (Extension: (Reiter 1980), Theorem 2.1) *A set $E \subseteq \mathscr{L}_P$ of formulas is called* extension *of a default theory $\langle \Sigma, \Delta \rangle$, where $P = \mathrm{Var}(\langle \Sigma, \Delta \rangle)$, iff there exists a sequence $E_0, E_1, \ldots$ such that*

$$\begin{aligned} E_0 &= \Sigma, \\ E_{j+1} &= Th(E_j) \cup \mathtt{con}(\{\delta \in \Delta : \mathtt{p}(\delta) \in E_j \text{ and } \\ &\qquad \neg \mathtt{j}_i(\delta) \notin E \text{ for every } i\}), \\ E &= \bigcup_{j=0}^{\infty} E_j. \end{aligned}$$

When we write "for every i" ($\forall i$ for short), we mean for each justification of the respective default.

Example 1 *The default theory $\langle \Sigma, \Delta \rangle$ with $\Sigma = \{b \to \neg a \wedge \neg c\}$ and $\Delta = \{\frac{\top : a}{a}, \frac{\top : b}{b}, \frac{\top : c}{c}\}$ has two extensions given by $E = Th(\Sigma \cup \{a, c\})$, and $E' = Th(\Sigma \cup \{b\})$.*

Assume that Σ of a given default theory $\langle \Sigma, \Delta \rangle$ is inconsistent. Then $Th(\Sigma)$ is the only extension of $\langle \Sigma, \Delta \rangle$ which is obviously inconsistent. Therefore, a default theory is called *consistent* iff it has at least one consistent extension. In this case Σ must be consistent, since it is included in every extension. But consistency of the initial belief does not guarantee that the default theory has an extension.

Example 2 *The default theory $\langle \Sigma, \Delta \rangle$ with $\Sigma = \emptyset$ and $\Delta = \{\frac{\top : p}{\neg p}\}$ has no extension!*

Since we are not interested in trivial inconsistent extensions, we will say that a default theory has *no extension* iff the default theory $\langle \Sigma, \Delta \rangle$ lacks an extension or Σ is inconsistent.

Default theories $\langle \Sigma, \Delta \rangle$ lacking an extension should not be confused with default theories that have a unique consistent extension which is just the deductive closure of Σ.

Example 3 *Let $\langle \Sigma, \Delta \rangle$ be a default theory with $\Sigma = \{p, q\}$ and $\Delta = \{\frac{p : \neg q}{\neg q}\}$. The reader may verify that $E = Th(\{p, q\})$ is the only extension.*

Two concepts of consequence are of main interested when working with default theories:

Definition 2 *A formula ϕ is called* credulous consequence *of a default theory $\langle \Sigma, \Delta \rangle$ iff it belongs to at least one extension, and* skeptical consequence *iff it belongs to all extensions.*

2.2 EMBEDDING GENERAL DEFAULT THEORIES

Let $\langle \Sigma, \Delta \rangle$ be a default theory with formulas in $\mathscr{L}_P$, where $P = \mathrm{Var}(\langle \Sigma, \Delta \rangle)$. For any default $\delta \in \Delta$ consider the following transformation t:

$$\begin{aligned} t(\delta) = &\{a^{\mathtt{p}}_\delta \to \neg \mathtt{p}(\delta)\} \cup \\ &\{a^{\mathtt{j}}_{\delta,i} \to \mathtt{j}_i(\delta) : i = 1, \ldots, |\mathtt{jus}(\delta)|\} \cup \\ &\{a^{\mathtt{c}}_\delta \to \mathtt{c}(\delta)\}. \end{aligned} \tag{1}$$

We call the triple $\langle a^{\mathtt{p}}_\delta, \{a^{\mathtt{j}}_{\delta,i} : i = 1, \ldots, |\mathtt{jus}(\delta)|\}, a^{\mathtt{c}}_\delta \rangle$ *default assumption* and denote it by a_δ. Moreover, let $A_\Delta = \{\langle a^{\mathtt{p}}_\delta, \{a^{\mathtt{j}}_{\delta,i} : i = 1, \ldots, |\mathtt{jus}(\delta)|\}, a^{\mathtt{c}}_\delta \rangle : \delta \in \Delta\}$ denote the set of default assumptions. Often, we will look at indexed defaults δ_j and abbreviate the default assumption a_{δ_j} by a_j. For a default assumption $\langle a^{\mathtt{p}}_\delta, \{a^{\mathtt{j}}_{\delta,i} : i = 1, \ldots, |\mathtt{jus}(\delta)|\}, a^{\mathtt{c}}_\delta \rangle \in A_\Delta$ we call $a^{\mathtt{p}}_\delta$ *prerequisitional*, $a^{\mathtt{j}}_{\delta,i}$ *justificational* ($\forall i$), and $a^{\mathtt{c}}_\delta$ *consequential assumption.* The sets $A_{\mathtt{p}} = \{a^{\mathtt{p}}_\delta : \delta \in \Delta\}$, $A_{\mathtt{j}} = \{a^{\mathtt{j}}_{\delta,i} : \delta \in \Delta, i = 1, \ldots, |\mathtt{jus}(\delta)|\}$ and finally $A_{\mathtt{c}} = \{a^{\mathtt{c}}_\delta : \delta \in \Delta\}$ denote the corresponding sets of prerequisitional, justificational, and consequential assumption, respectively. The set $A = A_{\mathtt{p}} \cup A_{\mathtt{j}} \cup A_{\mathtt{c}}$, which consists of pairwise distinct propositions such that $A \cap P = \emptyset$, is called the set of *assumptions.* The pair

$$\langle \Xi, A \rangle, \text{ were } \Xi = \Sigma \cup \bigcup_{\delta \in \Delta} t(\delta), \tag{2}$$

is called the *(propositional) argumentation system* associated with the default theory $\langle \Sigma, \Delta \rangle$ (cf. Section 3).

Example 4 *Let $\langle \Sigma, \Delta \rangle = \langle \{e \vee o\}, \{\frac{e : r}{r}, \frac{o : r}{r}\} \rangle$ be a default theory, then the corresponding argumentation system is $\langle \Xi, A \rangle = \langle \{e \vee o, a^{\mathtt{p}}_1 \to \neg e, a^{\mathtt{j}}_1 \to r, a^{\mathtt{c}}_1 \to r, a^{\mathtt{p}}_2 \to \neg o, a^{\mathtt{j}}_2 \to r, a^{\mathtt{c}}_2 \to r\}, \{a^{\mathtt{p}}_1, a^{\mathtt{p}}_2, a^{\mathtt{j}}_1, a^{\mathtt{j}}_2, a^{\mathtt{c}}_1, a^{\mathtt{c}}_2\} \rangle$, where the set of default assumptions is $A_\Delta = \{\langle a^{\mathtt{p}}_1, \{a^{\mathtt{j}}_1\}, a^{\mathtt{c}}_1 \rangle, \langle a^{\mathtt{p}}_2, \{a^{\mathtt{j}}_2\}, a^{\mathtt{c}}_2 \rangle\}$.*

Any subset $\alpha \subseteq A_c$ is called *consequential term.* Consequential terms will allow us, using the framework of propositional argumentation systems, to decide wether a given default theory has extensions, characterize all extensions, and wether a given formula is a skeptical or credulous consequence of the default theory.

3 PROPOSITIONAL ARGUMENTATION SYSTEMS

The major difference between argumentation systems (Haenni et al. 2000; Anrig 2000) and other nonmonotonic formalisms (Reiter 1980; McCarthy 1980; Moore 1985) is that in argumentation systems, all the inference is done with classical propositional logic.

Consider a propositional theory Ξ with formulas in $\mathscr{L}_V$, where $V = \{v_1, v_2, \ldots, v_m\}$. From V we choose a finite subset $A \subseteq V$ whose members are called *assumptions.* The other propositions in $P = V - A$ are called *non-assumables.* For convenience, we enumerate the assumptions, i.e. $A = \{a_1, \ldots, a_n\}$. The pair $\langle \Xi, A \rangle$ is called a (propositional) *argumentation system* and Ξ is said to be an *argumentation theory.* So we consider a fixed argumentation system $\langle \Xi, A \rangle$, with $\vdash$ as the derivability relation and Th the corresponding closure operator. Note that the case where $A = \emptyset$, corresponds to the usual propositional theory.

In an argumentation system $\langle \Xi, A \rangle$, the assumptions $A = \{a_1, \ldots, a_n\}$ of the argumentation theory Ξ are essential for expressing uncertain information. They represent uncertain events, unknown circumstances, or possible risks and outcomes. Formally, this is defined as follows:

Definition 3 *A* term α *is a subset of literals from* $A^{\pm}$*, so that if* $\ell \in \alpha$*, then* $\neg\ell \notin \alpha$*. The set of all terms is denoted by* T_A*.*

The non-assumables appearing in Ξ are treated as classical propositions for the given argumentation theory Ξ.

Definition 4 *A term* $\alpha \in \mathrm{T}_A$ *is called* inconsistent *relative to* Ξ *iff* $\alpha \cup \Xi$ *is unsatisfiable.* $\mathrm{I}(\Xi)$ *denotes the set of inconsistent terms relative to* Ξ*. An inconsistent term* α *is called* minimal *iff no proper subset of* α *is inconsistent relative to* Ξ*.* $\mu\mathrm{I}(\Xi)$ *denotes the set of all minimal inconsistent terms relative to* Ξ*, and is called the set of* minimal contradictions*.*

In the sequel we will write $\alpha, \Xi \vdash \bot$ to indicate that α is inconsistent, and $\alpha, \Xi \nvdash \bot$ to indicate that α is not inconsistent (relative to Ξ). $Th(\alpha, \Xi)$ denotes the deductive closure of $\alpha \cup \Xi$, and $\alpha, \Xi \vdash \psi$ is an abbreviation for $\alpha \cup \Xi \vdash \psi$. With this notation the set of minimal contradictions can be represented by $\mu\mathrm{I}(\Xi) = \mu\{\alpha \in \mathrm{T}_A : \alpha, \Xi \vdash \bot\}$.

Example 5 *Let* $\langle \Xi, A \rangle$ *be an argumentation system, where* $\Xi = \{a_1 \rightarrow p, a_2 \rightarrow q, \neg p, \neg q\}$ *is the argumentation theory and* $A = \{a_1, a_2\}$ *the set of assumptions. Then* $\mu\mathrm{I}(\Xi) = \{\{a_1\}, \{a_2\}\}$ *is the set of minimal contradictions.*

An argumentation system $\langle \Xi, A \rangle$ allows to reason about a specified propositional formula $\phi \in \mathscr{L}_V$, called *hypothesis*, with respect to A. The idea is to consider terms α which together with the argumentation theory Ξ imply this hypothesis ϕ.

Definition 5 ((Haenni et al. 2000), Theorem 2.7) *Let* $\langle \Xi, A \rangle$ *be an argumentation system. A term* α *is called a* supporting argument *for* ϕ *iff* $\alpha, \Xi \vdash \phi$ *and* $\alpha', \Xi \nvdash \bot$ *for all* $\alpha' \in \mathrm{T}_A$ *such that* $\alpha \subseteq \alpha'$*.*

$\mathrm{sp}(\phi; \Xi) = \{\alpha \in \mathrm{T}_A : \alpha, \Xi \vdash \phi$ and $\alpha', \Xi \nvdash \bot$ for all $\alpha' \in \mathrm{T}_A$ such that $\alpha \subseteq \alpha'\}$ denotes the corresponding set of supporting arguments. A supporting argument α for a hypothesis ϕ describes an environment in which the hypothesis is a logical consequence of the given argumentation theory, hence necessarily true. Moreover, extending this environment in such a way that α is still the case, does not lead to inconsistencies. For a general discussion of the requirement that $\alpha', \Xi \nvdash \bot$ for all $\alpha' \in \mathrm{T}_A$ such that $\alpha \subseteq \alpha'$ we refer to (Kohlas et al. 2002).

Example 6 *Let* $\Xi = \{a_1 \rightarrow p, a_2 \rightarrow q, p \rightarrow \neg q\}$ *and* $A = \{a_1, a_2\}$*. The supporting arguments for* p *are* $\mathrm{sp}(p; \Xi) = \{\{a_1, \neg a_2\}\}$*. Note that* $\{a_1\}, \Xi \vdash p$ *and* $\{a_1\}, \Xi \nvdash \bot$*, but* $\{a_1, a_2\}, \Xi \vdash \bot$*, hence* $\{a_1\}$ *is not a supporting argument for* p*. Similarly, we get* $\mathrm{sp}(q; \Xi) = \{\{\neg a_1, a_2\}\}$*.*

Definition 6 *Let* $\langle \Xi, A \rangle$ *be an argumentation system,* α *a term, and* $\mathcal{A} \subseteq A^{\pm}$ *a set of literals. The pair* $\langle \alpha, \mathcal{A} \rangle$ *is called* structure *iff*

(i) $\alpha, \Xi \nvdash \bot$*, and*
(ii) $\{\ell\} \cup \alpha, \Xi \nvdash \bot$*, for every* $\ell \in \mathcal{A}$*.*

α *is called the* anchor *and* $\mathcal{A}$ *the* set of irrelevant literals *of* $\langle \alpha, \mathcal{A} \rangle$ *w.r.t.* Ξ*.*

Note that $\{\ell\} \cup \alpha, \Xi \nvdash \bot$ for every $\ell \in \mathcal{A}$ does not guarantee that $\{\ell_i, \ell_j\} \cup \alpha, \Xi \nvdash \bot$ for $\ell_i, \ell_j \in \mathcal{A}$ and $i \neq j$. The utility of structures will become clear in Section 4.

We close our discussion of terms and scenarios and refer to (Haenni et al. 2000; Kohlas and Monney 1995; Kohlas et al. 1998; Kohlas and Monney 1993; Anrig et al. 1999; Haenni 1998; Haenni and Lehmann 2000) for readers wishing more details concerning this issue. For computational purposes, especially computing minimal contradictions and approximation techniques see (Haenni et al. 2000; Haenni 2001a).

4 EXTENSIONS AND TERMS

Here, we characterize extensions in the corresponding propositional argumentation system using structures $\langle \alpha, \mathcal{A} \rangle$ (cf. Definition 6), and provide, based on the anchor of some structures, an inductive procedure to compute extensions, if the original default theory has any. Here, only consequential terms (cf. Section 2.2) are allowed as anchor α, i.e. $\alpha \subseteq A_{\mathsf{c}}$ and the set of irrelevant literals $\mathcal{A}$ consists only of justificational assumptions, i.e. $\mathcal{A} \subseteq A_{\mathsf{j}}$. What follows is close in spirit to Lukaszewicz's (Lukaszewicz 1991) semantics for Reiter's default logic.

A pair $\langle \alpha, \mathcal{A} \rangle$, where $\alpha \subseteq A_{\mathsf{c}}$ is a consequential term and $\mathcal{A} \subseteq A_{\mathsf{j}}$ is a set of justificational assumptions is called *consequence-justification pair*, *CJ-pair* for short.

Definition 7 *Let $\langle \Sigma, \Delta \rangle$ be default theory, $\langle \Xi, A \rangle$ its corresponding argumentation system, $\alpha \subseteq A_{\mathsf{c}}$ a consequential term and $a = \langle a^{\mathsf{p}}, \{a_1^{\mathsf{j}}, \ldots, a_k^{\mathsf{j}}\}, a^{\mathsf{c}} \rangle$ a default assumption. We say that a is* applicable *with respect to α iff*

(i) $\{a^{\mathsf{p}}\} \cup \alpha \in \mathrm{I}(\Xi)$ *and*
(ii) $\{a_i^{\mathsf{j}}\} \cup \alpha \notin \mathrm{I}(\Xi)$ *for* $i = 1, \ldots, k$.

Note that we do not check if $\{a^{\mathsf{c}}\} \cup \alpha, \Xi \vdash \bot$. The idea is to obtain a sequence of consequences to add successively to the consequential term according to Definition 1 in an argumentation system.

Definition 8 *To each default assumption $a = \langle a^{\mathsf{p}}, \{a_1^{\mathsf{j}}, \ldots, a_k^{\mathsf{j}}\}, a^{\mathsf{c}} \rangle$ we assign a mapping, denoted by $a^{\prec}$, from CJ-pairs into CJ-pairs specified by*

(i) $a^{\prec}(\langle \alpha, \mathcal{A} \rangle) = \langle \{a^{\mathsf{c}}\} \cup \alpha, \{a_1^{\mathsf{j}}, \ldots, a_k^{\mathsf{j}}\} \cup \mathcal{A} \rangle$, *if $\langle \alpha, \mathcal{A} \rangle$ is a structure and a is applicable w.r.t. α;*
(ii) $a^{\prec}(\langle \alpha, \mathcal{A} \rangle) = \langle \alpha, \mathcal{A} \rangle$, *if $\langle \alpha, \mathcal{A} \rangle$ is a structure and a is not applicable w.r.t. α;*
(iii) $a^{\prec}(\langle \alpha, \mathcal{A} \rangle) = \langle \{\bot\}, \emptyset \rangle$, *otherwise.*

If a default assumption is applicable with respect to some consequential term, then we have to include its consequential assumption to the already given consequential term. But this inclusion may lead to inconsistencies. Consequently, we have to check if the obtained CJ-pair is still a structure.

Example 7 *Consider $\langle \{p\}, \{\frac{p:q}{\neg q}\} \rangle$ with $\Xi = \{p, a^{\mathsf{p}} \rightarrow \neg p, a^{\mathsf{j}} \rightarrow q, a^{\mathsf{c}} \rightarrow \neg q\}$. If we take the structure $\langle \{\top\}, \emptyset \rangle$, then $a = \langle a^{\mathsf{p}}, \{a^{\mathsf{j}}\}, a^{\mathsf{c}} \rangle$ is applicable w.r.t. $\{\top\}$. Thus $a^{\prec}(\langle \{\top\}, \emptyset \rangle) = \langle \{a^{\mathsf{c}}\}, \{a^{\mathsf{j}}\} \rangle$, but $\{a^{\mathsf{j}}\} \cup \{a^{\mathsf{c}}\}, \Xi \vdash \bot$, hence $\langle \{a^{\mathsf{c}}\}, \{a^{\mathsf{j}}\} \rangle$ is not a structure.*

Definition 9 *A structure $\langle \alpha, \mathcal{A} \rangle$ is called* accessible *(with respect to A_Δ) iff there is a sequence $\langle a_j \rangle = \langle a_1, a_2, \ldots, a_q \rangle$ of non-repeating default assumptions such that*

(i) $\langle \alpha_0, \mathcal{A}_0 \rangle = \langle \{\top\}, \emptyset \rangle$;
(ii) a_j *is applicable w.r.t.* α_{j-1} *and* $\langle \alpha_j, \mathcal{A}_j \rangle = a_j^{\prec}(\langle \alpha_{j-1}, \mathcal{A}_{j-1} \rangle)$ *for* $j = 1, 2, \ldots, q$;
(iii) $\langle \alpha_q, \mathcal{A}_q \rangle = \langle \alpha, \mathcal{A} \rangle$
(iv) *there is no default assumption $a_i \in A_\Delta$ which is applicable w.r.t. α but not included in $\langle a_j \rangle$.*

Such a $\langle a_j \rangle$ is called a generating sequence *of $\langle \alpha, \mathcal{A} \rangle$.*

If $\langle a_j \rangle$ is a generating sequence of a structure $\langle \alpha, \mathcal{A} \rangle$, then α contains all consequential assumptions of the default assumptions appearing in $\langle a_j \rangle$, $\mathcal{A}$ contains the union of all sets of justificational assumptions of the default assumptions appearing in $\langle a_j \rangle$, and $\langle \alpha, \mathcal{A} \rangle$ is still a structure. In particular $\alpha \subseteq A_{\mathsf{c}}$.

Example 8 *Consider the default theory $\langle \emptyset, \{\frac{\top:p}{p}, \frac{\top:q}{q}\} \rangle$ with corresponding $\Xi = \{\neg a_1^{\mathsf{p}}, a_1^{\mathsf{j}} \rightarrow p, a_1^{\mathsf{c}} \rightarrow p, \neg a_2^{\mathsf{p}}, a_2^{\mathsf{j}} \rightarrow q, a_2^{\mathsf{c}} \rightarrow q\}$. If we take the structure $\langle \{\top\}, \emptyset \rangle$, then $\langle a_1, a_2 \rangle$ and $\langle a_2, a_1 \rangle$ are generating sequences of the same structure $\langle \{a_1^{\mathsf{c}}, a_2^{\mathsf{c}}\}, \{a_1^{\mathsf{j}}, a_2^{\mathsf{j}}\} \rangle$.*

Obviously, there may be different generating sequences for the same structure. But two generating sequences of the same structure contain the same default assumptions, i.e. one is a permutation of the other (Berzati and Anrig 2001).

The definition of accessible structures is justified by the next theorem, which can be regarded as the dual result of Theorem 5.63 in (Lukaszewicz 1991). Duality in the sense that Lukaszewicz' (Lukaszewicz 1991) theorem is a semantic characterization of extensions and our result is a syntactic characterization of extensions. In fact the proof follows Lukaszewicz (Lukaszewicz 1991).

Theorem 1 (Berzati and Anrig 2001) *Let $\langle \Sigma, \Delta \rangle$ be a default theory and $\langle \Xi, A \rangle$ the corresponding argumentation system. There is a bijection between accessible structures of $\langle \Xi, A \rangle$ and consistent extensions of $\langle \Sigma, \Delta \rangle$.*

The theorem above states only that an accessible structure contains the information concerning some corresponding extension of the given default theory $\langle \Sigma, \Delta \rangle$. It says nothing about how the extension can be characterized. This observation motivates our next definition to simplify terminology.

Definition 10 *The anchor α of an accessible structure $\langle \alpha, \mathcal{A} \rangle$ is called* default term. *The set of all default terms is denoted by* $\mathrm{dt}(\Xi)$.

The notion of a default term permits to derive some corollaries of Theorem 1.

Corollary 1 (Berzati and Anrig 2001) *Let $\langle \Sigma, \Delta \rangle$ be a default theory with $P = \mathrm{Var}(\langle \Sigma, \Delta \rangle)$ and $\langle \Xi, A \rangle$ the corresponding argumentation system.*

(i) *Σ is inconsistent iff $\mu\mathrm{I}(\Xi) = \{\{\top\}\}$.*
(ii) *$\langle \Sigma, \Delta \rangle$ has no extension iff $\mathrm{dt}(\Xi) = \emptyset$ and $\mu\mathrm{I}(\Xi) \neq \{\{\top\}\}$.*
(iii) *$E = Th(\Sigma)$ is the only extension iff $\mathrm{dt}(\Xi) = \{\top\}$.*
(iv) *$E = Th((\alpha, \Xi)^{\downarrow P})$ is an extension of $\langle \Sigma, \Delta \rangle$ iff $\alpha \in \mathrm{dt}(\Xi)$.*

Now, that we have a complete characterization of extensions for a given default theory $\langle \Sigma, \Delta \rangle$ in terms of the default terms $\mathrm{dt}(\Xi) \subseteq 2^{A_c}$ of the corresponding argumentation system $\langle \Xi, A \rangle$, it remains to determine how those default terms can be computed. Interestingly, all we need to do that are the minimal contradictions $\mu\mathrm{I}(\Xi)$. The minimal contradictions $\mu\mathrm{I}(\Xi)$ are known to play a central role in argumentation systems, for example for computing supporting scenarios (Haenni et al. 2000) and in model-based diagnostics and reliability (Anrig and Kohlas 2002). Therefore a lot of effort has been made to develop efficient algorithms (Haenni et al. 2000; Haenni 2001b) to compute $\mu\mathrm{I}(\Xi)$. So default logic can profit from those efforts.

Theorem 2 (Berzati and Anrig 2001) *Let $\langle \Sigma, \Delta \rangle$ be a default theory and $\langle \Xi, A \rangle$ the corresponding argumentation system. A term $\alpha \subseteq A_c$ different from $\{\top\}$ is a default term iff there is an enumeration $a_1^c, \ldots, a_q^c$ of the consequential assumptions $a_j^c \in \alpha$ such that*

(i) $\{a_1^p\} \in \mathrm{I}(\Xi)$,
(ii) *For every $j = 1, \ldots, q-1$ we have* $\{a_{j+1}^p\} \cup \{a_1^c, \ldots, a_j^c\} \in \mathrm{I}(\Xi)$,
(iii) *For every $j = 1, \ldots, q$ and every i we have* $\{a_{ji}^j\} \cup \alpha \notin \mathrm{I}(\Xi)$,
(iv) *For every $a_l^p \in (A_p - \{a_1^p, \ldots, a_q^p\})$ we have*
  (a) $\{a_l^p\} \cup \alpha \notin \mathrm{I}(\Xi)$, or
  (b) $\{a_{li}^j\} \cup \alpha \in \mathrm{I}(\Xi)$ *for some i.*

Note that if $\alpha \in \mu\mathrm{I}(\Xi)$, then any term α', such that $\alpha \subseteq \alpha'$, is inconsistent, i.e. $\alpha' \in \mathrm{I}(\Xi)$. Clearly, only the minimal contradictions $\mu\mathrm{I}(\Xi)$ will be computed.

Theorem 2 indicates an iterative procedure for computing default terms given the translation of the default theory and the respective minimal contradictions. The following examples shows how.

Example 9 *We consider for each case of Corollary 1 a small example.*

- *For* $\langle \Sigma, \Delta \rangle = \langle \{p, \neg p\}, \{\frac{p:q}{q}\} \rangle$, $t(\langle \Sigma, \Delta \rangle) = \langle \{p, \neg p, a^p \rightarrow \neg p, a^j \rightarrow q, a^c \rightarrow q\}, \{a^p, a^j, a^c\} \rangle$ *and* $\mu\mathrm{I}(\Xi) = \{\{\top\}\}$. *Clearly,* $\Sigma \vdash \bot$.
- *For* $\langle \Sigma, \Delta \rangle = \langle \emptyset, \{\frac{\top:\neg p}{p}\} \rangle$ *we have* $t(\langle \Sigma, \Delta \rangle) = \langle \{\neg a^p, a^j \rightarrow \neg p, a^c \rightarrow p\}, \{a^p, a^j, a^c\} \rangle$ *and* $\mu\mathrm{I}(\Xi) = \{\{a^p\}, \{a^c, a^j\}\}$. *The only potentially possible default term different from $\{\top\}$ is $\{a^c\}$. $\{a^p\} \in \mu\mathrm{I}(\Xi)$, so point (i) of Theorem 2 is fulfilled. But $\{a^c, a^j\} \in \mu\mathrm{I}(\Xi)$, thus (iii) of Theorem 2 is violated, thus $\{a^c\}$ is no default term. Hence we conclude that there is no default term and $\langle \Sigma, \Delta \rangle$ has no extension.*
- *For* $\langle \Sigma, \Delta \rangle = \langle \{e \vee o\}, \{\frac{e:r}{r}\} \rangle$ *we have* $t(\langle \Sigma, \Delta \rangle) = \langle \{e \vee o, a^p \rightarrow \neg e, a^j \rightarrow r, a^c \rightarrow r\}, \{a^p, a^j, a^c\} \rangle$ *and* $\mu\mathrm{I}(\Xi) = \{\{\bot\}\}$. *As $\{a^p\} \notin \mu\mathrm{I}(\Xi)$ (point (i)), there is no default term different from $\{\top\}$ Indeed, this default theory has a single extension given by $E = Th(\{e \vee o\})$.*
- *Finally, for* $\langle \Sigma, \Delta \rangle = \langle \emptyset, \{\frac{\top:c}{\neg d}, \frac{\top:d}{\neg e}, \frac{\top:e}{\neg f}\} \rangle$ *we have* $\Xi = \{\neg a_1^p, a_1^j \rightarrow c, a_1^c \rightarrow \neg d, \neg a_2^p, a_2^j \rightarrow d, a_2^c \rightarrow \neg e, \neg a_3^p, a_3^j \rightarrow e, a_3^c \rightarrow \neg f\}$ *and* $\mu\mathrm{I}(\Xi) = \{\{a_1^p\}, \{a_2^p\}, \{a_3^p\}, \{a_1^c, a_2^j\}, \{a_2^c, a_3^j\}\}$. *There are three prerequisitional assumptions to start with:*
  - *First we start with $\{a_1^p\}$. The hypothesis that $\{a_1^c\}$ is a default term is rejected, since $\{a_3^p\} \in \mu\mathrm{I}(\Xi)$(point (a) of (iv) is violated). Thus $\{a_1^c, a_3^c\}$ could be a default term. And indeed $\{a_1^c, a_3^c\}$ is a default term, as (i) to (iv) is satisfied.*

- *Second we start with* $\{a_2^p\}$. *The hypothesis that* $\{a_2^c\}$ *is a default term is rejected, since* $\{a_1^p\} \in \mu I(\Xi)$*(point (a) of (iv) is violated). Thus* $\{a_2^c, a_1^c\}$ *could be a default term. But* $\{a_2^c, a_1^c\}$ *being a default term must be rejected, since* $\{a_1^c, a_2^j\} \in \mu I(\Xi)$ *(point (iii) is violated).*
- *Finally we start with* $\{a_3^p\}$. *We already know that* $\{a_1^c, a_3^c\}$ *is a default term. Thus we have to test if* $\{a_3^c, a_2^c\}$ *is a default term, as* $\{a_2^p\} \in \mu I(\Xi)$ *(*$\{a_3^c\}$ *being a default term violates point (a) of (iv)). Since* $\{a_2^c, a_3^j\} \in \mu I(\Xi)$*, the hypothesis* $\{a_3^c, a_2^c\}$ *being a default term is rejected (point (iii) is violated).*

So, $Th(\{\neg d, \neg f\})$ *is the only extension of this default theory.*

We need some notation to answer the question which formulas $\phi \in \mathscr{L}_P$ are provable in a default theory $\langle \Sigma, \Delta \rangle$, where $P = \mathrm{Var}(\langle \Sigma, \Delta \rangle)$. For a supporting argument $\alpha \in \mathrm{sp}(\phi; \Xi)$ we define $\alpha^+ = \alpha \cap A$ (i.e. we forget all literals $\ell \in \alpha$ such that $\ell = \neg a_i$ and $a_i \in A$).

Theorem 3 (Berzati and Anrig 2001) *Let* $\langle \Sigma, \Delta \rangle$ *be a default theory,* $\langle \Xi, A \rangle$ *the corresponding argumentation system, and* $\phi \in \mathscr{L}_P$ *a formula, where* $P = \mathrm{Var}(\langle \Sigma, \Delta \rangle)$.

(i) ϕ *is a credulous consequence of* $\langle \Sigma, \Delta \rangle$ *iff* $\{\alpha^+ : \alpha \in \mathrm{sp}(\phi; \Xi)\} \cap \mathrm{dt}(\Xi) \neq \emptyset$;
(ii) ϕ *is a skeptical consequence of* $\langle \Sigma, \Delta \rangle$ *iff* $\mathrm{dt}(\Xi) \subseteq \{\alpha^+ : \alpha \in \mathrm{sp}(\phi; \Xi)\}$.

Example 10 *The default theory* $\langle \Sigma, \Delta \rangle$*, where* $\Sigma = \{b \to d, c \to d\}$ *and* $\Delta = \{\frac{\top : \neg c}{b}, \frac{\top : \neg b}{c}\}$ *has two extensions* $E_1 = Th(\{b, d\})$ *and* $E_2 = Th(\{c, d\})$*. Clearly,* b *and* c *are credulous consequences of* $\langle \Sigma, \Delta \rangle$*, whereas* $b \vee c$ *and* d *are skeptical consequences. The translation returns* $\langle \Xi, A \rangle$ *with* $\Xi = \{b \to d, c \to d, \neg a_1^p, \neg a_2^p, a_1^j \to \neg c, a_2^j \to \neg b, a_1^c \to b, a_2^c \to c\}$*. One can verify that* $\mathrm{dt}(\Xi) = \{\{a_1^c\}, \{a_2^c\}\}$, $\{\alpha^+ : \alpha \in \mathrm{sp}(b; \Xi)\} = \{\{a_1^c\}\}$, $\{\alpha^+ : \alpha \in \mathrm{sp}(c; \Xi)\} = \{\{a_2^c\}\}$ *and* $\{\alpha^+ : \alpha \in \mathrm{sp}(b \vee c; \Xi)\} = \{\alpha^+ : \alpha \in \mathrm{sp}(d; \Xi)\} = \{\{a_1^c\}, \{a_2^c\}\}$.

5 CONCLUSION

We have provided a translation from default theories into argumentation systems which is linear in the number of defaults, from which we can decide if the given default theory has no extensions and if it has consistent extensions. Moreover, we are able to characterize all extensions, without leaving the monotone framework of propositional logic. All we need are assumptions. Finally, for any given formula we can decide if it is a skeptical or credulous consequence of the original default theory.

Acknowledgments

This work has been supported by grant No. 2000-061454.00 of the Swiss National Foundation for Research.

We would like to thank Philippe Besnard for his useful comments on earlier drafts of this article. We owe special thanks to the referees for their constructive and valuable suggestions which contributed to the final form of this article.

References

B. Anrig (2000). *Probabilistic Model-Based Diagnostics*. Ph.D. thesis. University of Fribourg (Switzerland), Institute of Informatics.

B. Anrig, R. Bissig, R. Haenni, J. Kohlas, and N. Lehmann (1999). *Probabilistic Argumentation Systems: Introduction to Assumption-Based Modeling with ABEL*. Technical Report 99-1. University of Fribourg, Institute of Informatics.

B. Anrig and J. Kohlas (2002). *Model-Based Reliability and Diagnostic: A Common Framework for Reliability and Diagnostics*. Technical Report 02-01. Department of Informatics, University of Fribourg.

R. Ben-Eliyahu and R. Dechter (1996). *Default reasoning using classical logic*. Artificial Intelligence. **84**: 113–149.

S. Benferhat, A. Saffiotti, and P. Smets (2000). *Belief Functions and Default Reasoning*. Artificial Intelligence. **122** (1–2): 1–69.

D. Berzati and B. Anrig (2001). *Embdding Default Theories in Propositional Argumentation Systems*. Technical Report 01–31. University of Fribourg, Departement of Informatics.

A. Bondarenko, P. M. Dung, R. A. Kowalski, and F. Toni (1997). *An Abstract, Argumentation-Theoretic Approach to Default Reasoning*. Artificial Intelligence. **93** (1–2): 63–101.

C. I. Chesñevar, A. G. Maguitman, and R. P. Loui (1998). *Logical Models of Argument*. draft for ACM Computing Surveys Logic of Arguments.

J. De Kleer (1986a). *An assumption-based TMS*. Artificial Intelligence. **28**: 127–162.

J. De Kleer (1986b). *Extending the ATMS*. Artificial Intelligence. **28**: 163–196.

K. D. Forbus and J. de Kleer (1993). *Building Problem Solvers*. Cambridge, Massachusetts: MIT Press.

R. Haenni (1998). *Modeling Uncertainty with Propositional Assumption-Based Systems*. In: S. Parson and A. Hunter, (eds.) *Applications of Uncertainty Formalisms*. Lecture Notes in Artifical Intelligence 1455. pp. 446–470. Springer.

R. Haenni (2001a). *Cost-bounded Argumentation*. International Journal of Approximate Reasoning. **26** (2): 101–127.

R. Haenni (2001b). *A Query-Driven Anytime Algorithm For Assumption-Based Reasoning*. Technical Report 01-26. Department of Informatics, University of Fribourg.

R. Haenni, J. Kohlas, and N. Lehmann (2000). *Probabilistic Argumentation Systems*. In: J. Kohlas and S. Moral, (eds.) *In: Handbook of Defeasible Reasoning and Uncertainty Management Systems*. volume 5: Algorithms for Uncertainty and Defeasible Reasoning. chapter 2, pp. 221–288. Kluwer, Dordrecht.

R. Haenni and N. Lehmann (2000). *Buidling Argumentation Systems on Set Constraint Logic*. In: B. Bouchon-Meunier, R. R. Yager, and L. A. Zadeh, (eds.) *Information, Uncertainty and Fusion*. pp. 393–406. Kluwer Academic Publishers.

A. Kean and G. Tsiknis (1992). *Assumption-based reasoning and clause management systems*. Computational Intelligence. **8**: 1–24.

A. Kean and G. Tsiknis (1993). *Clause management systems (CMS)*. Computational Intelligence. **9**: 11–40.

J. Kohlas, B. Anrig, R. Haenni, and P. Monney (1998). *Model-Based Diagnostics and Probabilistic Assumption-Based Reasoning*. Artificial Intelligence. **104**: 71–106.

J. Kohlas, D. Berzati, and R. Haenni (2002). *Probabilistic Argumentation Systems and Abduction*. Annals of Mathematics and Artificial Intelligence, Special Issue (AMAI). **34**: 177–195.

J. Kohlas, R. Haenni, and S. Moral (1999). *Propositional Information Systems*. Journal of Logic and Computation. **9 (5)**: 651–981.

J. Kohlas and P. Monney (1993). *Probabilistic Assumption-Based Reasoning*. In: D. Heckermann and A. Mamdani, (eds.) *Proc. 9th Conf. on Uncertainty in Artificial Intelligence, Washington*. pp. 485–491. Kaufmann, Morgan Publ.

J. Kohlas and P. Monney (1995). *A Mathematical Theory of Hints. An Approach to Dempster-Shafer Theory of Evidence*. volume 425 of *Lecture Notes in Economics and Mathematical Systems*. Springer-Verlag.

K. Konolige (1988). *On the Relation Between Default and Autoepistemic Logic*. Artificial Intelligence. **35** (3): 343–382.

V. Lifschitz (1990). *On open defaults*. In: *Proceedings of the Symposium on Computational Logics*. Springer Verlag (Berlin).

F. Lin and R. Reiter (1994). *Forget It!*. In: R. Greiner and D. Subramanian, (eds.) *Working Notes, AAAI Fall Symposium on Relevance*. pp. 154–159. American Association for Artificial Intelligence. Menlo Park, California: American Association for Artificial Intelligence.

T. Lukasiewicz (1998). *Probabilistic Deduction with Conditional Constraints over Basic Events*. In: A. G. Cohn, L. Schubert, and S. C. Shapiro, (eds.) *KR '98: Principles of Knowledge Representation and Reasoning*. pp. 380–391. San Francisco, California: Morgan Kaufmann.

W. Lukaszewicz (1991). *Non-Monotonic Reasoning; Formalization of Commonsense Reasoning*. Ellis Horwood series in Artificial Intelligence.

P. Marquis (2000). *Consequence Finding Algorithms*. In: J. Kohlas and S. Moral, (eds.) *In: Handbook of Defeasible Reasoning and Uncertainty Management Systems*. volume 5: Algorithms for Uncertainty and Defeasible Reasoning. chapter 1, pp. 41–145. Kluwer, Dordrecht.

J. McCarthy (1980). *Circumscription - a form of non-monotonic reasoning*. Artificial Intelligence. **13**: 27–39.

R. C. Moore (1985). *Semantical Considerations on Non-Monotonic Logic*. Artificial Intelligence. **25**: 75–94.

D. Nute (1994). *Defeasible Logic*. In: D. Gabbay, C. J. Hogger, and J. A. Robinson, (eds.) *Handbook of Logic in Artificial Intelligence and Logic Programming, Volume 3: Nonmonotonic Reasoning and Uncertain Reasoning*. pp. 353–395. Oxford: Oxford University Press.

D. Poole (1988). *A Logical Framework for Default Reasoning*. Artificial Intelligence. **36**: 27–47.

H. Prakken (1993). *An Argumentation Framework in Default Logic*. Annals of Mathematics and Artificial Intelligence. **9**: 91–132.

O. Raiman and J. De Kleer (1992). *A minimality maintenance system*. In: *Proceedings of the third International Conference on Principles of Knowledge Representation and Reasoning (KR'92)*. pp. 532–538.

R. Reiter (1980). *A logic for default reasoning*. Artificial Intelligence. **13**: 81–132.

Fuzzy Belief Revision

René Witte*
Institute for Program Structures and Data Organization (IPD)
Faculty of Informatics
University of Karlsruhe, Germany

Abstract

Fuzzy sets, having been the long-standing mainstay of modeling and manipulating imperfect information, are an obvious candidate for representing uncertain beliefs.

Unfortunately, unadorned fuzzy sets are too limited to capture complex or potentially inconsistent beliefs, because all too often they reduce to absurdities ("nothing is possible") or trivialities ("everything is possible").

However, we show that by combining the syntax of propositional logic with the semantics of fuzzy sets a rich framework for expressing and manipulating uncertain beliefs can be created, admitting Gärdenfors-style expansion, revision, and contraction operators and being moreover amenable to easy integration with conventional "crisp" information processing.

The model presented here addresses many of the shortcomings of traditional approaches for building fuzzy data models, which will hopefully lead to a wider adoptance of fuzzy technologies for the creation of information systems.

Keywords: fuzzy belief revision, fuzzy information systems, soft computing, fuzzy object-oriented data model

1 INTRODUCTION

Fuzzy information systems combine two basic ideas: the creation of a "mini-world" needed for building database and information systems, with the realization that this mini-world should encompass the imprecision and vagueness that exists in the real world being modeled. The so-called *soft computing* approach strikes for better, more efficient systems by incorporating this imprecision into the data models used for building systems.

A theoretical foundation for modeling imperfect information is the well-known fuzzy set theory by Zadeh [Zad65]. However, it is not obvious how to incorporate these fuzzy sets into data models used for building information systems. Although there has been extensive research in the area of fuzzy data models, especially for the relational and object-oriented model [BK95, Pet96, MS97, Cal97, YG99, LXHY99], none of the proposed fuzzy models had any noticible impact for building actual fuzzy applications.

A reason for this might be that fuzzy information systems have a different set of requirements. While classical, precise information systems usually capture the state of a changing world, fuzzy systems are especially helpful in identifying possible worlds by managing changing, imperfect knowledge.

Here, the integration of imperfect information, possibly from different sources, becomes important. For such applications, however, simple fuzzy sets are no longer sufficient, since they cannot distinguish a single imperfect information from a complex state obtained by combining them. This makes it impossible to provide semantically rich operators for modifying imperfect data.

In this paper, we present a model that combines syntactical features from propositional logic with the semantics of fuzzy sets. In this model, atoms are imperfect predicates that have an underlying interpretation in form of a fuzzy set. More complex fuzzy clauses and formulas can be constructed from these fuzzy atoms through a fuzzy conjunctive normal form. We show how the AGM postulates for expansion and revision can be adapted for this fuzzy model, and how they result in efficient operators that can be used for building fuzzy systems.

*Mail: Universität Karlsruhe, IPD, Am Fasanengarten 5, 76128 Karlsruhe, Germany. Email: witte@acm.org. Web: http://wwwipd.ira.uka.de/~witte/

2 MODELING IMPERFECT INFORMATION

Existing fuzzy data models usually incorporate fuzziness by (at least) adding a new primitive data type "fuzzy set", e.g. for object attributes. While this seems like an obvious extension for introducing fuzziness into an object-oriented data model, we identified several shortcomings with this idea.

The core problem is that this approach overloads a single fuzzy set with both syntax and semantics. Imperfect information first has a *structure* that results from processing single pieces of imperfect information — e.g., by combining imperfect requirements, collecting vague information from many heterogenous information sources, or processing fuzzy results from different heuristics or sensors. Secondly, both a single piece of imperfect information and a result obtained by combining them has a *semantic* interpretation that provides the relevant meaning for the fuzzy set within a system. By mixing both syntax and semantics together into a single fuzzy set, neither can be clearly identified. As a side-effect, it becomes extremely difficult to provide meaningful operators to modify imperfect data. For example, consider the case where conflicting information (like many witnesses describing a person's height conflictingly from "very short" to "very tall") results in a useless fuzzy interpretation (a membership function that is constant one or constant zero): when working only with the fuzzy set representation it is impossible to precisely identify the conflicting information and selectively remove data in order to maintain a consistent knowledge base, which is an important rationale for database and information system practitioners. Dubois and Prade [DP94] already pointed out that purely semantical representations, like the one offered by possibilistic belief revision, are poorer than syntactic ones.

Consequently, what we need is a representation model for imperfect information that allows to selectively identify single pieces of information, yet offers precise semantics in terms of fuzzy sets. Classical propositional logic offers the first part: the construction of complex propositions from primitives, the *atoms*, and their logical combination to *clauses* and *formulas*. By substituting precise atoms with fuzzy atoms, we can construct fuzzy propositions where each logical constituent has a semantic interpretation in form of a fuzzy set. Modifications then become possible on the syntactical level, e.g. by adding a new imprecise fact or vague requirement with a logical "and" operation, which leads to a new semantic interpretation in form of a fuzzy set.

For example, adding a new requirement b to an existing imperfect requirement a results in the logical structure $a \wedge b$ as well as the fuzzy interpretation $\mu_{a \wedge b}$. This makes it possible to identify conflicts and inconsistencies on both the structural and fuzzy set level — if, for example, a new requirement is incompatible with $a \wedge b$ but not a or b alone, one of these syntactical components can be removed and the fuzzy interpretation be re-computed. The following commutative diagram illustrates this principle; here $\mathcal{F}$ denotes a fuzzy conjunctive normal form (which we will define below), I its interpretation as a fuzzy set and OP a possible operation (which we will define later):

$$\begin{array}{ccc} \mathcal{F} & \xrightarrow{\text{OP}} & \mathcal{F}' \\ \downarrow I & & \downarrow I \\ \mu_{\mathcal{F}} & \xrightarrow{\text{OP}'} & \mu_{\mathcal{F}'} \end{array}$$

With this separation it was possible to incorporate concepts from belief revision into fuzzy data models, which we will present in section 3.

The representation model introduced here is a reworked and generalized version of the model proposed by Boss [Bos96] for the domain of fuzzy design databases.

2.1 THE FUZZY REPRESENTATION MODEL

We first present the basic definitions for fuzzy sets and their operators [Zad65, KF88]:

Definition 2.1 (Fuzzy Sets and Operators) *A fuzzy set μ of Ω is a mapping of the reference set Ω to the [0,1]-interval:*

$$\mu : \Omega \rightarrow [0, 1]$$

We denote the set of all fuzzy sets over Ω with $F(\Omega)$.

The intersection of two fuzzy sets μ_1 and μ_2, written $\mu_1 \cap \mu_2$, is defined by:

$$(\mu_1 \cap \mu_2)(\omega) := \min\{\mu_1(\omega), \mu_2(\omega)\}, \omega \in \Omega$$

The union of two fuzzy sets μ_1 and μ_2, written $\mu_1 \cup \mu_2$, is defined by:

$$(\mu_1 \cup \mu_2)(\omega) := \max\{\mu_1(\omega), \mu_2(\omega)\}, \omega \in \Omega$$

Finally, the complement of a fuzzy set μ, written $\overline{\mu}$, is defined by:

$$\overline{\mu}(\omega) := 1 - \mu(\omega), \omega \in \Omega$$

For the purpose of this paper we only consider possibilistic interpretation of fuzzy sets. In this interpretation, a fuzzy set μ models an imprecise concept (predicate) C_μ, and a particular value $\mu(\omega)$ describes the consistency of ω with C_μ:

$\mu(\omega) = 0$	then ω is definitely inconsistent with C_μ,
$\mu(\omega) = 1$	means, ω is consistent with C_μ without restriction of any kind,
$\mu(\omega) \in (0,1)$	gives the degree for the consistency of ω with C_μ.

Note that the case $\mu(\omega) = 1$ does *not* mean that C_μ is necessarily true of ω — it only means there is nothing that speaks against it.

We can now define our fuzzy representation model. As outlined above, we combine the structural features from propositional logic with the semantics of fuzzy sets.

The basic unit in this model is a *fuzzy atom*, which formalizes the notion of imprecise concepts introduced above:

Definition 2.2 (Fuzzy Concept Vocabulary, Fuzzy Atom) *Let a domain Ω be given. A fuzzy concept vocabulary over Ω is a triple $(\mathfrak{A}(\Omega), \mu., \mathcal{A}.)$, where $\mathfrak{A}(\Omega)$ is a countable set of symbols and $\mu., \mathcal{A}.$ are mappings*

$$\begin{aligned} \mu. &: \mathfrak{A}(\Omega) \to F(\Omega), \mathcal{A} \mapsto \mu_{\mathcal{A}} \\ \mathcal{A}. &: F(\Omega) \to \mathfrak{A}(\Omega), \mu \mapsto \mathcal{A}_\mu \end{aligned}$$

satisfying $\mu. \circ \mathcal{A}. = \mathrm{id}_{F(\Omega)}$.

The elements of $\mathfrak{A}(\Omega)$ are called fuzzy atomic concepts (or fuzzy atoms for short) over Ω. The fuzzy set $\mu_{\mathcal{A}}$ is called the interpretation of $\mathcal{A}$, and the atom $\mathcal{A}_\mu$ is called the (distinguished) name of μ.

Fuzzy atoms are somewhat similar to linguistic variables in that they decouple an imperfect information from its fuzzy set representation. The mapping $\mathcal{A}.$ is a technical requirement of our model, as we want to be able to name every fuzzy set within a system. Fuzzy atoms can be defined in a system lexicon that matches vague statements (e.g., "tall", "economical", "bendable") to fuzzy sets over a certain domain, but they can also stem from computations by fuzzy heuristics or provided by imprecise sensors.

A *fuzzy literal* is either an atom or the negation of an atom:

Definition 2.3 (Fuzzy Literal) *A fuzzy literal $\mathcal{L}$ over domain Ω is either a fuzzy atom $\mathcal{A} \in \mathfrak{A}(\Omega)$ or an expression of the form $\overline{\mathcal{A}}$ with $\mathcal{A} \in \mathfrak{A}(\Omega)$. The fuzzy atom $\overline{\mathcal{A}}$ is called the negation of $\mathcal{A}$, its interpretation $\mu_{\overline{\mathcal{A}}}$ is defined as*

$$\mu_{\overline{\mathcal{A}}}(\omega) = 1 - \mu_{\mathcal{A}}(\omega) \quad \forall \mathcal{A} \in \mathfrak{A}(\Omega), \omega \in \Omega$$

The set of all fuzzy literals over domain Ω is denoted by $\mathfrak{L}(\Omega)$.

Fuzzy literals can be joined together to form *fuzzy clauses*, which are interpreted as the logical disjunction of their constituent literals:

Definition 2.4 (Fuzzy Clause) *A fuzzy clause $\mathcal{K}$ over domain Ω is a finite set of fuzzy literals $\mathcal{L}_i \in \mathfrak{L}(\Omega)$, $1 \leq i \leq n$.*

We use the notation $\mathcal{K} = \mathcal{L}_1 \vee \mathcal{L}_2 \vee \ldots \vee \mathcal{L}_n$ if we want to emphasize the disjunctive nature of clauses.

The interpretation of a fuzzy clause $\mathcal{K}$ as a fuzzy set $\mu_{\mathcal{K}}$ is given by

$$\mu_{\mathcal{K}} \stackrel{\text{def}}{=} \begin{cases} \mu_\perp & \text{if } \mathcal{K} = \emptyset \\ \mu_{\mathcal{L}_1} \cup \mu_{\mathcal{L}_2} \cup \ldots \cup \mu_{\mathcal{L}_n} & \text{otherwise} \end{cases}$$

with $\mu_\perp(\omega) := 0 \quad \forall \omega \in \Omega$. The set of all fuzzy clauses over a domain Ω is denoted by $\mathfrak{K}(\Omega)$.

By combining fuzzy clauses with logical *and*-operations we get *fuzzy formulas*:

Definition 2.5 (Fuzzy Formula) *A fuzzy formula $\mathcal{F}$ over domain Ω is a finite set of fuzzy clauses $\mathcal{K}_i \in \mathfrak{K}(\Omega), 1 \leq i \leq m$. We use the notation $\mathcal{F} = \mathcal{K}_1 \wedge \ldots \wedge \mathcal{K}_m$ if we want to emphasize the conjunctive nature of formulas.*

The interpretation of a fuzzy formula $\mathcal{F}$ as a fuzzy set $\mu_{\mathcal{F}}$ is given by:

$$\mu_{\mathcal{F}} \stackrel{\text{def}}{=} \begin{cases} \mu_\top & \text{if } \mathcal{F} = \emptyset \\ \mu_{\mathcal{K}_1} \cap \mu_{\mathcal{K}_2} \cap \ldots \cap \mu_{\mathcal{K}_m} & \text{otherwise} \end{cases}$$

with $\mu_\top(\omega) := 1 \quad \forall \omega \in \Omega$. We denote the set of all fuzzy formulas over Ω with $\mathfrak{F}(\Omega)$.

We also call a fuzzy formula a *fuzzy conjunctive normal form* (FCNF).

Example (fuzzy formula) We illustrate the use of fuzzy formulas with a small example from a *fuzzy decision support system* for selecting a car based upon a number of vague requirements.

For this example we assume that a user demands the following features from a car:

$$\text{not(expensive)} \wedge (\text{economical} \vee \text{powerful})$$

Fuzzy atoms for these vague requirements with their corresponding fuzzy set interpretations might look like this:

	Mercedes	**Porsche**	**VW**	**Smart**
expensive	0.75	1.00	0.50	0.25
economical	0.50	0.25	0.75	1.00
powerful	0.75	1.00	0.50	0.00

If we use these definitions, the above requirement can be modeled as a fuzzy formula $\mathcal{F} = \{\mathcal{K}_1, \mathcal{K}_2\}$ with

$\mathcal{K}_1 = \{\mathcal{L}_1\}$, $\mathcal{K}_2 = \{\mathcal{L}_2, \mathcal{L}_3\}$, $\mathcal{L}_1 = \{\overline{\mathcal{A}_{\text{expensive}}}\}$, $\mathcal{L}_2 = \{\mathcal{A}_{\text{economical}}\}$, and $\mathcal{L}_3 = \{\mathcal{A}_{\text{powerful}}\}$. This fuzzy formula has a fuzzy set interpretation of $\{$*Mercedes/0.25, Porsche/0.0, VW/0.5, Smart/0.75*$\}$, i.e., a *Smart* would be a good (albeit not perfect) candidate for the stated requirements.

The language of fuzzy literals, fuzzy clauses, and fuzzy formulas permits the aggregation of arbitrary pieces of imperfect information, however inconsistent they may be. This is an important feature, because the system does not have to reject inconsistent information out of hand (as is the case with conventional information systems), but can rather attempt to restore consistency by giving preference to some pieces over others. This will be discussed later on in more detail. For now, we just define a notion of consistence:

Definition 2.6 (Degree of Consistency) *The degree of consistency C of a fuzzy clause $\mathcal{K} \in \mathfrak{K}(\Omega)$ is defined as:*

$$C(\mathcal{K}) \stackrel{\text{def}}{=} \sup_{\omega \in \Omega} \{\mu_{\mathcal{K}}(\omega)\}.$$

Analogously, we define the degree of consistency for a fuzzy formula $\mathcal{F} \in \mathfrak{F}(\Omega)$:

$$C(\mathcal{F}) \stackrel{\text{def}}{=} \sup_{\omega \in \Omega} \{\mu_{\mathcal{F}}(\omega)\}.$$

For a pair of fuzzy formulas $\mathcal{F}_1, \mathcal{F}_2$, we define the mutual consistency of $\mathcal{F}_1$ and $\mathcal{F}_2$ as $C(\mathcal{F}_1, \mathcal{F}_2) \stackrel{\text{def}}{=} C(\mathcal{F}_1 \cup \mathcal{F}_2)$ and say that $\mathcal{F}_1$ and $\mathcal{F}_2$ are γ-consistent for $\gamma \in [0,1]$, if $C(\mathcal{F}_1, \mathcal{F}_2) \geq \gamma$.

We will also need the *negation* of a fuzzy formula:

Definition 2.7 (Negation of Fuzzy Formulas) *The negation $\overline{\mathcal{F}}$ of a fuzzy formula $\mathcal{F} \in \mathfrak{F}(\Omega)$ is computed by pushing the negation to its constituent atoms (using de Morgan's laws) and re-computing the conjunctive normal form.*

However, this definition can lead to an exponential operator execution time. But we can provide a simpler negation for the implementation: for some applications it is sufficient to create a negated fuzzy formula from the negation of its fuzzy interpretation, instead of its logical structure, which can be computed very efficiently.

3 FUZZY BELIEF REVISION

We now address one of the most important points within our representation model: *modifications*. Most existing data models for fuzzy object-oriented systems are only concerned with structural fuzziness, introduced at various points within an object-oriented model (e.g., attributes, objects, inheritance). Information, however, is not static. It is a typical feature of information systems that data changes over time. Thus, a fuzzy object-oriented data model must address modifications of imperfect data. However, we found that there are two typical problems that arise when processing fuzzy information:

- Computations are application specific. The challenge here is to define operators that are abstract enough to be used within a multitude of object-oriented fuzzy information systems, yet specific enough to be still useful.

- Combining different pieces of imperfect information can easily lead to meaningless results. Especially in the case of inconsistent information, fuzzy operations often lead to useless results like "everything is possible" or "nothing is possible".

Since we introduced a new representation model with our fuzzy formulas, we now have to define suitable operators for modifying them. As fuzzy clauses and formulas are sets, modifications could be done by simply adding or removing fuzzy literals or clauses. Simple set operators, however, are not concerned with semantics — clauses added to a fuzzy formula in this way can easily lead to inconsistencies in the fuzzy set interpretation. But maintaining consistency is one of the prime functions of database and information systems.

As mentioned above, we aim to support applications that require the management of complex imperfect states. Examples of systems we want to model are: a fuzzy decision support system that manages vague requirements, a fuzzy reverse engineering framework that maintains information about system artifacts, and a fuzzy noun phrase coreference resolution system that computes and combines results from fuzzy heuristics. Modifications within these applications are not simple updates (in the sense of Katsuno and Mendelzon [KM91]), but rather revisions, where we collect more and more information about an existing or emerging world.

Consequently, our solution is to define fuzzy operators based on Gärdenfors-style belief revision [AGM85, Gär88, Gär92]. They are called γ-expansion, γ-revision, γ-contraction, and allow for modifying a complex fuzzy formula while maintaining a specified degree of consistency γ (in this paper, we only consider γ-expansion and γ-revision). In the case of γ-revision for example, a new piece of information is either rejected, merged with all existing information, or added while removing some fuzzy clauses that are inconsistent to the new fuzzy formula within a specified degree γ.

These operators have the following semantics:

Expansion: A γ-*expansion* adds new information (expressed as a fuzzy formula) to an existing piece of information (again a fuzzy formula), without removing any existing information while ensuring that the resulting fuzzy formula reaches a consistency degree of at least γ. If this is not possible, the new information is *rejected.*

Revision: The γ-*revision* operation always adds new information, if its consistency degree reaches at least γ. Existing information (fuzzy clauses) may be removed in case there is a partial or complete inconsistency.

Contraction: The γ-*contraction* operation removes information from a knowledge set, in a way that preserves as much of the existing information as possible while ensuring that the result's consistency degree reaches at least γ.

In the fuzzy decision support system example, we collect requirements in form of fuzzy formulas. Adding a new requirement (FCNF) always results in a potential conflict — for example, while it is quite understandable that someone wishes for an impressive, big, and rather new car with lots of horsepower, adding a requirement that it must also be very inexpensive will most likely result in a conflict. Traditional fuzzy models would just return a fuzzy set that is close or equal to zero for all cars in the domain. This is rather useless, since it does neither show *why* there is a conflict nor *which* pieces of information actually cause the conflict. Our fuzzy γ-revision, however, can add a new requirement while maintaining a consistent state by showing exactly which existing requirements participate in the conflict and removing some of them.

The following sections show how the AGM postulates can be extended for our fuzzy model. Since we do not consider inference, we formulate the postulates and operators for individual formulas rather then deductively closed sets. This makes our fuzzy belief revision comparable to the base revision [Neb91, Neb92] known from propositional logic. One reason for not considering inference is that it simply is too costly for an (object-oriented) database and information system environment, where efficient implementations of operators are paramount. Another reason is that we are very concerned with the semantic *consistency* of information, which is defined on a formula's model. We therefore examined how syntactic revision can be performed in absence of an inference mechanism by directly checking the semantic interpretation of a fuzzy formula through the consistency operator C defined above. This consistency operator can be implemented very efficiently on the fuzzy set interpretations. We therefore based the notion of consistency on the interpretation of a fuzzy formula when we translated the AGM postulates for our model, and subsequently examined the features of operators satisfying these translated postulates.

3.1 FUZZY EXPANSION

Expansion is a monotonous extension of a fuzzy formula $\mathcal{F}$ by a fuzzy formula $\mathcal{G}$, i.e., none of the fuzzy clauses in $\mathcal{F}$ are removed. More precisely, we have:

Definition 3.1 (γ-Expansion) *A γ-expansion operator $+_\gamma$ is a mapping that takes two fuzzy formulas $\mathcal{F}, \mathcal{G} \in \mathfrak{F}(\Omega)$ and a value $\gamma \in [0,1]$ to a new fuzzy formula in $\mathfrak{F}(\Omega)$:*

$$+_\gamma : \mathfrak{F}(\Omega) \times \mathfrak{F}(\Omega) \to \mathfrak{F}(\Omega)$$

and satisfies the expansion postulates $E_\gamma 1$–$E_\gamma 6$ (see below).

The semantics of the γ-expansion operator is defined by the following postulates, here $\mathcal{F}, \mathcal{G}$ denote fuzzy formulas from $\mathfrak{F}(\Omega)$:

E_γ1 (Stability) The result of a γ-expansion is always a fuzzy formula:

$$\mathcal{F} +_\gamma \mathcal{G} \in \mathfrak{F}(\Omega).$$

E_γ2a (Success) If the consistency degree of $\mathcal{F} \cup \mathcal{G}$ is at least γ, the γ-expansion is successful:

$$\text{If } C(\mathcal{F} \cup \mathcal{G}) \geq \gamma, \text{ then } \mathcal{G} \subseteq \mathcal{F} +_\gamma \mathcal{G}.$$

E_γ2b (Failure) If the consistency degree of $\mathcal{F} \cup \mathcal{G}$ is less than γ, the original formula remains unchanged:

$$\text{If } C(\mathcal{F} \cup \mathcal{G}) < \gamma, \text{ then } \mathcal{F} +_\gamma \mathcal{G} = \mathcal{F}.$$

E_γ3 (Expansion) The set of information expands:

$$\mathcal{F} +_\gamma \mathcal{G} \supseteq \mathcal{F}.$$

E_γ4 (Invariance) If the new information $\mathcal{G}$ is already known, invariance holds:

$$\mathcal{G} \subseteq \mathcal{F} \Rightarrow \mathcal{F} +_\gamma \mathcal{G} = \mathcal{F}.$$

E_γ5 (Monotonicity) Let $\mathcal{F}' \subseteq \mathcal{F}$. If the expansion $\mathcal{F} +_\gamma \mathcal{G}$ is successful or the expansion $\mathcal{F}' +_\gamma \mathcal{G}$ fails, monotonicity holds:

$$\text{If } (\mathcal{G} \subseteq \mathcal{F} +_\gamma \mathcal{G}) \vee (\mathcal{G} \not\subseteq \mathcal{F}' +_\gamma \mathcal{G}),$$
$$\text{then } \mathcal{F}' +_\gamma \mathcal{G} \subseteq \mathcal{F} +_\gamma \mathcal{G}.$$

E_γ6 (Predictability) The result of a γ-expansion must only contain fuzzy clauses from $\mathcal{F}$ or $\mathcal{G}$:

$$\mathcal{F} +_\gamma \mathcal{G} \subseteq \mathcal{F} \cup \mathcal{G}.$$

It is easy to see that these postulates admit only a single expansion operator, namely the one defined by

$$\mathcal{F} +_\gamma \mathcal{G} \stackrel{\text{def}}{=} \begin{cases} \mathcal{F} \cup \mathcal{G} & \text{if } C(\mathcal{F} \cup \mathcal{G}) \geq \gamma, \\ \mathcal{F} & \text{otherwise.} \end{cases}$$

Thus, in case of fuzzy formulas, γ-expansion is actually a quite simple operation: the result is the union of the clause sets of both fuzzy formulas if the union reaches the consistency degree γ. Otherwise, the result is the original formula $\mathcal{F}$, i.e., the new information $\mathcal{G}$ is rejected.

The main difference between this operation and the classical expansion is that we do not always add the new information. Since our main goal is to guarantee consistency within an information system, we reject new information if it is not consistent to the existing information within a specified degree γ.

Example (Fuzzy Expansion) We continue the fuzzy decision support example. Within this context, the fuzzy expansion operator can be used to implement a simple decision support algorithm that adds new requirements while making sure they are compatible with all existing requirements. Suppose a user started with the requirement not(*expensive*), represented by a fuzzy formula $\mathcal{F}_1 = \{\mathcal{K}_1\}$, with $\mathcal{K}_1 = \{\overline{\mathcal{A}_{\text{expensive}}}\}$. He now demands that the car should also be *economical*, represented by a fuzzy clause $\mathcal{K}_2 = \{\mathcal{A}_{\text{economical}}\}$. Adding this request through γ-expansion with a consistency degree of 0.75:

$$\mathcal{F}_2 = \mathcal{F}_1 +_{0.75} \{\mathcal{K}_2\} = \{\mathcal{K}_1, \mathcal{K}_2\}$$

leads to a new fuzzy formula $\mathcal{F}_2$, i.e., the expansion is successful, and its fuzzy set interpretation $\mu_{\mathcal{F}_2}$ is {*Mercedes/0.25, Porsche/0.0, VW/0.5, Smart/0.75* }. So, a good candidate for the stated requirements would be a *Smart*. Impressed by this result, the user requests that the car should also be *spacious*, represented by a fuzzy atom $\mathcal{A}_{\text{spacious}}$ that has the following fuzzy set interpretation:

	Mercedes	**Porsche**	**VW**	**Smart**
spacious	1.0	0.25	0.75	0.0

However, the γ-expansion operator rejects this request, i.e., the operation $\mathcal{F}_2 +_{0.75} \{\mathcal{K}_3\}$ results in an unchanged $\mathcal{F}_2$, since adding the new requirement would lead to a fuzzy set interpretation that has a consistency degree of only $0.5 < \gamma = 0.75$.

3.2 FUZZY REVISION

So far we rejected all inconsistent information to guarantee the integrity of our knowledge base. But this not a generally feasible approach, since most applications cannot turn away all new information, just because they do not fit into the established world. In these cases it becomes necessary to remove some existing information to preserve a consistent state. Because of this requirement, γ-revision is a *non-monotonous* operation. However, it is not practical to simply remove all existing information, which would be a trivial way to always ensure a consistent belief state. What makes belief revision a hard problem is that there is usually no unique result that incorporates all new information while preserving as much existing information as possible. Since our model is designed to work with information systems, efficiency becomes a prime objective (most of the more theoretical oriented research in classical belief revision would result in very inefficient implementations).

We begin again by defining a formal notion of γ-revision and then show the possible revision operators:

Definition 3.2 (γ-Revision) *A γ-revision operator $\oplus_\gamma$ is a mapping that takes two fuzzy formulas $\mathcal{F}, \mathcal{G} \in \mathfrak{F}(\Omega)$ and a value $\gamma \in [0, 1]$ to a fuzzy formula from $\mathfrak{F}(\Omega)$:*

$$\oplus_\gamma : \mathfrak{F}(\Omega) \times \mathfrak{F}(\Omega) \to \mathfrak{F}(\Omega)$$

and satisfies the revision postulates $R_\gamma 1$–$R_\gamma 6$ (see below).

We now present our translation of the original revision postulates by Gärdenfors. To make it easier to compare our postulates with the original, we kept the original numbering, adding letter indices in cases where we splitted a single postulate.

Here, $\mathcal{F}, \mathcal{G}, \mathcal{H}$ are fuzzy formulas over $\mathfrak{F}(\Omega)$:

$R_\gamma 1$ (Stability) The result of a revision operation is always a fuzzy formula:

$$\mathcal{F} \oplus_\gamma \mathcal{G} \in \mathfrak{F}(\Omega).$$

$R_\gamma 2a$ (Priority) If the new fuzzy formula $\mathcal{G}$ has a consistency degree of at least γ, then it must be contained in the result:

$$\text{If } C(\mathcal{G}) \geq \gamma, \text{ then } \mathcal{G} \subseteq \mathcal{F} \oplus_\gamma \mathcal{G}.$$

$R_\gamma 2b$ (Failure) If the consistency degree of the new fuzzy formula is less than γ, the original formula remains unchanged:

$$\text{If } C(\mathcal{G}) < \gamma, \text{ then } \mathcal{F} \oplus_\gamma \mathcal{G} = \mathcal{F}.$$

$R_\gamma 3$ (Predictability) The result of a revision contains only fuzzy clauses from $\mathcal{F}$ or $\mathcal{G}$:

$$\mathcal{F} \oplus_\gamma \mathcal{G} \subseteq \mathcal{F} \cup \mathcal{G}.$$

$R_\gamma 4$ (Compatibility) If the fuzzy clauses of the new formula are compatible with the existing clauses, the expansion is a subset of the revision:

$$\text{If } \mathcal{G} \subseteq \mathcal{F} +_\gamma \mathcal{G} \text{ and } C(\mathcal{F} +_\gamma \mathcal{G}) \geq \gamma,$$
$$\text{then } \mathcal{F} \oplus_\gamma \mathcal{G} \supseteq \mathcal{F} +_\gamma \mathcal{G}.$$

$R_\gamma 5$ (Consistency) In case of a successful revision, the consistency degree must reach at least γ:

$$\text{If } C(\mathcal{G}) \geq \gamma, \text{ then } C(\mathcal{F} \oplus_\gamma \mathcal{G}) \geq \gamma.$$

$R_\gamma 6$ (Identity) The interpretation of the result does not depend on the syntactic form of $\mathcal{G}$:

$$\text{If } \mu_\mathcal{G} = \mu_\mathcal{H}, \text{ then } \mu_{\mathcal{F} \oplus_\gamma \mathcal{G}} = \mu_{\mathcal{F} \oplus_\gamma \mathcal{H}}.$$

These are the so-called *basic postulates* that must be satisfied by every revision operator. There are two supplementary postulates that control iterative changes:

$R_\gamma 7$ If two fuzzy formulas $\mathcal{G}$ and $\mathcal{H}$ can be successfully added to a formula $\mathcal{F}$ by revision and expansion, the result is a superset of the revision of $\mathcal{F}$ with $\mathcal{G} \cup \mathcal{H}$:

$$\text{If } \mathcal{G} \cup \mathcal{H} \subseteq (\mathcal{F} \oplus_\gamma \mathcal{G}) +_\gamma \mathcal{H}, \text{ then}$$
$$\mathcal{F} \oplus_\gamma (\mathcal{G} \cup \mathcal{H}) \subseteq (\mathcal{F} \oplus_\gamma \mathcal{G}) +_\gamma \mathcal{H}.$$

$R_\gamma 8$ Extension of postulate $R_\gamma 4$:

$$\text{If } \mathcal{G} \cup \mathcal{H} \subseteq (\mathcal{F} \oplus_\gamma \mathcal{G}) +_\gamma \mathcal{H}, \text{ then}$$
$$(\mathcal{F} \oplus_\gamma \mathcal{G}) +_\gamma \mathcal{H} \subseteq \mathcal{F} \oplus_\gamma (\mathcal{G} \cup \mathcal{H}).$$

The semantics of most postulates is straightforward. Noteworthy is postulate $R_\gamma 6$, which ensures that the interpretation of the result of a γ-revision is independent of the syntax of the involved fuzzy formulas: if two fuzzy formulas have a different syntax but the same interpretation, the results of the revisions with these two formulas must also have the same interpretation.

There are a number of additional properties that follow from the postulates: firstly, if the γ-expansion $\mathcal{F} +_\gamma \mathcal{G}$ is successful, the result of the γ-revision $\mathcal{F} \oplus_\gamma \mathcal{G}$ is equal to the expansion. Secondly, if the new information is already known ($\mathcal{G} \subseteq \mathcal{F}$) and satisfies the consistency requirement ($C(\mathcal{F}) \geq \gamma$), then $\mathcal{F}$ remains unchanged: $(\mathcal{G} \subseteq \mathcal{F}) \wedge (C(\mathcal{F}) \geq \gamma) \Rightarrow \mathcal{F} \oplus_\gamma \mathcal{G} = \mathcal{F}$, i.e., the γ-revision is invariant with respect to known information. And thirdly, if an operation satisfies both supplementary postulates, the sets $(\mathcal{F} \oplus_\gamma \mathcal{G}) +_\gamma \mathcal{H}$ and $\mathcal{F} \oplus_\gamma (\mathcal{G} \cup \mathcal{H})$ are actually equal: If $\mathcal{G} \subseteq \mathcal{F} \oplus_\gamma \mathcal{G}$ and $\mathcal{H} \subseteq (\mathcal{F} \oplus_\gamma \mathcal{G}) +_\gamma \mathcal{H}$, then $(\mathcal{F} \oplus_\gamma \mathcal{G}) +_\gamma \mathcal{H} = \mathcal{F} \oplus_\gamma (\mathcal{G} \cup \mathcal{H})$. We omit the proofs for these claims since they are quite straightforward.

There are two additional properties, which we will need for the following revision theorem:

Lemma 3.3 *Let $\mathcal{F} \in \mathfrak{F}(\Omega)$ and $\mu \in F(\Omega) - \{\mu_\perp\}$ arbitrary. Then a fuzzy formula $\mathcal{G} \in \mathfrak{F}(\Omega)$ exists with $\mathcal{F} \cap \mathcal{G} = \emptyset$ and $\mu_\mathcal{G} = \mu$.*

Proof. *It is easy to see that there are infinitely many fuzzy clauses $\mathcal{K} \in \mathfrak{K}(\Omega)$ with $\mu_\mathcal{K} = \mu$. Only a finite number of these can be contained in $\mathcal{F}$. Therefore, select a clause $\mathcal{K} \in \mathfrak{K}(\Omega) - \mathcal{F}$ with $\mu_\mathcal{K} = \mu$ and let $\mathcal{G} := \{\mathcal{K}\}$.* ∎

Lemma 3.4 *Let $\mathcal{F}, \mathcal{G}_1, \mathcal{G}_2 \in \mathfrak{F}(\Omega)$ and $\gamma \in [0,1]$ arbitrary. If $\mu_{\mathcal{G}_1} = \mu_{\mathcal{G}_2}$ and $C(\mathcal{G}_1) = C(\mathcal{G}_2) \geq \gamma$, then*

$$(\mathcal{F} \oplus_\gamma \mathcal{G}_1) \cup \mathcal{G}_2 = (\mathcal{F} \oplus_\gamma \mathcal{G}_2) \cup \mathcal{G}_1$$

If additionally $\mathcal{F}, \mathcal{G}_1, \mathcal{G}_2$ are pairwise disjunct, it holds that

$$(\mathcal{F} \oplus_\gamma \mathcal{G}_1) - \mathcal{G}_1 = (\mathcal{F} \oplus_\gamma \mathcal{G}_2) - \mathcal{G}_2$$

Proof. *Since $\mu_{\mathcal{G}_1} = \mu_{\mathcal{G}_2}$ it holds that $C((\mathcal{F} \oplus_\gamma \mathcal{G}_1) \cup \mathcal{G}_2) = C((\mathcal{F} \oplus_\gamma \mathcal{G}_1) \cup \mathcal{G}_1) = C(\mathcal{F} \oplus_\gamma \mathcal{G}_1) \geq \gamma$, and analogously $C((\mathcal{F} \oplus_\gamma \mathcal{G}_2) \cup \mathcal{G}_1) \geq \gamma$. From $R_\gamma 7$ and $R_\gamma 8$ it follows that*

$$\begin{aligned}(\mathcal{F} \oplus_\gamma \mathcal{G}_1) \cup \mathcal{G}_2 &= \mathcal{F} \oplus_\gamma (\mathcal{G}_1 \cup \mathcal{G}_2) \\ &= (\mathcal{F} \oplus_\gamma \mathcal{G}_2) \cup \mathcal{G}_1\end{aligned}$$

and therefore the proposition. ∎

The next step is to obtain a more precise description of the possible revision operators satisfying $R_\gamma 1$–$R_\gamma 8$. It turns out that the following simple characterization can be given:

Theorem 3.5 (Revision theorem) *Let $\mathcal{F} \in \mathfrak{F}(\Omega)$ and $\gamma \in [0,1]$ be arbitrary, but fixed. A mapping $\mathcal{F}^{\oplus_\gamma} : \mathfrak{F}(\Omega) \to \mathfrak{F}(\Omega)$, $\mathcal{G} \mapsto \mathcal{F} \oplus_\gamma \mathcal{G}$ satisfies the postulates $R_\gamma 1$–$R_\gamma 8$ if and only if an enumeration $2^\mathcal{F} = \{\mathcal{F}_0, \mathcal{F}_1, \ldots, \mathcal{F}_{2^{|\mathcal{F}|}-1}\}$ of the power set of $\mathcal{F}$ exists with*

$$\mathcal{F} \oplus_\gamma \mathcal{G} = \begin{cases} \mathcal{F} & \text{if } C(\mathcal{G}) < \gamma, \\ \mathcal{F} \cup \mathcal{G} & \text{if } C(\mathcal{F} \cup \mathcal{G}) \geq \gamma, \\ \mathcal{F}_i \cup \mathcal{G} & \text{otherwise} \end{cases}$$

where $i = \min\{j | C(\mathcal{G} \cup \mathcal{F}_j) \geq \gamma\}$.

Proof. *The direction "$\Leftarrow$" (i.e., such an operator satisfies postulates $R_\gamma 1$–$R_\gamma 8$) is easy to see.*

Therefore, we only show the direction "$\Rightarrow$". The cases $C(\mathcal{G}) < \gamma$ and $C(\mathcal{F} \cup \mathcal{G}) \geq \gamma$ are trivial. The interesting case is $C(\mathcal{G}) \geq \gamma \wedge C(\mathcal{F} \cup \mathcal{G}) < \gamma$. Let $\oplus_\gamma$ be a fixed operator satisfying $R_\gamma 1$–$R_\gamma 8$. Define an enumeration $\mathcal{F}_0, \mathcal{F}_1, \ldots, \mathcal{F}_{2^{|\mathcal{F}|}-1}$ of the sub-formulas of $\mathcal{F}$ by induction:

$$\begin{aligned}\mathcal{F}_0 &:= \mathcal{F} \\ \mathcal{F}_{i+1} &:= \begin{cases} (\mathcal{F} \oplus_\gamma \mathcal{G}_i) - \mathcal{G}_i & \text{if } C(\mathcal{G}_i) \geq \gamma \\ \textit{else arbitrary from } 2^\mathcal{F} - \{\mathcal{F}_0, \ldots, \mathcal{F}_i\} \end{cases}\end{aligned}$$

whereas $\mathcal{G}_i \in \mathfrak{F}(\Omega)$ is an arbitrary formula with $\mathcal{F} \cap \mathcal{G}_i = \emptyset$ and

$$\mu_{\mathcal{G}_i}(\omega) = \begin{cases} 1 & \text{if } \mu_{\mathcal{F}_j}(\omega) < \gamma \text{ for } 0 \leq j \leq i, \\ 0 & \text{else} \end{cases}$$

The existence of such $\mathcal{G}_i$ and the independence of the $\mathcal{F}_i$ from the selected $\mathcal{G}_i$ follows from the Lemmas 3.3 and 3.4.

Now let $\mathcal{G} \in \mathfrak{F}(\Omega)$ with $C(\mathcal{G}) \geq \gamma$, $C(\mathcal{F} \cup \mathcal{G}) < \gamma$ be arbitrary selected. Then the set $\{j | C(\mathcal{G} \cup \mathcal{F}_j) \geq \gamma\}$ is not empty (it contains at least the j with $\mathcal{F}_j = \emptyset$), and it does not contain $\mathcal{F}_0 = \mathcal{F}$. Therefore, $i := \min\{j | C(\mathcal{G} \cup \mathcal{F}_j) \geq \gamma\}$ is well-defined, and it holds that $i > 0$. We have to show that $\mathcal{F} \oplus_\gamma \mathcal{G} = \mathcal{G} \cup \mathcal{F}_i$.

Let $\varepsilon > 0$ arbitrary. From $R_\gamma 5$ follows $C(\mathcal{F} \oplus_\gamma \mathcal{G}) \geq \gamma$, hence there exists an $\omega \in \Omega$ with $\mu_{\mathcal{F} \oplus_\gamma \mathcal{G}}(\omega) \geq \gamma - \varepsilon$. From $R_\gamma 2a$ then follows $\mu_{\mathcal{G}}(\omega) \geq \gamma - \varepsilon$. Moreover, it follows from the selection of i that for $0 \leq j < i$ holds: $C(\mathcal{G} \cup \mathcal{F}_j) < \gamma$, in particular $\mu_{\mathcal{F}_j}(\omega) < \gamma$. For this reason $\mu_{\mathcal{G}_{i-1}}(\omega) = 1$ and consequently $C((\mathcal{F} \oplus_\gamma \mathcal{G}) \cup \mathcal{G}_{i-1}) \geq \min(\mu_{\mathcal{F} \oplus_\gamma \mathcal{G}}(\omega), \mu_{\mathcal{G}_{i-1}}(\omega)) \geq \gamma - \varepsilon$. Since $\varepsilon > 0$ is arbitrary, $C((\mathcal{F} \oplus_\gamma \mathcal{G}) \cup \mathcal{G}_{i-1}) \geq \gamma$ also holds. From $R_\gamma 7$ and $R_\gamma 8$ then follows

$$(\mathcal{F} \oplus_\gamma \mathcal{G}) \cup \mathcal{G}_{i-1} = \mathcal{F} \oplus_\gamma (\mathcal{G} \cup \mathcal{G}_{i-1})$$

Furthermore, because of $C(\mathcal{G} \cup \mathcal{F}_i) \geq \gamma$ there exists an $\omega' \in \Omega$ with $\mu_{\mathcal{G}}(\omega') \geq \gamma - \varepsilon$ and $\mu_{\mathcal{F}_i}(\omega') \geq \gamma - \varepsilon$. Like above it follows $\mu_{\mathcal{G}_{i-1}}(\omega') = 1$ and $C(\mathcal{F}_i \cup \mathcal{G}_{i-1} \cup \mathcal{G}) \geq \gamma$. By definition of $\mathcal{F}_i$ it holds that $\mathcal{F}_i \cup \mathcal{G}_{i-1} = ((\mathcal{F} \oplus_\gamma \mathcal{G}_{i-1}) - \mathcal{G}_{i-1}) \cup \mathcal{G}_{i-1} = (\mathcal{F} \oplus_\gamma \mathcal{G}_{i-1}) \cup \mathcal{G}_{i-1} = \mathcal{F} \oplus_\gamma \mathcal{G}_{i-1}$, whereby the last equality follows from $C(\mathcal{G}_{i-1}) \geq \mu_{\mathcal{G}_{i-1}}(\omega') = 1$ and $R_\gamma 2a$. Thus $C((\mathcal{F} \oplus_\gamma \mathcal{G}_{i-1}) \cup \mathcal{G}) = C(\mathcal{F}_i \cup \mathcal{G}_{i-1} \cup \mathcal{G}) \geq \gamma$. From $R_\gamma 7$ and $R_\gamma 8$ then follows:

$$(\mathcal{F} \oplus_\gamma \mathcal{G}_{i-1}) \cup \mathcal{G} = \mathcal{F} \oplus_\gamma (\mathcal{G} \cup \mathcal{G}_{i-1})$$

and in combination with the corresponding equation above

$$(\mathcal{F} \oplus_\gamma \mathcal{G}_{i-1}) \cup \mathcal{G} = (\mathcal{F} \oplus_\gamma \mathcal{G}) \cup \mathcal{G}_{i-1}$$

Since $\mathcal{G}_{i-1}$ can be selected disjunctive from $\mathcal{F} \cup \mathcal{G}$ without loss of generality, this can be transformed to

$$((\mathcal{F} \oplus_\gamma \mathcal{G}_{i-1}) - \mathcal{G}_{i-1}) \cup \mathcal{G} = \mathcal{F} \oplus_\gamma \mathcal{G}$$

and by inserting the definition of $\mathcal{F}_i$ we receive the proposition. ■

Obviously, only a subset of all possible operators maximize the result set. We therefore introduce an additional postulate:

$R_\gamma 9$ (Preservation) A revision operator must compute the maximal possible result set:

$$C(\mathcal{G}) \geq \gamma \wedge \mathcal{H} \subseteq \mathcal{F} \cup \mathcal{G} \wedge \mathcal{H} \not\supseteq \mathcal{F} \oplus_\gamma \mathcal{G} \Rightarrow C(\mathcal{H}) < \gamma$$

I.e., there must be no clause from $\mathcal{F} \cup \mathcal{G}$ that can be added to the result of a revision without falling short of the requested consistency degree. It is easy to see that a revision operator satisfies $R_\gamma 9$ if and only if for every fixed formula $\mathcal{F}$ the enumeration of $2^{\mathcal{F}}$ given by Theorem 3.5 is compatible with the superset of $2^{\mathcal{F}}$, i.e., $\mathcal{F}_i \supseteq \mathcal{F}_j \Rightarrow i < j$.

3.2.1 Revision Based on Orders

According to Theorem 3.5, every revision operator is obtained by providing an ordering of the subsets of $\mathcal{F}$, for every fuzzy formula $\mathcal{F}$. In practice, it is often more natural to provide an ordering of the elements of $\mathcal{F}$, i.e., the clauses constituting $\mathcal{F}$. For example, if a γ-revision with $\mathcal{G}$ forces some clauses of $\mathcal{F}$ to be eliminated, but there is a choice as to which ones, it might be preferable to retain clauses that have more recently been added to $\mathcal{F}$, or those that have a higher degree of internal consistency. We call such preference orders among clauses *epistemic relevance orders*, in analogy to [Neb90].

More precisely, we say that an epistemic relevance order for a fuzzy formula $\mathcal{F}$ is a total ordering $\leq$ of the clauses of $\mathcal{F}$. If an epistemic relevance order $\leq$ is given for the clauses of a formula $\mathcal{F}$, an ordering of the subsets of $\mathcal{F}$ can be constructed by extending $\leq$ in the natural way:

$$\begin{aligned} \mathcal{F}_1 < \mathcal{F}_2 \quxad &\overset{\text{def}}{\iff} \quad (\exists \mathcal{K} : \mathcal{K} \in \mathcal{F}_2 \wedge \mathcal{K} \notin \mathcal{F}_1 \\ &\qquad \wedge (\forall \mathcal{K}' \geq \mathcal{K} : \\ &\qquad\quad (\mathcal{K}' \in \mathcal{F}_1 \Leftrightarrow \mathcal{K}' \in \mathcal{F}_2))), \\ \mathcal{F}_1 \leq \mathcal{F}_2 \quad &\overset{\text{def}}{\iff} \quad \mathcal{F}_1 < \mathcal{F}_2 \vee \mathcal{F}_1 = \mathcal{F}_2, \end{aligned}$$

where $\mathcal{F}_1$ and $\mathcal{F}_2$ are subsets of $\mathcal{F}$. In other words, a subset $\mathcal{F}_2$ of $\mathcal{F}$ is more relevant than a subset $\mathcal{F}_1$ if and only if when examining the clauses of $\mathcal{F}$ in order of decreasing relevance, the first clause distinguishing $\mathcal{F}_1$ and $\mathcal{F}_2$ belongs to $\mathcal{F}_2$.

This ordering of the subsets of $\mathcal{F}$, in turn, can be used to define a revision operator by enumerating the subsets of $\mathcal{F}$ in order of decreasing relevance and then applying Theorem 3.5. More precisely, we have the following:

Definition 3.6 (γ-Revision with Epistemic Relevance) *Let $\mathcal{F}, \mathcal{G} \in \mathfrak{F}(\Omega)$ be fuzzy formulas, $\leq$ an epistemic relevance order for $\mathcal{F}$ and $\gamma \in [0,1]$ arbitrary. Let $\mathcal{K}_0, \ldots, \mathcal{K}_{n-1}$ be the clauses of $\mathcal{F}$ numbered in ascending relevance order and let $\mathcal{F}_0, \mathcal{F}_1, \ldots, \mathcal{F}_{2^{|n|}-1}$ be the enumeration of the subsets of $\mathcal{F}$ obtained by assigning to each subset $\{\mathcal{K}_{i_1}, \ldots, \mathcal{K}_{i_n}\}$ of $\mathcal{F}$ the index i whose binary representation has a zero at position $i_1, \ldots, i_n$ and ones elsewhere. Set*

$$\mathcal{F} \oplus_\gamma^{\leq} \mathcal{G} := \begin{cases} \mathcal{F} & \text{if } C(\mathcal{G}) < \gamma, \\ \mathcal{F} \cup \mathcal{G} & \text{if } C(\mathcal{F} \cup \mathcal{G}) \geq \gamma, \\ \mathcal{F}_i \cup \mathcal{G} & \text{else,} \end{cases}$$

where $i = \min\{j | C(\mathcal{F}_j \cup \mathcal{G}) \geq \gamma\}$. *Then the mapping* $(\mathcal{F}, \mathcal{G}) \mapsto \mathcal{F} \oplus_\gamma^\leq \mathcal{G}$ *is called the* γ*-revision operator based on the relevance order* $\leq$.

Theorem 3.5 guarantees that the mapping $\oplus_\gamma^\leq$ defined above is indeed a revision operator. Moreover, we have:

Theorem 3.7 (Maximal result set) *Every revision operator based on epistemic relevance satisfies the maximality postulate* $R_\gamma 9$.

Proof. *It is easy to see that for every fuzzy formula* $\mathcal{F}$, *the enumeration of* $2^\mathcal{F}$ *defined above is compatible with the superset ordering, i.e. if* $\mathcal{F}_i \supseteq \mathcal{F}_j$, *then* $i \leq j$. ■

Although constructive, Definition 3.6 is not directly usable for implementing a revision operator based on epistemic relevance, because searching all subsets of $\mathcal{F}$ for the lowest-numbered one compatible with $\mathcal{G}$ would be unacceptably slow. Fortunately, $\mathcal{F} \oplus_\gamma^\leq \mathcal{G}$ can be computed in a much more direct way:

Theorem 3.8 (Efficient γ-revision operator with epistemic relevance) *Let* $\mathcal{F}, \mathcal{G}, \gamma$ *be as in Definition 3.6. Then*

$$\mathcal{F} \oplus_\gamma^\leq \mathcal{G} = \begin{cases} \mathcal{F} & \text{if } C(\mathcal{G}) < \gamma, \\ ((\mathcal{G} +_\gamma \{\mathcal{K}_{n-1}\}) & \\ \quad +_\gamma \ldots) +_\gamma \{\mathcal{K}_0\} & \text{else} \end{cases}$$

where $\mathcal{K}_0, \ldots, \mathcal{K}_{n-1}$ *are the clauses of* $\mathcal{F}$ *numbered in ascending relevance order.*

Proof. *The claim is obviously true if* $C(\mathcal{G}) < \gamma$, *so suppose that* $C(\mathcal{G}) \geq \gamma$. *Let* $2^\mathcal{F} = \{\mathcal{F}_0, \ldots, \mathcal{F}_{2^n-1}\}$ *be an enumeration of the subsets of* $\mathcal{F}$ *as in Definition 3.6, i.e., if* $\mathcal{F}_j = \{\mathcal{K}_{i_1}, \ldots, \mathcal{K}_{i_n}\}$, *then the binary representation of* j *has zeros in position* $i_1, \ldots, i_n$ *and ones elsewhere. Let* $\mathcal{F}_i$ *and* $\mathcal{F}_j$ *be the largest subsets of* $\mathcal{F}$ *satisfying* $\mathcal{F} \oplus_\gamma^\leq \mathcal{G} = \mathcal{G} \cup \mathcal{F}_i$ *and* $((\mathcal{G} +_\gamma \{\mathcal{K}_{n-1}\}) +_\gamma \ldots) +_\gamma \{\mathcal{K}_0\} = \mathcal{G} \cup \mathcal{F}_j$, *respectively. Since* $C(\mathcal{G} \cup \mathcal{F}_j) = C(\mathcal{G} +_\gamma \ldots) \geq \gamma$ *and* $i = \min\{k | C(\mathcal{G} \cup \mathcal{F}_k) \geq \gamma\}$, $i \leq j$ *must hold. Suppose now, for the purpose of reaching a contradiction, that* i *were strictly less than* j. *In this case, there would be a bit position, say* p, *where the binary expansion of* i *had a zero and the binary expansion of* j *a one, with all bits to the left of* p *being equal. In terms of clauses, this would imply* $\mathcal{K}_p \in \mathcal{F}_i, \mathcal{K}_p \notin \mathcal{F}_j$ *and* $\mathcal{K}_q \in \mathcal{F}_i \Leftrightarrow \mathcal{K}_q \in \mathcal{F}_j$ *for* $n > q > p$. *But then*

$$\begin{aligned} &(((\mathcal{G} +_\gamma \{\mathcal{K}_{n-1}\}) +_\gamma \ldots) +_\gamma \{\mathcal{K}_{p+1}\}) \cup \{\mathcal{K}_p\} \\ &= \mathcal{G} \cup \{\mathcal{K}_q | \mathcal{K}_q \in \mathcal{F}_j \wedge n > q > p\} \cup \{\mathcal{K}_p\} \\ &\subseteq \mathcal{G} \cup \mathcal{F}_i \\ &= \mathcal{G} \oplus_\gamma^\leq \mathcal{F} \end{aligned}$$

hence $C((((\mathcal{G} +_\gamma \{\mathcal{K}_{n-1}\}) +_\gamma \ldots) +_\gamma \{\mathcal{K}_{p+1}\}) \cup \{\mathcal{K}_p\}) \geq C(\mathcal{G} \oplus_\gamma^\leq \mathcal{F}) \geq \gamma$, *hence by virtue of* $E_\gamma 2a$ $\mathcal{K}_p \in (((\mathcal{G} +_\gamma \{\mathcal{K}_{n-1}\}) +_\gamma \ldots) +_\gamma \{\mathcal{K}_{p+1}\}) +_\gamma \mathcal{K}_p \subseteq \mathcal{G} \cup \mathcal{F}_j$, *and hence (because* $\mathcal{F}_j$ *was chosen to be maximal)*

$$\mathcal{K}_p \in \mathcal{F}_j,$$

thus reaching the desired contradiction. ■

Example (Fuzzy Revision) We continue the car decision example. Remember that the γ-expansion rejected the user's demand for a *spacious* car, since the result would be below the requested consistency degree. The user now has the option to *force* this request through γ-revision, resulting in:

$$\mathcal{F}_3 = \mathcal{F}_2 \oplus_{0.75} \{\mathcal{K}_3\} = \{\mathcal{K}_3, \mathcal{K}_2\},$$

i.e., the clause $\mathcal{K}_1$ representing the requirement not(*expensive*) has been removed in order to maintain a consistent state (we assume the clauses have been ordered, with $\mathcal{K}_1 < \mathcal{K}_2$). The result $\mathcal{F}_3$ has a fuzzy set interpretation of $\mu_{\mathcal{F}_3} = \{$ *Mercedes/0.5, Porsche/0.25, VW/0.75, Smart/0.0* $\}$, i.e., a *Volkswagen* would be a good fit for the revised requirements *economical* $\wedge$ *spacious*.

4 CONCLUSIONS

Fuzzy information systems have new requirements for both modeling and manipulating imperfect data, which cannot adequately be satisfied with existing approaches. In this paper, we defined a fuzzy representation model that combines both syntax and semantics, which allows for a strict separation of the structure of imperfect information, modeled as fuzzy atoms, fuzzy clauses, and fuzzy formulas, and their semantic interpretation in form of fuzzy sets. Modifications on these fuzzy formulas can be performed by the operators γ-expansion and γ-revision. Both always guarantee a specified consistency degree — an important feature for database and information system practitioners.

Since these operations abstract from the manipulation of single fuzzy sets, more complex application-specific algorithms dealing with imperfection can be developed in a more natural way.

The model presented here has been successfully incorporated into an object-oriented data model, which in turn has been implemented with a number of class libraries for the *Java* language. An accompanying fuzzy information system architecture provides the necessary extensions to a classical architecture for creating fuzzy information systems. With this approach we have been able to selectively enhance information systems with the capabilities to represent complex imperfect information and process them in a robust and consistency-preserving way. So far, we successfully built a fuzzy decision support system and a fuzzy noun phrase coreference system. Both proved to be easier

to design and implement than comparative classical, non-fuzzy systems while at the same time providing new features like adjustable certainty degrees.

Acknowledgements. I would like to thank Gerd Hillebrand for many helpful comments on several versions of this paper, as well as numerous suggestions on the model presented herein.

References

[AGM85] C.E. Alchourrón, P. Gärdenfors, and D. Makinson. On the logic of theory change: Partial meet contraction and revision functions. *Journal of Symbolic Logic*, 2(50):510–530, June 1985.

[BK95] P. Bosc and J. Kacprzyk, editors. *Fuzziness in Database Management Systems*, volume 5 of *Studies in Fuzziness*. Physica-Verlag, 1995.

[Bos96] Birgit Boss. *Fuzzy-Techniken in objektorientierten Datenbanksystemen zur Unterstützung von Entwurfsprozessen*. Number 4 in Dissertationen zu Datenbanken und Informationssystemen. Infix, 1996.

[Cal97] Rita De Caluwe, editor. *Fuzzy and Uncertain Object-Oriented Databases*, volume 13 of *Advances in Fuzzy Systems — Applications and Theory*. World Scientific, 1997.

[DP94] Didier Dubois and Henri Prade. A survey of belief revision and updating rules in various uncertainty models. *International Journal of Intelligent Systems*, 9(1):61–100, January 1994.

[Gär88] Peter Gärdenfors. *Knowledge in Flux*. MIT Press, 1988.

[Gär92] Peter Gärdenfors. Belief revision: An introduction. In P. Gärdenfors, editor, *Belief Revision*, pages 1–28. Cambridge University Press, Cambridge, UK, 1992.

[KF88] George J. Klir and Tina A. Folger. *Fuzzy Sets, Uncertainty, and Information*. Prentice-Hall, 1988.

[KM91] Hirofumi Katsuno and Alberto O. Mendelzon. On the difference between updating a knowledge base and revising it. In *KR*, pages 387–394. Cambridge, Massachusetts, April 22–25 1991.

[LXHY99] Jonathan Lee, Nien-Lin Xue, Kuo-Hsun Hsu, and Stephen J. Yang. Modeling imprecise requirements with fuzzy objects. *Information Sciences*, (118):101–119, 1999.

[MS97] Amihai Motro and Philippe Smets, editors. *Uncertainty Management In Information Systems*. Kluwer Academic Publishers, 1997.

[Neb90] Bernhard Nebel. *Reasoning and Revision in Hybrid Representation Systems*. Lecture Notes in Artificial Intelligence. Springer-Verlag, 1990.

[Neb91] Bernhard Nebel. Belief revision and default resoning: Syntax-based approaches. In J.A. Allen, R. Fikes, and E. Sandewall, editors, *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR '91)*, pages 417–428. Morgan Kaufmann, April 1991.

[Neb92] Bernhard Nebel. Syntax-based approaches to belief revision. In P. Gärdenfors, editor, *Belief Revision*, pages 52–88. Cambridge University Press, Cambridge, UK, 1992.

[Pet96] Frederick E. Petry. *Fuzzy Databases Principles and Applications*. International Series in Intelligent Technologies. Kluwer Academic Publishers, 1996.

[YG99] Adnan Yazici and Roy George. *Fuzzy Database Modeling*, volume 26 of *Studies in Fuzziness and Soft Computing*. Physica-Verlag, 1999.

[YOTN87] R.R. Yager, S. Ovchinnikov, R.M. Tong, and H.T. Nguyen, editors. *Fuzzy Sets and Applications: Selected Papers by L.A. Zadeh*. Wiley&Sons, 1987.

[Zad65] L.A. Zadeh. Fuzzy sets. In Yager et al. [YOTN87], pages 29–44. Originally published in *Information and Control*, Vol. 8, New York: Academic Press, 1965, Seiten 338–353.

Nonmonotonic Inference Operators for Fuzzy Logic

Eva Richter
University of Leipzig
Institute of Computer Science,

Abstract

Fuzzy logic in an abstract sense allows to draw (graded) conclusions from only *partially* "given" premises i.e. fuzzy sets of formulas. This means every formula (of the premises as well as of the conclusion) is only given to some degree, which can be seen as the degree of truth or degree of confidence. On the other hand, one aspect of nonmonotonic reasoning is to pull up some further information from the context to complete the given one. Because in real life information is always both, fuzzy and incomplete, it seems reasonable to have inference operators that can combine techniques for completing information and dealing with uncertainty. The aim of this paper is to suggest a technique to construct such combined operators. The paper is organized as follows: We start with the notion of an abstract fuzzy deduction system, then we show how we can mimic the technique of defining nonmonotonic inference operators by the use of model selection operators in an abstract fuzzy deduction system. We present some propositions characterizing its properties. Then we give a very short overview of some special aspects of the logic **FLn** that is based on Łukasiewicz logic. For fuzzy inference frames based on it we can prove some generalizations of results from two valued NMR. Finally we define two operators associated to a fuzzy Poole system to exemplify the technique for creating nonmonotonic inference operators.

1 Abstract Fuzzy Logic

In classical logic the notion of a closure operator is a powerful tool to describe deduction operators. Indeed, embracing Tarski's viewpoint a monotone logic is a set (of formulas) together with a closure operator (on the power set of the formulas.) Generalizing this (see [Ger01]) we come to the notion of an *abstract deduction system*: we only have to replace the power set of formulas by a complete lattice. A pair (L, D) is said to be an *abstract deduction system* if $[L, \geq_L]$ is a complete lattice (whose elements are pieces of information) and $D : L \longrightarrow L$ is a closure operator (i.e. it is reflexive, monotone and idempotent). The intended interpretation of $v \geq_L w$ is: v carries more (more exact) information than w. An *abstract semantics* in L is a subset $\mathcal{M} \subseteq L$, and its elements are called models. If $v \in L$, $m \in \mathcal{M}$ and $m \geq_L v$, then we say m is a *model* of v and we write $m \models v$. If v and w admit the same models then we say that v is *logically equivalent* to w.

Every semantics $\mathcal{M}$ induces a closure operator $\boldsymbol{J}_{\mathcal{M}} : L \longrightarrow L$ which is defined by setting $\boldsymbol{J}_{\mathcal{M}}(v) := \bigcap\{m \in \mathcal{M} : m \geq_L v\}$ for a given piece of information v. We can also define $\boldsymbol{J}_{\mathcal{M}}$ as follows: Consider the operators

$$\mathbf{mod}^{\models} : L \longrightarrow \mathcal{P}(\mathcal{M}) \quad \text{and}$$

$$\mathbf{th}^{\models} : \mathcal{P}(\mathcal{M}) \longrightarrow L$$

defined by setting for every $v \in L$ and $N \in \mathcal{P}(\mathcal{M})$,

$$\mathbf{mod}^{\models}(v) = \{m \in \mathcal{M} \mid m \models v\} \text{ and}$$

$$\mathbf{th}^{\models}(N) = \sup\{v \in L \mid \forall m \in N : m \models v\}.$$

Then, as usual, $\mathbf{mod}^{\models}(v)$ is the set of all models of v and $\mathbf{th}^{\models}(N)$ the information shared by all the models in N. It is easy to verify that $\mathbf{mod}^{\models}$ and $\mathbf{th}^{\models}$ define

a Galois connection such that $\mathbf{th}^{\models} \circ \mathbf{mod}^{\models}$ coincides with $\boldsymbol{J}_{\mathcal{M}}$.

An *abstract logic* is an abstract deduction system together with an abstract semantics $\mathcal{M}$, s.t. $D = \boldsymbol{J}_{\mathcal{M}}$, i.e. the completeness theorem holds. A *theory* is a fixpoint τ of D, i.e. some information closed under D.

This approach now will be used for the case that the given information is a fuzzy set of premises (cf. [Ger01]): Given a language J, the set F_J of formulas in J and a complete lattice $(W, \geq_W)$, a *fuzzy subset* or *W-set of formulas (in J)* is a map $v : F_J \longrightarrow W$. The *support* of a fuzzy set v is the set $\mathrm{supp}(v) = \{\varphi \in F_J \mid v(\varphi) \gneq 0\}$. We denote by $\mathcal{F}(F_J)$ the class of fuzzy subsets of F_J. The class $\mathcal{F}(F_J)$ is a complete lattice. The ordering $\geq_{\mathcal{F}(F_J)}$ here is the partial order $\supseteq$ of fuzzy sets defined by: $v \supseteq w$ iff $v(\phi) \geq_W w(\phi)$ for all formulas $\phi \in F_J$. The information carried by $v \in \mathcal{F}(F_J)$ is that for every formula ϕ the actual value of ϕ is at least $v(\phi)$. An *abstract fuzzy deduction system* is an abstract deduction system, where for the given language J and its formulas F_J the complete lattice $L = \mathcal{F}(F_J)$ and $D(v)$ is the fuzzy set of consequences of v. Having in mind that an abstract semantics $\mathcal{M}$ for this $L = \mathcal{F}(F_J)$ is a subset of $\mathcal{F}(F_J)$, we name $\mathcal{M}$ in this case an *abstract fuzzy semantics*. The closure operator $\boldsymbol{J}_{\mathcal{M}}$, induced by an abstract fuzzy semantics $\mathcal{M}$ can be represented pointwise, i.e. by the following equation: $\boldsymbol{J}_{\mathcal{M}}(v)(\phi) = \inf\{m(\phi) \mid m \in \mathcal{M}, m \supseteq v\}$. A fuzzy operator $T : \mathcal{F}(F_J) \longrightarrow \mathcal{F}(F_J)$ is *compact* if for every fuzzy set v

$$T(v) = \bigcup\{T(v_{\mathrm{fin}}) \mid \quad v_{\mathrm{fin}} \subseteq v \text{ and } \mathrm{supp}(v_{\mathrm{fin}}) \text{ is finite}\}.$$

Note that in fuzzy logic the notions of compactness and continuity don't coincide.

2 Nonmonotonic Fuzzy Inference

In this section, we suggest a technique to establish nonmonotonic fuzzy inference operators by generalizing a construction principle known from two-valued logic. Starting with the classical operator $\mathbf{Mod}^{\models}$: $\mathcal{P}(F) \longrightarrow \mathcal{P}(Int)$ that assigns the class $\mathbf{Mod}^{\models}(V)$ of interpretations satisfying the formulas in V to every set of formulas $V \subseteq F$ one can define (see [DH97]) nonmonotonic inference operators using additional model operators $\Psi : \mathcal{P}(F) \longrightarrow \mathcal{P}(Int)$, that satisfy the condition $\Psi(V) \subseteq \mathbf{Mod}^{\models}(V)$. Then some metalogical properties can be described by the properties of the model selectors.

We mimic this construction with the following definition.

Definition 1 Let $(\mathcal{F}(F_J), \boldsymbol{J}_{\mathcal{M}}, \mathcal{M})$ be a fuzzy logic, then a *(fuzzy) model operator* Φ is an operator $\Phi : \mathcal{F}(F_J) \longrightarrow \mathcal{P}(\mathcal{M})$. The *fuzzy inference operator* $\mathbf{C}_\Phi : \mathcal{F}(F_J) \longrightarrow \mathcal{F}(F_J)$ that is *associated to a model operator* Φ is defined by $\mathbf{C}_\Phi(v) := \mathbf{th}^{\models}(\Phi(v))$.

From now on we have to deal with two kinds of fuzzy operators: a monotonic one $\mathbf{C}_{\mathrm{L}}$ (for instance $\boldsymbol{J}_{\mathcal{M}}$ or D from above) and a not necessarily monotonic one $\mathbf{C}$ (sometimes with some indices). Because some of the metalogical properties of the nonmonotonic operators we are interested in depend on a monotonic fuzzy operator $\mathbf{C}_{\mathrm{L}}$ we will define them in association with inference frames:

Definition 2 Let $(\mathcal{F}(F), \mathbf{C}_{\mathrm{L}})$ be an abstract deduction system where $\mathbf{C}_{\mathrm{L}}$ is compact. A system $(\mathcal{F}(F), \mathbf{C}_{\mathrm{L}}, \mathbf{C})$ is a *fuzzy inference frame* iff it satisfies the condition of *supraclassicality*: $\mathbf{C}(v) \supseteq \mathbf{C}_{\mathrm{L}}(v)$.

1. $(\mathcal{F}(F), \mathbf{C}_{\mathrm{L}}, \mathbf{C})$ satisfies *left absorption* iff $\mathbf{C}_{\mathrm{L}}(\mathbf{C}(v)) = \mathbf{C}(v)$.
2. $(\mathcal{F}(F), \mathbf{C}_{\mathrm{L}}, \mathbf{C})$ satisfies *right absorption* iff $\mathbf{C}(\mathbf{C}_{\mathrm{L}}(v)) = \mathbf{C}(v)$.
3. $(\mathcal{F}(F), \mathbf{C}_{\mathrm{L}}, \mathbf{C})$ is a *fuzzy deduction frame* iff left and right absorption are satisfied.
4. A fuzzy inference system $(\mathcal{F}(F), \mathbf{C}_{\mathrm{L}}, \mathbf{C})$ is called *cumulative* if always $v \subseteq w \subseteq \mathbf{C}(v)$ implies $\mathbf{C}(v) = \mathbf{C}(w)$.

Proposition 1 *Let $\Phi : \mathcal{F}(F) \longrightarrow \mathcal{P}(\mathcal{M})$, s.t. $\Phi(v) \subseteq \mathbf{mod}^{\models}(v)$, be a fuzzy model operator. Then the fuzzy inference frame $(\mathcal{F}(F), \boldsymbol{J}_{\mathcal{M}}, \mathbf{C}_\Phi)$ satisfies left absorption.*

PROOF: $\boldsymbol{J}_{\mathcal{M}}(\mathbf{C}_\Phi(v)) = \mathbf{th}^{\models} \circ \mathbf{mod}^{\models}(\mathbf{C}_\Phi)(v) =$
$= \mathbf{th}^{\models} \circ \mathbf{mod}^{\models}(\mathbf{th}^{\models}(\Phi)(v)) = \mathbf{th}^{\models}(\Phi)(v) = \mathbf{C}_\Phi(v)$. □

Definition 3 1. The set $cl(\mathcal{M})$ of *elementary classes* of a semantics $\mathcal{M}$ is defined by $cl(\mathcal{M}) = \{K \subseteq \mathcal{M} \mid K = \mathbf{mod}^{\models}(w) \text{ for some } w \in \mathcal{F}(F_J)\}$.
2. A *selection function* is a function $\Sigma : cl(\mathcal{M}) \longrightarrow \mathcal{P}(\mathcal{M})$, s.t. $\Sigma(K) \subseteq K$.
3. A selection function Σ is called *closed*, if always $\Sigma(K) \in cl(\mathcal{M})$.
4. A model operator Φ is a *model selector*, if there is a selection function $\Sigma : cl(\mathcal{M}) \longrightarrow \mathcal{P}(\mathcal{M})$ with $\Phi(v) = \Sigma(\mathbf{mod}^{\models}(v))$. Then we write Φ_Σ for Φ.

Proposition 2 *1. Let Φ_Σ be a model selector, then Φ_Σ is invariant, i.e. $\Phi_\Sigma(v) = \Phi_\Sigma(\boldsymbol{J}_{\mathcal{M}}(v))$ for all $v \in F_J$.*

2. *Let Φ_Σ be a model selector, then the fuzzy inference frame $(\mathcal{F}(F), \boldsymbol{J}_\mathcal{M}, \mathbf{C}_{\Phi_\Sigma})$ is a fuzzy deduction frame.*
3. *For every selection function Σ there is a closed selection function Δ s.t. $\mathbf{C}_{\Phi_\Sigma}(v) = \mathbf{C}_{\Phi_\Delta}(v)$ for all $v \in \mathcal{F}(F_J)$.*

PROOF:
1. $\Phi_\Sigma(v) = \Sigma(\mathbf{mod}^\models(v)) = \Sigma(\mathbf{mod}^\models(\boldsymbol{J}_\mathcal{M}(v))) = \Phi_\Sigma(\boldsymbol{J}_\mathcal{M}(v))$.
2. By Proposition 1 we already have left absorption. Let $\Phi(v) = \Sigma(\mathbf{mod}^\models(v))$ be given by a selection function Σ and we set $w := \boldsymbol{J}_\mathcal{M}(v)$. Therefore $\mathbf{C}_{\Phi_\Sigma}(v) = \mathbf{th}^\models(\Sigma(\mathbf{mod}^\models(v))) = \mathbf{th}^\models(\Sigma(\mathbf{mod}^\models(w))) = \mathbf{C}_{\Phi_\Sigma}(w)$.
3. is obvious. □

Definition 4 Let Σ be a closed selection function. The binary *relation* $\leq_\Sigma$, $\leq_\Sigma \subseteq cl(\mathcal{M}) \times cl(\mathcal{M})$ is defined as follows: $K \leq_\Sigma K_1$, iff $K = K_1$ or there is a $K_2 \supseteq K_1$, $K_2 \in cl(\mathcal{M})$ with $\Sigma(K_2) = K$.

Proposition 3 *Let $\Phi_\Sigma : \mathcal{F}(F) \longrightarrow \mathcal{P}(\mathcal{M})$ be a model selector and let the relation $\leq_\Sigma$ be antisymmetric. Then $(\mathcal{F}(F), \boldsymbol{J}_\mathcal{M}, \mathbf{C}_\Phi)$ is a cumulative fuzzy deduction frame.*
Let $(\mathcal{F}(F), \mathbf{C}_\mathrm{L}, \mathbf{C})$ be a cumulative fuzzy deduction frame, then there exists a closed selection function Σ, where the corresponding relation $\leq_\Sigma$ is antisymmetric, s.t $\mathbf{C} = \mathbf{C}_{\Phi_\Sigma}$.

PROOF: We have to prove that $v \subseteq w \subseteq \mathbf{C}_{\Phi_\Sigma}(v)$ implies $\mathbf{C}_{\Phi_\Sigma}(w) = \mathbf{C}_{\Phi_\Sigma}(v)$. Let $v \subseteq w$, then $\mathbf{mod}^\models(v) \supseteq \Sigma(\mathbf{mod}^\models(w))$ holds. By the definition of $\leq_\Sigma$ we get $\Sigma(\mathbf{mod}^\models(v)) \leq_\Sigma \Sigma(\mathbf{mod}^\models(w))$. Because of $v \subseteq w \subseteq \mathbf{C}_{\Phi_\Sigma}(v) = \mathbf{th}^\models(\Sigma(\mathbf{mod}^\models(v)))$ we have $\mathbf{mod}^\models(w) \supseteq \mathbf{mod}^\models(\mathbf{th}^\models(\Sigma(\mathbf{mod}^\models(v))))$ and thus $\Sigma(\mathbf{mod}^\models(v)) \subseteq \mathbf{mod}^\models(w)$ and thus $\Sigma(\mathbf{mod}^\models(w)) \leq_\Sigma \Sigma(\mathbf{mod}^\models(v))$. The relation $\leq_\Sigma$ was supposed to be antisymmetric and therefore we have $\Sigma(\mathbf{mod}^\models(w)) = \Sigma(\mathbf{mod}^\models(v))$. This implies $\mathbf{C}_{\Phi_\Sigma}(v) = \mathbf{C}_{\Phi_\Sigma}(w)$. □

Proposition 4 *Let $(\mathcal{F}(F), \boldsymbol{J}_\mathcal{M}, \mathcal{M})$ be an abstract logic. Let $N \in cl(\mathcal{M})$ be a model class and let Σ be a selection function defined in the following way*

$$\Sigma(K) = \begin{cases} K \cap N & \text{if the intersection is nonempty} \\ K & \text{otherwise} \end{cases}$$

then $(\mathcal{F}(F), \boldsymbol{J}_\mathcal{M}, \mathbf{C}_{\Phi_\Sigma})$ is a cumulative fuzzy deduction frame.

PROOF: The selection function Σ is closed because $cl(\mathcal{M})$ is closed under meet. We show that the relation $\leq_\Sigma$ is antisymmetric: Let $K_1, K_2 \in cl(\mathcal{M})$ be such that $K_1 \leq_\Sigma K_2$ and $K_2 \leq_\Sigma K_1$. For $K_1 = K_2$ there is nothing to prove. Otherwise by definition of $\leq_\Sigma$ there exist $K_3, K_4 \in cl(\mathcal{M})$ s.t. $K_3 \supseteq K_2$ with $\Sigma(K_3) = K_1$ and $K_4 \supseteq K_1$ with $\Sigma(K_4) = K_2$. The rest of the proof is simply to check all cases for $\Sigma(K_3)$ and $\Sigma(K_4)$. □

3 Łukasiewicz Fuzzy Logic FLn

3.1 Semantics

In the following we assume the lattice W from section 1 to be the real unit interval: $W = [0,1]$. The language J is assumed to include for each rational number $r \in [0,1]$ a truth constant $\mathbf{r}$. Formulas are either truth constants or built from propositional variables and truth constants using the connectives $\wedge, \vee, \otimes, \rightarrow, \neg$.

Definition 5 Let $\overline{m}$ be a valuation of all the variables of the language J then the fuzzy set $m \in \mathcal{F}(F_J)$ is a *truth-functional valuation of the formulas* in J iff $m(\varphi) = f_\varphi(\overline{m})$ holds for every formula ϕ where f_φ is the function obtained by interpretation of φ.

If a fuzzy semantics is the set of all truth-functional valuations of a many-valued logic where the lattice of truth values is the real unit interval and the language J contains only n variables with $n \in \mathbb{N}$ then there is a bijection $\bar{}\ : \mathcal{P}(\mathcal{M}) \longrightarrow [0,1]^n$ that maps every model class N to the subset $\overline{N}$ of the valuations of the variables in J, s.t.

$$\overline{N} = \{\overline{m} \in [0,1]^n \mid m(\varphi) = f_\varphi(\overline{m}) \, \forall \varphi \in F_J, m \in N\}.$$

From now on we deal only with the special logic, named $\mathbf{FLn} = (\mathcal{F}(F_J), \boldsymbol{J}_{\mathcal{M}_L}, \mathcal{M}_L)$, where $\mathcal{M}_L$ is the class of truth-functional valuations of the formulas in the Łukasiewicz logic. A deep investigation of this logic can be found e.g. in [NPM99] or in [Háj98, Ch.3].

In the Łukasiewicz logic the logical constants $\mathbf{r}$ are interpreted by the real r and logical connectives are interpreted as follows:

$$\begin{aligned} \neg a &= 1 - a \\ a \wedge b &= \min\{a, b\}, \\ a \vee b &= \max\{a, b\}, \\ a \otimes b &= \max\{0, a + b - 1\}, \\ a \rightarrow b &= \min\{1, 1 - a + b\} \end{aligned}$$

where $a, b \in W = [0,1]$. As usual we obtain the functions f_φ by interpretation of the connectives occuring in φ.

The most important results for our purposes is the theorem below (c.f. [Ger01, Theorem 7.3. in Ch. 8] and its corollaries(see [Ger01, Theorem 8.1. in Ch. 2 and Theorem 2.6 in Ch.4]).

Proposition 5 *$\mathcal{M}_L$ is continuous and closed under ultraproducts.*

Corollary 1 *For every formula $\phi \in F$ holds that $J_{\mathcal{M}_L}(v)(\phi) = a$ implies that there is a model $m \in \mathbf{mod}^{\models}(v)$ of v with $m(\phi) = a$.*

Corollary 2 *There is a fuzzy Hilbert system $\mathcal{S}_L = (\mathrm{LAx}, R)$ where LAx is a fuzzy set of axioms and R is a set of fuzzy inference rules such that the continuous deduction operator $\boldsymbol{D}_{\mathcal{S}}$ coincides with $\boldsymbol{J}_{\mathcal{M}_L}$. (The exact definition of $\boldsymbol{D}_{\mathcal{S}}$ will be given in section 3.2.) That means we have completeness and therefore $\mathbf{FLn} = (\mathcal{F}(F_J), \boldsymbol{D}_{\mathcal{S}}, \mathcal{M}_L)$ is an abstract fuzzy logic.*

Due to the fact, that all connectives are interpreted by continuous and linear functions the truth valuation functions of the formulas, sometimes called McNaughton-functions, have a special form. This special form is described by the *McNaughton theorem* (see e.g. [Got00]):

Proposition 6 (McNaughton Theorem) *Let $\phi \in F_J$ be a formula with n variables $x_1, \ldots, x_n$ then*

(i) its truth valuation function f_ϕ is continuous and

(ii) there exists a rational partition $\overline{D} = \{\overline{D_1}, \ldots, \overline{D_k}\}$ of the n-cube $[0,1]^n$, s.t. the restriction of f_ϕ to $\overline{D_i}$ is a linear polynomial $g_i(x_1, \ldots, x_n) = b_i + \sum a_{i,j} x_j$, where all $a_{i,j}, b_i$ are integers.

The converse of this theorem holds and gives a complete characterization of the truth valuation functions. A constructive proof of it and the corollaries listed below can be found in [Mun94].

This gives us a facility to compute model classes for fuzzy sets with finite support in the following way (see e.g. [Ger01, p. 167ff.]):

Corollary 3 *Let $v \in \mathcal{F}(F)$ be a fuzzy set with rational values whose support is the finite set $\{\phi_1, \ldots, \phi_s\}$ of formulas. Then we can find a rational partition $\overline{D} = \{\overline{D_1}, \ldots, \overline{D_t}\}$ and linear polynomials $g_1^1, \ldots, g_s^1, \ldots, g_1^t \ldots, g_s^t$ in such a way that :*

$$\begin{aligned} \mathbf{mod}^{\models}(v) &= \{m \in \mathcal{M}_L \mid \overline{m} \in \overline{N_1} \cup \ldots \cup \overline{N_t}\} \\ &= N_1 \cup \ldots \cup N_t, \end{aligned}$$

where $\overline{N_i}$ is the simplex defined by the linear system

$$g_j^i(x_1, \ldots, x_k) \geq v(\phi_j), \text{ for all } j = 1, \ldots, s \quad \text{where } (x_1, \ldots, x_k) \in \overline{D_i}.$$

The following Corollary (see [Ger01]) allows us to compute for a given formula ϕ its degree of being a consequence of the initial valuation v i.e. the value $J_{\mathcal{M}_L}(v)(\phi)$.

Corollary 4 *Given a fuzzy set v with rational values and a finite support for every formula φ we can find (in an effective way) a class $\overline{N} = \{\overline{N_1}, \ldots, \overline{N_l}\}$ of simplices and linear polynomials $h_1, \ldots, h_l$ such that we can compute the minimal value to which a formula φ is included in the consequences of a fuzzy set v by*

$$\begin{aligned} J_{\mathcal{M}_L}(v)(\varphi) = \; & \min_{\overline{N_1}}\{h_1(x_1, \ldots, x_n)\} \wedge \ldots \\ & \ldots \wedge \min_{\overline{N_l}}\{h_l(x_1, \ldots, x_n)\}. \end{aligned}$$

3.2 Syntax

As stated in the Corollary 2, there is a fuzzy Hilbert system $\mathcal{S}_L = (\mathrm{LAx}, R)$ that axiomatizes the semantics $\mathcal{M}_L$. The fuzzy set of logical axioms $\mathrm{LAx} \subseteq \mathcal{F}(F)$ and the set of fuzzy inference rules R are listed in detail in [NPM99, p.115 f.]. Every k-ary inference rule consists of a syntactical part r' and a valuation part r'', that means $r = (r', r'')$ where r' is a partial function $r' : (F_J)^k \longrightarrow F_J$ and r'' maps $[0,1]$ to $[0,1]$. A *formal evaluated proof* of a formula ψ from a fuzzy set v is a finite sequence of evaluated formulas: $\pi_\psi := (\phi_0/a_0), \ldots, (\phi_n/a_n)$, s.t. $\phi_n := \psi$ and for every $i \leq n$ either

1. ϕ_i will be regarded as a logical axiom: $a_i = \mathrm{LAx}(\phi_i)$, or
2. ϕ_i will be regarded as a proper axiom: $a_i := v(\phi_i)$, or
3. there exist a k-ary inference rule $r \in R$ and $i_1, \ldots, i_k < i$ s.t. $(x_i/a_i) = (r'(\phi_{i_1}, \ldots, \phi_{i_k})/r''(a_{i_1}, \ldots, a_{i_k}))$.

The *value of the proof* is: $\mathrm{Val}_v(\pi_\psi) := a_n$. Depending on the proof the values of the proofs for the same formula can be different. The *deduction operator* $\boldsymbol{D}_{\mathcal{S}} : \mathcal{F}(F_J) \longrightarrow \mathcal{F}(F_J)$ that is *associated to* $\mathcal{S}$ is defined by setting:

$$\boldsymbol{D}_{\mathcal{S}}(v)(\phi) = \sup\{\mathrm{Val}_v(\pi_\phi) \mid \pi_\phi \text{ is a proof of } \phi\}.$$

We denote $\boldsymbol{D}_{\mathcal{S}}(v)(\phi) = a$ by $v \vdash_a \phi$.

The deduction theorem holds only in the following restricted form (see [NPM99, Theorem 4.12]).

Proposition 7 (Deduction Theorem) *Let v be a fuzzy set of formulas, ϕ a closed formula. Then for every formula $\psi \in F_J$ holds: there exists some $k \in \mathbb{N}$, s.t.*

$$v \vdash_a \phi^k \to \psi \text{ iff } v \cup \phi^* \vdash_a \psi,$$

where ϕ^* denotes the fuzzy set whose support is the single formula $\{\phi\}$ s.t. $\phi^*(\phi) = 1$.

3.3 Consistency and Completeness of Fuzzy Sets

Since in classical logic a contradictory theory turns into a useless theory where everything is provable it is one of the crucial questions also for fuzzy logic to decide whether a theory is consistent or not.

Definition 6 In the logic **FLn** a fuzzy set v is *contradictory* iff $J_{\mathcal{M}_L}(v) = F(F_J)$. It is called *consistent* in the opposite case.

We have the following characterisation for contradictory theories (cf. [GN97]):

Proposition 8 *A fuzzy set v is contradictory iff any of the following holds:*

(i) there is a formula ϕ and proofs π_ϕ, $\pi_{\neg\phi}$ s.t., $\mathrm{Val}_v(\pi_\phi) \otimes \mathrm{Val}_v(\pi_{\neg\phi}) > 0$.

(ii) $Incons(v) := \sup\{a \mid v \vdash_a (\phi \wedge \neg\phi), \phi \in F\} > \frac{1}{2}$.

Recall that in classical logic a theory is complete if every formula or its negation is provable in it. A similar concept can be introduced in fuzzy logic too ([NPM99]), such theories where the membership degrees of all formulas, have the structure of truthvalues are the complete ones.

Definition 7 *A fuzzy theory τ is called complete iff it is consistent and for every closed formula ϕ and every $a \in W$ holds: $\tau \vdash_a \phi$ implies $\tau \vdash \phi \to \boldsymbol{a}$.*

It is easy to see that the complete theories are the elements of $\mathcal{M}_L$.

4 Nonmonotonic Fuzzy Inference II

In this section we restrict ourselves to inference frames based on **FLn**, where $\mathbf{C}_\mathrm{L} = \boldsymbol{D}_\mathcal{S} = \boldsymbol{J}_{\mathcal{M}_L}$. The following observations are extensions of the respective propositions in [Mak94]. Because of the restricted form of the deduction theorem for **FLn** a few of them can only be proved in a restricted form.

Lemma 1 *Let $(\mathcal{F}(F), \mathbf{C}_\mathrm{L}, \mathbf{C})$ be a fuzzy inference frame.*

1. *Let $\mathbf{C}$ be left absorbing, then:*
   (a) $\mathbf{C}(v)(\phi) = a$ and $\mathbf{C}(v)(\psi) = b$ imply $\mathbf{C}(v)(\phi \wedge \psi) = a \wedge b$.
   (b) If $\mathbf{C}(v)(\phi) = a$ and $\mathbf{C}_\mathrm{L}(\phi^*)(\psi) = b$ then there exists a $k \in \mathbb{N}$, s.t. $\mathbf{C}(v)(\psi) \geq a^k \otimes b$, where k depends only on ϕ and ψ.
2. *Let $\mathbf{C}$ be right absorbing, then:*
   (a) $\mathbf{C}_\mathrm{L}(v) = \mathbf{C}_\mathrm{L}(w)$ implies $\mathbf{C}(v) = \mathbf{C}(w)$.
   (b) If $v(\phi) \leq w(\phi) \leq \mathbf{C}_\mathrm{L}(v)(\phi)$ for every $\phi \in F$, then $\mathbf{C}(v) = \mathbf{C}(w)$.

PROOF: 1a: Left absorption gives $\mathbf{C}_\mathrm{L}(\mathbf{C}(v))(\phi) = \mathbf{C}(v)(\phi)$ and $\mathbf{C}_\mathrm{L}(\mathbf{C}(v))(\psi) = \mathbf{C}(v)(\phi)$, by [NPM99, Theorem 4.10.].
$\mathbf{C}_\mathrm{L}(\mathbf{C}(v))(\phi \wedge \psi) = a \wedge b$ holds and by left absorption also $\mathbf{C}(v)(\phi \wedge \psi) = a \wedge b$.
1b: From $\mathbf{C}_\mathrm{L}(\phi^*)(\psi) = b$ follows by deduction theorem [NPM99, Theorem 4.12] that there exists some $k \in \mathbb{N}$, such that $\mathbf{C}_\mathrm{L}(\emptyset)(\phi^k \to \psi) = b$. Now we will construct a proof for ψ from v:
$\pi_\psi := ((a \longrightarrow \phi)/1)\{$introduction rule of constants$\}$, (ϕ^k/a^k) $\{$by using $\vdash (\phi \otimes \psi)^n \to (\phi^n \to \psi^n)$ and MP$\}$, $((\phi^k \to \psi)/b)\{$deduction theorem$\}$, $(\psi/a^k \otimes b)\{$MP$\}$. From this follows the assertion by $\mathbf{C}_\mathrm{L}(v)(\psi) = \sup\{\mathrm{Val}_v(\pi_\psi) \mid \pi_\psi$ proof for ψ $\}$.
2a: $\mathbf{C}(v) = \mathbf{C}(\mathbf{C}_\mathrm{L}(v)) = \mathbf{C}(\mathbf{C}_\mathrm{L}(w)) = \mathbf{C}(w)$.
2b: Due to the monotony of $\mathbf{C}_\mathrm{L}$: the outcome of the application of $\mathbf{C}_\mathrm{L}$ to the inequality is $\mathbf{C}_\mathrm{L}(v)(\phi) \leq \mathbf{C}_\mathrm{L}(w)(\phi)$. This, together with 2a gives the assertion. □

Definition 8 $\mathbf{C}$ is said to be *distributive*, if $\mathbf{C}(v) \cap \mathbf{C}(w) \subseteq \mathbf{C}(\mathbf{C}_\mathrm{L}(v) \cap \mathbf{C}_\mathrm{L}(w))$ holds for every $v, w \in \mathcal{F}(F)$.

Lemma 2 *Let $(\mathcal{F}(F), \mathbf{C}_\mathrm{L}, \mathbf{C})$ be a fuzzy deduction frame, and let $\mathbf{C}$ be distibutive, then $\mathbf{C}(v \cup \phi^*) \cap \mathbf{C}(v \cup \psi^*)(z) \leq \mathbf{C}(v \cup (\phi \vee \psi)^*)(z)$ holds.*

PROOF: Analogously to the classical case the following equation holds :

$$\mathbf{C}_\mathrm{L}((\phi \vee \psi)^*) = \mathbf{C}_\mathrm{L}(\phi^*) \cap \mathbf{C}_\mathrm{L}(\psi^*).$$

$$\begin{aligned}
&\mathbf{C}(v \cup \phi^*) \cap \mathbf{C}(v \cup \psi^*)(z) = \\
&\quad = \min\{\mathbf{C}(v \cup \phi^*)(z), \mathbf{C}(v \cup \psi^*)(z)\} \\
&\quad \leq \mathbf{C}(v \cup (\mathbf{C}_\mathrm{L}(\phi^*) \cap \mathbf{C}_\mathrm{L}(\psi^*)))(z) \\
&\quad = \mathbf{C}(v \cup \mathbf{C}_\mathrm{L}((\phi \vee \psi)^*))(z) \\
&\quad = \mathbf{C}\mathbf{C}_\mathrm{L}(v \cup \mathbf{C}_\mathrm{L}(\phi \vee \psi)^*)(z) \\
&\quad = \mathbf{C}(v \cup (\phi \vee \psi)^*)(z).
\end{aligned}$$

□

5 Poole Systems

5.1 Definitions

Recall that in classical logic a Poole system is defined in the following way (see [Poo89]):

Definition 9 1. A *Poole system* is a pair (D, K) where D and K are both sets of closed formulas in the language J of ordinary first order or of propositional logic. D is called the set of *default propositions* of the system and K is called set of *constraints* of the system. When $K = \emptyset$ then (D, K) is said to be a Poole system *without constraints.*

2. Given any Poole system (D, K), we define *its associated extension family function* for a set U of propositions by:

$$E(U) = \{\mathbf{C}_{\mathrm{L}}(U \cup \hat{D}) : \hat{D} \text{ is maximal among the subsets } \tilde{D} \text{ of } D \text{ such that } U \cup \tilde{D} \cup K \text{ is consistent}\}.$$

3. The *inference operation* $\mathbf{C}_{D,K}$ associated with a Poole system (D, K) is then defined by $\mathbf{C}_{D,K}(U) = \bigcap E(U)$.

We restrict ourselves to a language J of propositional logic and since we have Proposition 4 only for fuzzy sets with finite support and rational values we will only deal with such ones. Let the set of defaults, denoted by Δ, be a set of evaluated formulas $\Delta = \{\phi_i/a_i\}_{i \in I}$. Let the set of constraints Γ be a set of evaluated formulas. In some cases it is useful to distinguish between the set of evaluated formulas Δ and the fuzzy set of formulas d that is *associated* to it by $d(\phi_i) = a_i$, for all $i \in I$. Neverless the consistency notions are equivalent: A set Δ is *consistent* iff the fuzzy set d is consistent.

Now we have to define operator(s) $\mathbf{C}_{\Delta,\Gamma} : \mathcal{F}(F_J) \longrightarrow \mathcal{F}(F_J)$ by generalizing the ideas used for the two-valued Poole systems. It seems that there are at least two different ways for defining inference operations associated with the system of (evaluated) defaults (Δ, Γ) for fuzzy logic.

1. Let $v \in \mathcal{F}(F_J)$ be a fuzzy set of premises (with finite support). In the first case we choose from the (crisp) set Δ subsets $\hat{\Delta}$ of evaluated formulas that are maximal among all subsets of $\tilde{\Delta}$ and such that $v \cup \Gamma \cup \hat{\Delta}$ is consistent. Then for every $v \in \mathcal{F}(F)$, we can define the (crisp) set $e_1^*(v) \in \mathcal{P}(\mathcal{F}(F))$ of extensions by:

$$e_1^*(v) := \{\boldsymbol{J}_{\mathcal{M}}(v \cup \hat{d}) \mid \text{where } \hat{d} \text{ is associated to a maximal (among the } \tilde{\Delta}) \ \hat{\Delta}\},$$

and the associated operator in the usual way by:

$$\mathbf{C}^1_{\Delta,\Gamma}(v) := \bigcap e_1^*(v).$$

2. If instead of the crisp subsets of Δ we regard fuzzy subsets of the associated fuzzy set d (see the remark above) we get another way to generalize the idea of Poole systems.

   Let d^* be the class of all consistent subsets of d. We define a set $\hat{d_v}^*$ of fuzzy subsets of d, whose elements $\hat{d}_v$ are those maximal elements of d^* s.t. $v \cup g \cup \hat{d}_v$ is consistent, where g is the fuzzy set associated to Γ.

   We define the set of extensions by:

$$e_2^*(v) := \{\boldsymbol{J}_{\mathcal{M}}(v \cup \hat{d}_v) \mid \hat{d}_v \in \hat{d_v}^*\}.$$

   The associated operator will again be defined by intersection of the class $e_2^*(v)$:

$$\mathbf{C}^2_{\Delta,\Gamma}(v) := \bigcap e_2^*(v).$$

Since one can obtain d from Δ uniquely the notion $\mathbf{C}_{\Delta,\Gamma}$ is justified.

Before we will study some properties of the operators defined above we want to have a look at their intentions. In the first case the intention is to regard the defaults as independent from each other. We try to maximize the subsets of defaults which are consistent and we interpret the value $\mathbf{C}^1_{\Delta}(\phi)$ of a formula ϕ as the minimal value it can take in all these cases. The second operator interprets the set of defaults as a fuzzy set, and each extension is one possibility to meet the criteria in Δ jointly as well as possible. The result gives us for each formula the value it has at least in all these optimal extensions.

Proposition 9 *Let (Δ, Γ) be any fuzzy Poole system and $\mathbf{C}^1_{\Delta,\Gamma}$, $\mathbf{C}^2_{\Delta,\Gamma}$ are its associated inference relations. Then the inference frames $(\mathcal{F}(F), \mathbf{C}_{\mathrm{L}}, \mathbf{C}^1_{\Delta,\Gamma})$ and $(\mathcal{F}(F), \mathbf{C}_{\mathrm{L}}, \mathbf{C}^2_{\Delta,\Gamma})$ are supradeductive and cumulative.*

PROOF: We define the selection functions Σ_j, $j = 1, 2$ s.t. $\mathbf{C}^j_{\Delta,\Gamma} = \mathbf{C}_{\Sigma_j}$.
Let $\tilde{d}^*$ be the set of subsets of d consistent with v and Γ. Let I be the index set $\hat{d}^*$ the set of its maximal elements. We denote with $M_i = \mathbf{mod}^{\models}(v \cup \hat{d}_i)$. We show that $\mathbf{C}^2_{\Delta,\emptyset}(v) = \bigcap\{\bigcup_{i \in I}(\mathbf{mod}^{\models}(v) \cap \mathbf{mod}^{\models}(\hat{d}_i))\} = \bigcap\{\bigcup_{i \in I} M_i\}$.
Claim "$\supseteq$": Let ϕ be fixed, then we have:

$$\mathbf{C}^2_{\Delta,\emptyset}(v)(\phi) = \inf_{i \in I}\{\mathbf{C}_L(v \cup \hat{d}_i)(\phi)\}$$
$$= \inf_{i \in I}\{\inf\{m(\phi) \mid m \in M_i\}\}$$

By Corollary 1 for every $i \in I$ there exists a model $m_{i,\phi} \in M_i$ s.t. $m_{i,\phi} = \inf\{m(\phi) \mid m \in M_i\}$. Therefore we can write $\mathbf{C}^2_{\Delta,\emptyset}(v)(\phi) = \inf_{i\in I}\{m_{i,\phi}\}$. It is easy to see that $\mathbf{C}^2_{\Delta,\emptyset}(v)(\phi) \geq \inf\{m(\phi) \mid m \in \bigcup_{i\in I} M_i\}$.
Claim: "$\subseteq$"
On the other hand, if we choose an arbitrary model $m' \in \bigcup_{i\in I} M_i$, then there exists a $j \in I$ s.t. $m' \in M_j$ and therefore $m'(\phi) \geq \inf\{m(\phi) \mid m \in M_j\}$.
This gives us that $\mathbf{C}^2_{\Delta,\emptyset}(v)(\phi) = \inf\{m(\phi) \mid m \in \bigcup_{i\in I} M_i\}$ and so we have identity. This equation enables us to define a function Σ_2:

$$\Sigma_2(\mathbf{mod}^{\models}(v)) = \begin{cases} \mathbf{mod}^{\models}(v) & \text{if the } \hat{d} = \emptyset, \\ \bigcup_{i\in I} \mathbf{mod}^{\models}(v) \cap \mathbf{mod}^{\models}(\hat{d}_i) & \text{otherwise.} \end{cases}$$

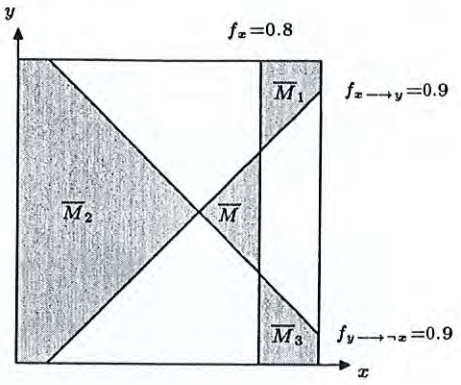
This is of course a selection function since $\Sigma_2(N) \subseteq N$ for any $N \in cl(\mathcal{M})$. Because $\bigcup_{i\in I} M_i$ is a model class, Σ_2 is closed and by Proposition 2 the relation $\leq_{\Sigma_2}$ is antisymmetric. Analogously the proof for Σ_1 works. Moreover, because the index set I is finite there is by Corollary 3 a finite fuzzy set w with $\bigcup_{i\in I} \mathbf{mod}^{\models}(v) \cap \mathbf{mod}^{\models}(\hat{d}_i) = \mathbf{mod}^{\models}(w)$. □

5.2 Example

Let $\Delta = \{(x/0.8), (x \to y/0.9), (y \to \neg x/0.9)\}$ be the set of defaults.
Then there are three subsets which are consistent and maximal among the subsets of Δ:
$D_1 = \{(x/0.8), (x \to y/0.9)\}$,
$D_2 = \{(x \to y/0.9), (y \to \neg x/0.9)\}$ and
$D_3 = \{(x/0.8), (y \to \neg x/0.9)\}$.
The consistent fuzzy subsets $\hat{d}$ which are maximal are those with

$$\hat{d}(\varphi) = \begin{cases} \alpha_{\hat{d}} & \text{if } \varphi = x, \\ \beta_{\hat{d}} & \text{if } \varphi = x \to y, \\ \gamma_{\hat{d}} & \text{if } \varphi = y \to \neg x, \end{cases}$$

where: $\gamma_{\hat{d}} = 3 - 2\alpha_{\hat{d}} - \beta_{\hat{d}}$ and $0.6 \leq \alpha_{\hat{d}} \leq 0.8$, $0.4 \leq \beta_{\hat{d}} \leq 0.9$ and $0.4 \leq \gamma_{\hat{d}} \leq 0.9$. We get the following representation for the model classes:



$\mathbf{mod}^{\models}(D_1) = \{m \in \mathcal{M}_L \mid \overline{m} \in \overline{M_1}\}$
$\mathbf{mod}^{\models}(D_2) = \{m \in \mathcal{M}_L \mid \overline{m} \in \overline{M_2}\}$
$\mathbf{mod}^{\models}(D_3) = \{m \in \mathcal{M}_L \mid \overline{m} \in \overline{M_3}\}$

$\mathbf{mod}^{\models}(\hat{d}) = \{m_{\hat{d}}\}$ with $\overline{m_{\hat{d}}} \in \overline{M}$.

Let v be a given initial valuation, then there exist (by Corollary 3) $\overline{N_1}, \ldots, \overline{N_r} \subseteq [0,1]^2$ s.t.

$$\mathbf{mod}^{\models}(v) = \{m \in \mathcal{M}_L \mid \overline{m} \in \overline{N_1} \cup \ldots \cup \overline{N_r}\}.$$

Let J be the index set such that $v \cup D_j$ is consistent for all $j \in J$. For a formula φ we get:

$$\mathbf{C}^1_{\Delta,\emptyset}(v)(\varphi) = \begin{cases} \min_{j\in J}\{\min_{(\bigcup_{i=1,\ldots,r} N_i)\cap\overline{M_j}}\{f_\varphi(\overline{m})\}\} & \text{if } J \neq \emptyset, \\ \min_{\overline{N_1}}\{f_\varphi(\overline{m})\} \wedge \ldots \wedge \min_{\overline{N_r}}\{f_\varphi(\overline{m})\} & \text{otherwise.} \end{cases}$$

The same procedure for $\mathbf{C}^2_{\Delta,\emptyset}(v)(\varphi)$ gives

$$\mathbf{C}^2_{\Delta,\emptyset}(v)(\varphi) = \begin{cases} \min_{(\bigcup_{i=1,\ldots,r} N_i)\cap\overline{M}}\{f_\varphi(\overline{m})\} & \text{if } (\bigcup_{i=1,\ldots,r} N_i) \cap \overline{M} \neq \emptyset, \\ \min_{\overline{N_1}}\{f_\varphi(\overline{m})\} \wedge \ldots \wedge \min_{\overline{N_r}}\{f_\varphi(\overline{m})\} & \text{otherwise.} \end{cases}$$

5.3 Observations

Even in the case of Poole systems without constraints, there is no inclusion relation between both operators.

Lemma 3 *Let $(\Delta, \emptyset)$ be a Fuzzy Poole system without constraints. Let $\mathbf{C}^1_{\Delta,\emptyset}$ and $\mathbf{C}^2_{\Delta,\emptyset}$ be its associated inference operators as defined in 5.1. Then neither $\mathbf{C}^1_{\Delta,\emptyset} \subseteq \mathbf{C}^2_{\Delta,\emptyset}$ nor $\mathbf{C}^2_{\Delta,\emptyset} \subseteq \mathbf{C}^1_{\Delta,\emptyset}$ hold.*

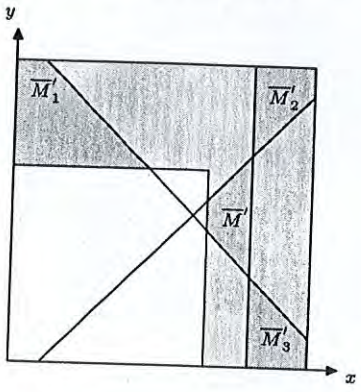
PROOF: To demonstrate this we elaborate our example above: We choose v to be

$$v(\varphi) = \begin{cases} 0.65 & \text{if } \varphi = x \vee y, \\ 0 & \text{otherwise.} \end{cases}$$

Let $\overline{N_v} = \{(0,0) \leq \overline{m} \leq (0.65, 0.65)\}^c$, then the model class of v can be described as:

$$\mathbf{mod}^{\models}(v) = \{m \in \mathcal{M}_L : \overline{m} \in \overline{N_v}\},$$

and we get the following representation ($\overline{N_v}$ is the area coloured with the two darkest grays):



where:

$\overline{M'_1} = \overline{M_1} \cap \overline{N_v}$, $\overline{M'_2} = \overline{M_2} \cap \overline{N_v}$, $\overline{M'_3} = \overline{M_3} \cap \overline{N_v}$
$\overline{M'} = \overline{M} \cap \overline{N_v}$.

Now we only have to find two formulas ψ_1,ψ_2 where

$$\min_{\overline{M'_1} \cup \overline{M'_2} \cup \overline{M'_3}} f_{\psi_1} > f_{\psi_1}(\overline{m}) \text{ for all } \overline{m} \in \overline{M}$$

and where

$$\min_{\overline{M'}} f_{\psi_2} > f_{\psi_2}(\overline{m}) \text{ for all } \overline{m} \in (\overline{M'_1} \cup \overline{M'_2} \cup \overline{M'_3}).$$

Let $\psi_1 = \neg(x \otimes y)$. The truthvaluation function is: $f_{\psi_1}(x,y) = 1 - \max\{0, x+y-1\}$. It takes its minimum in $\overline{m_1} = (1,1) \in \overline{M'_1}$. For all $\overline{m} \in \overline{M'}$ is $f_{\psi_1}(\overline{m}) \geq 0.5$. Thus we have

$$0.5 = \mathbf{C}^2_{\Delta,\emptyset}(v)(\psi_1) > \mathbf{C}^1_{\Delta,\emptyset}(v)(\psi_1) = 0.$$

On the other hand, let $\psi_2 = (\neg x \to x) \to (\neg x \otimes \neg x)$. Then

$$f_{\psi_2}(x,y) = \begin{cases} 1 & \text{if } x \leq 0.25 \\ -4x+2 & \text{if } 0.25 \leq x \leq 0.5 \\ 2x-1 & \text{if } x \geq 0.5 \end{cases},$$

$$\min_{\overline{M'_1} \cup \overline{M'_2} \cup \overline{M'_3}} f_{\psi_2} = \min_{\overline{M'_2}} f_{\psi_2} = 0.4$$

$$\min_{\overline{M'}} f_{\psi_2} = 0.3$$

This gives:

$$0.4 = \mathbf{C}^1_{\Delta,\emptyset}(v)(\psi_2) > \mathbf{C}^2_{\Delta,\emptyset}(v)(\psi_2) = 0.3.$$

□

6 Conclusion

We have exemplified our technique for creating nonmonotonic inference operators by the definition of two variants of fuzzy Poole systems. One of these operators has a really new intuition for the use of constraints. For both operators we have an effective way to compute for any formula its degree of being a conclusion of a finite set of premises. This is due to the fact that we have focussed to the Łukasiewicz semantics. Future work will be to examine how we can get generalizations for other nonmonotonic operators (in particular fuzzy circumscription).

7 Acknowledgements

I wish to thank my supervisors Prof. Gottwald and for Prof. Brewka for their friendly support and many discussions. I wish to thank the students of our GK for critical remarks.

References

[DH97] J. Dietrich and H. Herre. Contributions to the theory of nonmonotonic inference systems. 1997.

[Ger01] G. Gerla. *Fuzzy logic. Mathematical tools for approximate reasoning.* Trends in Logic-Studia Logica Library. 11. Dordrecht: Kluwer Academic Publishers., 2001.

[GN97] S. Gottwald and V. Novák. An approach towards consistency degrees of fuzzy theories. *Int. J. General Systems*, 29(4):499–510, 1997.

[Got00] S. Gottwald. *A Treatise on Many-Valued Logic.* Studies in Logic and Computation 9. Research Studies Press, Baldock, 2000.

[Háj98] P. Hájek. *Metamathematics of Fuzzy Logic.* Kluwer Academic Publishers, Dordrecht, 1998.

[Mak94] D. Makinson. General patterns in nonmonotonic reasoning. In Dov Gabbay, Christopher J. Hogger, and J. A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming, Volume 3: Nonmonotonic and Uncertain Reasoning*, pages 35–111. Oxford University Press, Oxford, 1994.

[Mun94] D. Mundici. A constructive proof of mcnaughtons theorem in infinite-valued logic. *J.of Symbolic Logic*, 59:596–602, 1994.

[NPM99] V. Novák, I. Perfilieva, and J. Mockor. *Mathematical Principles of Fuzzy Logic.* Kluwer Academic Publishers, Dordrecht, 1999.

[Poo89] D. Poole. What the lottery paradox tells us about default reasoning. In H. J. Levesque R. J. Brachman and R. Reiter, editors, *KR '89: Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*, pages 333–340. Morgan Kaufmann, San Mateo, California, 1989.

Representing Action and Change

Towards a Conditional Logic of Actions and Causation

Laura Giordano
Dipartimento di Informatica
Università del Piemonte Orientale
Corso Borsalino 54 - 15100 Alessandria, Italy
E-mail: laura@di.unito.it

Camilla Schwind
MAP, UMR CNRS - MCC N 694
183 Avenue de Luminy, Case 901
13288 Marseille Cedex 9, France,
E-mail: Camilla.Schwind@gamsau.map.archi.fr

Abstract

In this paper we present a new approach to reason about actions and causation which is based on a conditional logic. The conditional implication is interpreted as causal implication. This makes it possible to formalize in a uniform way causal dependencies between actions and their immediate and indirect effects. Furthermore, this new approach provides a natural formalization of concurrent actions and causal dependencies between actions. An abductive semantics is adopted for dealing with the frame problem.

1 Introduction

Causality plays a prominent role in the context of reasoning about actions, as the ramification effects of actions can be regarded as causal dependencies. Many approaches for reasoning about actions have been proposed which allow causal dependencies to be captured [5, 13, 16, 21, 1]. Schwind [18] has studied how causal inferences have been integrated and used in action theories by analyzing four formalisms, which are approaches to action and causality, and comparing them with respect to criteria she established for causality. Namely, the article analyses Lin's approach [12, 13], McCain and Turner's causal theory for action and change [16, 22], Thielscher's theory of ramification and causation [21], and Giordano, Martelli and Schwind's dynamic causal action logic [6]. More recently, Zhang and Foo [23, 2, 3] propose to extend propositional dynamic logic, where actions are modalities, by introducing modalities which are propositions. Sentence "ϕ causes ψ" is represented by the formula $[\phi]\psi$, where $[\phi]$ is a new modality. Note that this representation corresponds to a conditional logic approach, since EPDL formula $[\phi]\psi$ is interpreted as conditional formula $\phi > \psi$. Zhang and Foo's approach has the merit of providing a clean representation of causation as well as a uniform representation of direct and indirect effects of actions. But the notion of causality as defined in EPDL is too strong since it entails material implication. This property, $[\phi]\psi \rightarrow (\phi \rightarrow \psi)$, comes from a crucial axiom introduced for actions in EPDL, and is motivated by the fact that when "ϕ causes ψ" then the state constraint $\phi \rightarrow \psi$ must also hold. While we agree with this argument, we may object that the status of causal laws (and consequently of material implications) in [23] is not that of domain constraints. Instead, causal laws are regarded as domain axioms, which gives the corresponding material implications also the status of domain axioms, so that they can be used for inference in all possible states. In particular, this allows a contrapositive use of material implications $\phi \rightarrow \psi$, which may lead to unintended conclusions. Therefore, though EPDL preserves the directionality of causation (from $[\phi]\psi$ we cannot conclude that $[\neg\psi]\neg\phi$), the fact that causation entails material implication (from $[\phi]\psi$ we can conclude that $\phi \rightarrow \psi$ and thus that $\neg\psi \rightarrow \neg\phi$) anyhow leads to unwanted conclusions when reasoning about the effects of actions. Another difference with our present approach is due to the different modelization of actions. In EPDL, syntactically, actions are not formulas. This makes it impossible to combine assertions about actions with assertions about causality. Concurrent actions cannot be constructed from single actions. It is not possible to express for example that action a and fact B cause the effect C.

We propose to represent causality by a binary logical operator. Our causality operator has the properties of causality relations as discussed in [18] and -perhaps more important- does not have some of the undesired (or doubtful) properties many other approaches have. Our causal operator is not monotonic and does not entail material implication which makes it weaker than the one proposed in [23]. Traditionally, considering a conditional as a causal implication has frequently attracted the attention of logicians in conditional logic ([10, 11]. It allows to model both causal laws and action laws: the causal law "ϕ causes ψ" is represented by the conditional formula $\phi > \psi$ and the action law "action a

causes proposition ψ" is represented by the conditional formula $do(a) > \psi$, where $do(a)$ is a special atomic proposition associated with each action a. This uniform representation of the causal relationship between actions and their results as well as between facts and their effects gives us a great flexibility for handling both concepts in an simple way when representing actions. For example, in this setting, concurrent execution of actions is naturally modelled by conjunctions of the form $do(a_1) \wedge \ldots \wedge do(a_n)$ in the antecedents of conditionals. It is also very natural to express dependency (and independency) relations between actions and actions, actions and propositions, etc.

2 The Causal Action Logic AC

The language, $\mathcal{L}_>$, of our action logic is that of propositional logic augmented with a conditional operator $>$. The set of propositional variables in $\mathcal{L}_>$, Var, includes the set $\{do(a) : a \in \Delta_0\}$, where Δ_0 is a set of *elementary actions* including the "empty" action ϵ. Formulas are defined as usual and the modalities $\Box$ and $\Diamond$ are defined by $\Box A \equiv (\neg A > \bot)$ and $\Diamond A \equiv \neg\Box\neg A$. Intuitively, $\Box A$ means that A necessarily holds, while $\Diamond A$ means that A is possible. If A is an action proposition $do(a)$, $\Diamond do(a)$ means that a is executable. In order to express that a proposition *always* holds, we introduce the additional operator $\overline{\Box}$: $\overline{\Box} A \equiv (\Box A \wedge A)$. The following is an axiom system for logic AC.

We define an axiom system for logic AC as follows:

Definition 1 (AC) *The conditional logic AC is the smallest logic containing the following axioms and deduction rules:*

(CLASS) All classical propositional axioms and inference rules

(CV) $(\neg(A > \neg C) \wedge (A > B)) \rightarrow (A \wedge C) > B)$

(CA) $(A > C) \wedge (B > C) \rightarrow ((A \vee B) > C)$

(CE) $(do(a) > B) \wedge (do(a) > (B > C)) \rightarrow (do(a) > C)$ *where* $a \in \Delta_0$

(MOD) $\Box A \rightarrow (do(a) > A)$ *where* $a \in \Delta_0$

(RCEA) if $\vdash A \leftrightarrow B$, *then* $\vdash (A > C) \equiv (B > C)$

(RCK) if $\vdash A_1 \wedge \ldots \wedge A_n \rightarrow B$, *then* $\vdash (C > A_1) \wedge \ldots \wedge (C > A_n) \rightarrow (C > B)$.

Note that all axioms and inference rules are standard in conditional logics and, in particular, they belong to the axiomatization of Lewis's logic VC (see [17]). However, we have excluded several of the standard axioms of conditional logics such as (ID), (MP) and (CS) since they model unwanted properties of causality. Reflexivity axiom (ID) $A > A$ is excluded since a proposition (or an action) should not cause itself. Axiom (MP) $(A > B) \rightarrow (A \rightarrow B)$ should not hold for causal implication, because we do not want to be able to infer the material implication from a causal rule, since the former is stronger and would give undesired properties to the latter. (CS) $A \wedge B \rightarrow (A > B)$ should not be a property of causal implication since A and B could both hold conjunctively without A being a cause of B.

Axiom (CE) allows action laws and causal laws to interact, it provides the chain effects between causal laws and action laws. (CE) says that the causal consequences of action effects are in turn action effects: if executing action a (in a state S) causes B in the next state S' $(do(a) > B)$, and the causal law $B > C$ holds in S', then executing action a (in S) causes C to hold in S'. (CE) weakens (MP) as it is clear form the following formulation of (CE) $(do(a) > (B > C)) \rightarrow ((do(a) > B) \rightarrow (do(a) > C))$, which is obtained from (MP) by (RCK). (CE) has similarities with the property of transitivity (TRANS) $(do(a) > B) \wedge (B > C) \rightarrow (do(a) > C)$[1], which however requires that the causal law $B > C$ holds in the state S in which the action is executed. As we will see in our action theory causal laws do not necessarily hold in all possible states, as they may have preconditions which make them hold in some states only.[2]

(MOD) allows to deduce $\Box A \rightarrow (do(a_1) > \ldots > (do(a_n) > A)\ldots)$ for any finite sequence of actions $a_1, \ldots a_n$ $(n \geq 0)$ including the empty sequence and meaning that a formula A which is true in every state is also true after the occurrence of any finite sequence of actions. So, the subsequent occurrence of actions structures a world and its subsequent states according to time, although time is not represented explicitly in our formalism.

Entailment $\vdash$ is defined as usual and given a set of formulas E, the deductive closure of E is denoted by $Th(E)$. AC is characterized semantically in terms of selection function models.

Definition 2 *An AC-structure M is a triplet $\langle W, f, [[]]\rangle$, where W is a non-empty set, whose elements are called possible worlds, f, called* the selection function, *is a function of type $\mathcal{L}_> \times W \rightarrow 2^W$, $[[]]$, called the evaluation function, is a function of type $\mathcal{L}_> \rightarrow 2^W$ that assigns a subset of W,*

[1] For standard conditional logic with reflexivity, adding TRANS would collapse the conditional to material implication. But this is not the case for our causal action logic AC, since reflexivity $A > A$ is not an axiom.

[2] As an example of causal law with precondition consider the following one: $\overline{\Box}(at(y,r) \rightarrow (at(z,r) > at(y, next(r)))$ used in Example 4 below, which says that if block y is at r then moving block z to position r causes y to move to a next position.

$[[A]]$ *to each formula* A. *The following conditions have to be fulfilled by* $[[\,]]$:

(1) $[[A \wedge B]] = [[A]] \cap [[B]]$
(2) $[[\neg A]] = W - [[A]]$[3]
(3) $[[A > B]] = \{w : f(A, w) \subseteq [[B]]\}$

We assume that the selection function f *satisfies the following properties which correspond to the axioms of our logic AC:*

(S-RCEA) if $[[A]] = [[B]]$ *then* $f(A, w) = f(B, w)$

(S-CV) if $f(A, w) \cap [[C]] \neq \emptyset$ *then* $f(A \wedge C, w) \subseteq f(A, w)$

(S-CA) $f(A \vee B, w) \subseteq f(A, w) \cup f(B, w)$

(S-CE) if $f(do(a), w) \subseteq [[B]]$
then $f(do(a), w) \subseteq f(B, f(do(a), w))$

(S-MOD) if $f(B, w) \cap [[do(a)]] \neq \emptyset$
then $f(do(a), w) \neq \emptyset$,

where $a \in \Delta_0$ *and* $f(B, f(do(a), w))$ *represents the set of worlds* $\{z \in f(B, x) : x \in f(do(a), w)\}$.

We say that a formula A is true in a AC-structure $M = \langle W, f, [[]]\rangle$ if $[[A]] = W$. We say that a formula α is AC-valid ($\models A$)if it is true in every AC-structure. We also introduce the following notation $S \models_M A$ to say that, given a AC-structure M, a set of formulas S and a formula A, for all $w \in M$ if $w \in [[B]]$ for all $B \in S$, then $w \in [[A]]$.

The above axiom system is sound and complete with respect to this semantics.

Theorem 1 $\models A$ *iff* $\vdash A$

The completeness proof is shown by the canonical model construction [19] and can be found in the appendix. Moreover, the axiomatization is consistent and the logic is decidable. Since the logic AC is weaker than VC, each VC-structure is an AC-structure, which shows that the logic AC is "non-trivial" in some sense.

3 Action Theories

3.1 Domain descriptions

We use atomic propositions $f, f_1, f_2, \ldots \in Var$ for *fluent names*. A *fluent literal*, denoted by l , is a fluent name f or its negation $\neg f$. Given a fluent literal l, such that $l = f$ or $l = \neg f$, we define $|l| = f$. Moreover, we will denote by $\mathcal{F}$ the set of all fluent names, by Lit the set of all fluent literals, and by small greek letters $\alpha, \beta, \ldots$ any formula not containing conditional formulas.

We define a *domain description* as a tuple (Π, $Frame_0$, Obs). Π is a set of *action laws*, *causal laws*, *precondition laws*, *domain constraints* and *causal independency constraints*.

Action laws have the form:

$$\overline{\Box}(\pi \rightarrow (do(a) > \rho)),$$

for an action a with precondition π and effect ρ: executing action a in a state where π holds causes ρ to hold in the resulting state. For action laws with no precondition, i. e. $\pi = true$, we just obtain $\overline{\Box}(do(a) > \rho)$.

Causal laws have the form:

$$\overline{\Box}(\pi \rightarrow (\alpha > \beta)),$$

meaning that "if π holds, then α causes β".

Precondition laws have the form:

$$\overline{\Box}(\pi \equiv \neg(do(a) > \bot)),$$

meaning that "action a is executable iff π holds". According to the definition of $\Diamond$, this is equivalent to $\overline{\Box}(\pi \equiv \Diamond do(a))$.

Domain constraints include formulas of the form:

$$\overline{\Box}\alpha,$$

(meaning that "α always holds").

Causal independency constraints have the form:

$$\overline{\Box}(\neg(A > \neg B)),$$

meaning that A does not cause $\neg B$ (that is, B might be true in a possible situation caused by A). In particular, when the above constraints concern action execution, we have $\Box\neg(do(a) > \neg do(b))$, meaning that the execution of action a does not prevent action b from being executed (does not interfere with its execution). Note that as a consequence of this constraint we have, by (CV), that $(do(a) > C) \rightarrow (do(a) \wedge do(b) > C)$, namely, the effects of action a are also effects of the concurrent execution of a and b, as a does not interfere with b. Moreover, from $(do(a) > \bot) \rightarrow (do(a) \wedge do(b) > \bot)$, we have that if a is not executable it cannot be executed concurrently with b.

$Frame_0$ is a set of pairs $(f, do(a))$, where $f \in \mathcal{F}$ is a fluent and $a \in \Delta_0$ is an elementary action, meaning that f is a *frame fluent* for action a, that is, f is a fluent to which persistency applies when a is executed. Fluents which are

[3] Using the standard boolean equivalences, we obtain $[[A \vee B]] = [[A]] \cup [[B]]$, $[[A \rightarrow B]] = (W - [[A]]) \cup [[B]]$, $[[\top]] = W$, $[[\bot]] = \emptyset$.

non-frame with respect to a do not persist and may change in a nondeterministic way when a occurs.

The set $Frame_0$ defines a sort of *independence* relationship between elementary actions and fluents. It is closely related to dependency (and influence) relations that have been used and studied by several authors including Thielscher [21], Giunchiglia and Lifschitz [7], and Castilho, Gasquet and Herzig [14]. We use $Frame_0$ for defining persistency rules of the form $A_1 > \ldots > A_n > (l \rightarrow (do(a) > l))$ for every literal l, such that $(|l|, a) \in Frame_0$. These persistency rules behave like *defaults*: they belong to an "action extension" whenever no inconsistency arises. The $Frame_0$-relationship is extended to concurrent actions. Let us denote by $Frame$ the extension of $Frame_0$ to concurrent actions. (i) $Frame_0 \subset Frame$; (ii) If $(f, do(a_1)), \ldots, (f, do(a_n)) \in Frame$ then $(f, do(a_1) \wedge \ldots \wedge do(a_n)) \in Frame$.

Obs is a set of observations about the value of fluents in different *states* which we identify with action sequences. Though our language does not provide an explicit representation of time, as we abandon (MP), time can be embedded in the operator $>$. Given the properties of $>$ we assume a delay between happening of an action and occurrence of its effects, while we do not assume any delay between causes and their effects in causal laws. Observations are formulas of the form: $A_1 > \ldots > A_j > \alpha$ (where each A_i is a possibly concurrent action formula of the form $do(a_1) \wedge \ldots \wedge do(a_n)$), meaning that α holds after the concurrent execution of the actions in A_1, then those in A_2, $\ldots$, then those of in A_n. So, every acction occurrence leads from one state to the new state. In particular, we assume an *initial state* characterized by the occurrence of the empty action ϵ. If *Obs* contains observations α about fluents in the initial state this is written as $do(\epsilon) > \alpha$[4].

Sometimes, when we do not want to consider observations, we will then use the notion of *domain frame*, which is a pair $(\Pi, Frame_0)$.

Let us consider the following example from [23], which formalizes an electrical circuit with two serial switches.

Example 1 There is a circuit with two switches and a lamp. If both switches are on, the lamp is alight. One of the switches being off causes the lamp not to be alight. There are two actions of toggling each of the switches. The domain description is the following (for $i = 1, 2$):

Π: $\overline{\Box}(\neg sw_i \rightarrow (do(tg_i) > sw_i))$
$\overline{\Box}(sw_i \rightarrow (do(tg_i) > \neg sw_i))$
$\overline{\Box}(sw_1 \wedge sw_2 > light)$
$\overline{\Box}(\neg sw_1 \vee \neg sw_2 > \neg light)$
$\overline{\Box}(\neg(do(tg_1) > \neg do(tg_2)))$
$\overline{\Box}(\neg(do(tg_2) > \neg do(tg_1)))$
Obs: $do(\epsilon) > (\neg sw_1 \wedge \neg sw_2 \wedge \neg light)$
$Frame_0 = \{(f, a) : a \in \Delta_0, f \in \mathcal{F}\}$.

The first two rules in Π describe the immediate effects of the action of toggling a switch. The third and forth rule are causal laws which describe the dependencies of the light on the status of the switches. The last two laws are constraints saying that the two actions tg_1 and tg_2 do not interfere. All fluents are supposed to be persistent and the actions tg_1 and tg_2 are independent. As we will see, from the above domain description we can derive $do(tg_1) > \neg light$, $do(tg_1) > do(tg_2) > light$ and $do(tg_1) \wedge do(tg_2) > (sw_1 \wedge sw_2 \wedge light)$ (as actions $do(tg_1)$ and $do(tg_i)$ are independent).

Observe that we could have avoided introducing $\neg light$ in the initial state, as it can be derived, for instance, from $\neg sw_1$: from $do(\epsilon) > \neg sw_1$ and the forth action law we can derive $do(\epsilon) > \neg light$ by (CE).

The axiom (CA) makes it possible to deduce consequences of actions even when it is not deterministically known which action occurs.

Example 2 If the temperature is low, then going to swim causes you to get a cold. If you have no umbrella, then raining causes you to get cold. We have the following domain description:

Π: $\overline{\Box}(cold \rightarrow (do(swim) > get_cold))$
$\overline{\Box}(no_umbrella \rightarrow (do(rain) > get_cold))$
Obs: $do(\epsilon) > (cold \wedge no_umbrella)$
$Frame_0 = \{(f, a) : a \in \Delta_0, f \in \mathcal{F}\}$

From this theory, we can derive $do(swim) \vee do(rain) > get_cold$.

The following example is inspired by one discussed by Halpern and Pearl in [8]. Suppose that heavy rain occurred in April causing wet forests in May and electrical storms in May and June. The lightning in May did not cause forest fires since the forests were still wet. But the lightning in June did since the forest dried in the meantime. Pearl and Halpern argue that the April rain caused that the fire did not occur in May and occurred in June instead. We think rather that the April rain prevents the fire from occurring in may which is expressed by the precondition of the action law.

Example 3 i is ranging over months, such that $i+1$ is the month following i.

Π: $\overline{\Box}(do(rain_i) > wet_forest_{i+1})$

[4] In the following, when identifying a state with an action sequence $A_1, \ldots, A_j$, we will implicitly assume that $A_1 = do(\epsilon)$. Also, in a conditional formula $A_1 > \ldots > A_j > \alpha$, we will assume that $A_1 = do(\epsilon)$.

$\overline{\Box}(do(rain_i) > (do(lightning_{i+1}) \wedge do(lightning_{i+2})))$
$\overline{\Box}(wet_forest_i \rightarrow (do(lightning_i) > \neg forest_fire_i))$
$\overline{\Box}(\neg wet_forest_i \rightarrow (do(lightning_i) > forest_fire_i))$
$\overline{\Box}(do(sun_i) > \neg wet_forest_{i+1})$
$\overline{\Box}(\neg(do(sun_i) > \neg do(rain_j)))$ *(for* $i \neq j$*)*
$\overline{\Box}(\neg(do(rain_j) > \neg do(sun_i)))$ *(for* $i \neq j$*)*
Obs: $\{\}$
$Frame_0 = \{(f, a) : a \in \Delta_0, f \in \mathcal{F}\}$.

The second action law allows to say that rain in April causes electrical storms in May and June.

From the first two action laws we derive $do(rain_{April}) > (wet_forest_{May} \wedge do(lightning_{May}) \wedge do(lightning_{June}))$. Then by (CE), from the third action law (for $i = May$) we get $do(rain_{April}) > \neg forest_fire_{May}$.

Moreover, it hods that $\overline{\Box}(do(sun_{May}) > \neg wet_forest_{June})$ (fifth action law) and, as actions $do(rain_{April})$ and $do(sun_{May})$ are independent, we can derive $do(rain_{April}) \wedge do(sun_{May}) > (\neg wet_forest_{June} \wedge do(lightning_{June}))$. Then, by (CE), from the forth action law (for $i = June$), we conclude: $do(rain_{April}) \wedge do(sun_{May}) > forest_fire_{June}$.

Therefore, we have that $do(rain_{April}) \wedge do(sun_{May}) > \neg forest_fire_{May} \wedge forest_fire_{June}$.

The following example, taken from [5], involves causal laws with preconditions.

Example 4 Consider the following scenario, where a number of blocks are in a sequence: when the first block, a, is pushed from the place p_1 to the place p_2, all other blocks move also to the next place. Let $p_1, \ldots p_n$ be places and a, b, c be blocks, and let $push(x, p)$ be the action which consists in pushing the block x from the place p to the next place $next(p)$.

Π: $\overline{\Box}(at(x,p) \rightarrow (do(push(x,p)) > at(x, next(p))))$
$\overline{\Box}(at(y,r) \wedge (at(z,r) > at(y, next(r))))$, for $z \neq y$
$\overline{\Box}(at(x,p) > \neg at(x,q))$, for $p \neq q$
Obs: $do(\epsilon) > (at(a,p_1) \wedge at(b,p_2) \wedge at(c,p_3))$
$Frame_0 = \{(f, a) : a \in \Delta_0, f \in \mathcal{F}\}$

The first (action) law says that, if block x is in p, pushing x from the place p moves it to the next place $next(p)$. The second (causal) law says that if block y is at r then moving block z to position r causes y to move to a next position. The third laws causes block x not to be at q if it is at p (different from q).

Given the initial state, it holds that: $do(push(a, p_1)) > (at(a, p_2) \wedge at(b, p_3) \wedge at(c, p_4))$.

3.2 Extensions for a domain description

In order to address the frame problem, we introduce a set of persistency laws, which can be assumed in each extension. Persistency laws are essentially frame axioms. They are used, in addition to the formulas in Π, to determine the next state when an action is performed. As a difference with the formulas in Π, persistency laws are *defeasible*. They are regarded as assumptions to be maximized. Changes in the world are minimized by maximizing these assumptions. Moreover, persistency laws have to be assumed if this does not lead to inconsistencies.

Let $A_1, \ldots, A_n$ be (possibly) concurrent actions of the form $do(a_1) \wedge \ldots \wedge do(a_m)$ (for $m = 1$ we have an atomic action). We introduce a set of *persistency laws* of the form $A_1 > \ldots > A_n > (l \rightarrow (A > l))$ for every sequence of (concurrent) actions $A_1, \ldots, A_n$ and for every fluent literal l which is a frame fluent with respect to the (concurrent) action A (according to the definition of $Frame$ in the last subsection), that is, for every fluent literal l which is frame for *every elementary action* in A. The persistency law says, that, "if l holds in the state obtained by executing the sequence of actions $A_1, \ldots, A_n$, then l persists after executing action A in that state" [5].

Our notion of extension will require to introduce two different kinds of assumptions. The first kind of assumptions, as we have seen, are persistency assumptions. Given a set $Frame$ of frame fluents, the set of persistency assumptions $WP_{A_1,\ldots,A_n}$ is defined as follows:

$$WP_{A_1,\ldots,A_n} = \{A_1 > \ldots > A_n > (l \rightarrow (A > l))) : (|l|, A) \in Frame\}.$$

Note that the set of persistency assumptions has been defined relative to a sequence of (concurrent) actions, that is, a state.

In addition to persistency assumptions, we introduce another kind of assumptions, which are needed to deal with non frame fluents. If a fluent f is not persistent with respect to a concurrent action A then, in the state obtained after executing A, the value of f might be either true or false. Hence, we introduce assumptions which allow to assume, in any state, the value true or false for each nonframe fluent f, as well as assumptions for all fluents in the initial state. Given a set $Frame$ of frame fluents, we define the set of assumptions $Ass_{A_1,\ldots,A_n}$ (relative to a sequence $A_1, \ldots, A_n$) as follows:

$$Ass_{A_1,\ldots,A_n} = \{A_1 > \ldots > A_n > l : (|l|, A_n) \notin Frame\} \cup \{do(\epsilon) > l : l \in Lit\}$$

[5] Notice that introducing persistency laws of the form $\overline{\Box}(l \rightarrow (A > l))$ wouldn't be enough to deal with the persistency of literals at each different state.

We represent a generic assumption in this set by $A_1 > \ldots > A_n > l$, which includes assumptions on the initial state (for $n = 0$).

Now we can introduce our notion of extension, first for domain frames $(\Pi, Frame_0)$, and then for domain descriptions $(\Pi, Frame_0, Obs)$. An extension E of a domain frame is obtained by augmenting Π by as many as possible persistency laws, such that E is consistent. We define an extension *relative to a state*, which can be identified by the sequence of actions $A_1, \ldots, A_n$ leading to that state.

Definition 3 *An* extension *of a domain frame* $D = (\Pi, Frame)$ *relative to the action sequence* $A_1, \ldots, A_n$ *is a set* $E = Th(\Pi \cup WP' \cup F)$, *such that* $WP' \subseteq WP_{A_1,\ldots,A_n}$, $F \subseteq Ass_{A_1,\ldots,A_n}$ *and*

a) *if* $A_1 > \ldots > A_{n-1} > (l \rightarrow (A_n > l)) \in WP_{A_1,\ldots,A_n}$ *then:*
$A_1 > \ldots > A_{n-1} > (l \rightarrow (A_n > l)) \in WP'$
$\Longleftrightarrow A_1 > \ldots > A_n > \neg l \notin E$

b) *if* $A_1 > \ldots > A_n > l \in Ass_{A_1,\ldots,A_n}$ *then*
$A_1 > \ldots > A_n > l \in F \iff A_1 > \ldots > A_n > \neg l \notin E$.

The $\Rightarrow$-part of condition a) is a consistency condition, which guarantees that a persistency axiom $A_1 > \ldots > A_{n-1} > (l \rightarrow (A_n > l))$ cannot be assumed in WP' if $\neg l$ can be deduced as an immediate or indirect effect of the action A_n. We say that the formula $A_1 > \ldots > A_n > \neg l$ blocks the persistency axiom. The $\Leftarrow$-part of condition a) is a maximality condition which forces the persistency axiom to be assumed in WP', if the formula $A_1 > \ldots > A_n > \neg l$ is not proved. Condition b) forces each state of an extension to be complete: for all finite sequences of actions $A_1, \ldots, A_n$ each non persistent fluent must be assumed to be true or false in the state obtained after executing them. In particular, since the sequence of actions may be empty, the initial state has to be complete in a given extension E. This is essential for dealing with domain descriptions in which the initial state is incompletely specified and with postdiction. The conditions above have a clear similarity with the applicability conditions for a default rule in an extension.

Definition 4 E *is an* extension for a domain description $(\Pi, Frame, Obs)$ relative to the action sequence $A_1, \ldots, A_n$ *if it is an extension for the domain frame* $(\Pi, Frame)$ *relative to the action sequence* $A_1, \ldots, A_n$ *and* $E \vdash Obs$.

Notice that first we have defined extensions of a domain frame $(\Pi, Frame)$; then we have used the observations in Obs to filter out those extensions which do not satisfy them. As a difference with [4, 5] an extension only describes a single course of actions, and assumptions are localized to that sequence of actions. In this way, we deal with concurrent actions without the need of introducing two different modalities for actions, which in [4] are called open and closed modalities and they are introduced to avoid that the (AND) law $(do(a) > C \rightarrow do(a) \wedge do(b) > C)$ is applied to the non-monotonic consequences of actions, derived by means of the persistency assumptions. In our present approach, we can derive $do(a) \wedge do(b) > C$ from $do(a) > C$ using the axiom (CV) provided a and b are independent. Independency is formulated in the action language by $\neg(do(a) > \neg do(b)) \wedge \neg(do(b) > \neg do(a))$.

Let us consider again Example 1. Relative to the action sequence $\{do(\epsilon)\}, \{do(tg_1)\}, \{do(tg_2)\}$ we get one extension E containing the frame laws

$$\neg light \rightarrow (do(tg_1) > \neg light,$$
$$\neg sw_2 \rightarrow (do(tg_1) > \neg sw_2,$$
$$do(tg_1) > (sw_1 \rightarrow (do(tg_2) > sw_1)),$$

in which the following sentences hold:

(1) $do(tg_1) > \neg light$,
(2) $do(tg_1) > (do(tg_2) > light)$,
(3) $do(tg_1) \wedge do(tg_2) > light$,
(4) $do(tg_1) > ((do(tg_1) \wedge do(tg_2)) > \neg light)$.

An extension E relative to $A_1, \ldots, A_n$ determines an initial state and a transition function among the states obtained by executing actions $A_1, \ldots, A_n$. In particular, the *state* reachable through an action sequence $A_1, \ldots, A_j$ $(0 \leq j \leq n)$ in E can be defined as :

$$S^E_{A_1,\ldots,A_j} = \{l : E \vdash A_1 > \ldots > A_j > l\}$$

(where S^E_ϵ represents the initial state). Due to condition (b) of definition 3, we can prove that each state $S^E_{A_1;\ldots;A_j}$ is *complete*: for each fluent f, it contains either f or $\neg f$. Moreover, it can be shown that the state obtained after execution of the sequence of actions $A_1, \ldots, A_n$, is only determined by the assumptions made from the initial state up to that state.

Referring to Example 1,the extension E above relative to the action sequence $\{do(\epsilon)\}, \{do(tg_1)\}, \{do(tg_2)\}$ determines the following states:

$$S^E_\epsilon = \{\neg sw_1, \neg sw_2, \neg light\}$$
$$S^E_{\{do(\epsilon)\},\{do(tg_1)\}} = \{sw_1, \neg sw_2, \neg light\}$$
$$S^E_{\{do(\epsilon)\},\{do(tg_1)\},\{do(tg_2)\}} = \{sw_1, sw_2, light\}$$

Observe that for the domain description in Example 1 we do not obtain the unexpected extension in which $do(tg_1) >$

$(do(tg_2) > (\neg sw_1 \wedge sw_2 \wedge \neg light)$ holds: we do not want to accept that toggling sw_2 in the state $\{sw_1, \neg sw_2, \neg light\}$ mysteriously changes the position of sw_1 and lets $\neg light$ persist. To avoid this extension it is essential that causal rules are directional (see [1, 15, 12, 21]). Indeed, the causal rules in Π are different from the constraint $\overline{\Box}((sw_1 \leftrightarrow sw_2) \rightarrow light)$ and, in particular, they do not entail the formula $\neg sw_1 \wedge \neg light \rightarrow sw_2$. As observed in [12] and [21], though this formula must be clearly true in any state, it should not be applied for making causal inferences. In our formalism, contraposition of causal implication is ruled out by the fact that the conditional $>$ is not reflexive: from $\overline{\Box}(\alpha > \beta)$ and $\overline{\Box}\neg\beta$ we cannot conclude $\overline{\Box}\neg\alpha$. On the other hand, it is easy to see that, in any state of any extension, if $\alpha > \beta$ holds, and α holds, β also holds.

Our solution to the frame problem is an abductive solution and is very different from the solution proposed for EPDL in [3]. There persistency laws of the form $l \rightarrow [a]l$ are added explicitly at every state. In EPDL, persistency laws are not global to an extension but they have to be added state by state, according to which action is expected. In our theory, the frame problem is solved globally by minimizing changes modulo causation. As a further difference, in [3] unexpected solutions can be obtained by adding persistency laws as above to the domain description. As observed by Zhang and Foo (see [3], Example 4.1) in the circuit example above the state $S_1 = \{sw_1, \neg sw_2, \neg light\}$ has two possible next states under action $toggle_2$, namely $S_2' = \{sw_1, sw_2, light\}$ and $S_2'' = \{\neg sw_1, sw_2, \neg light\}$. The second one is unexpected.

This behaviour is a side effect of (MP), which holds for EPDL and allows the material implication to be derived form the causal implication. To overcome this problem, Zhang and Foo propose an alternative approach to define the next-state function which makes use of a fixpoint property in the style of McCain and Turner's fixpiont property [15]. Their definition employs the causal operator for determining whether the indirect effects of the action are caused by its immediate effects together with the unchanged part of the state, according to the causal laws. It has to be observed, that this definition of the next state function does not require any integrated use of causal laws and action laws in the theory. In fact, "if the direct effects of an action have been given, $EPDL^-$ [that is, the logic obtained from EPDL when the set of action symbols is empty] is enough to determine how effects of actions are propagated by causal laws" [3]. On the contrary, our solution to the frame problem in the conditional logic CA relies on an integrated use of action laws and causal laws to derive conclusion about actions effects.

A domain description may have extensions containing a formula $A > \bot$. Consider the following example also mentioned by [15]:

Π: $\quad \overline{\Box}(do(a) > p) \quad \overline{\Box}(q > \neg p)$
Obs: $\quad do(\epsilon) > (q \wedge \neg p)$
$Frame_0 = \{(f, a) : a \in \Delta_0, f \in \mathcal{F}\}$.

If $q \wedge \neg p$ holds in the initial state, performing action a makes p true, but this cannot block the persistency of q since $\neg q$ cannot be derived from p since the causal rule is not contrapositive. However, assuming that q persists after the action leads to $do(a) > q$, since $q > \neg p$, by (CE), we derive $do(a) > \neg p$ from which we get together with $do(a) > p$, $do(a) > \bot$. This means that the execution of action a leads to an inconsistent state, i.e. a cannot be executed in the state in which $q \wedge \neg p$ holds.

Another situation which can lead to an inconsistent state can occur when two conflicting actions are executed simultaneously. In [4], conflicting actions could occur leading to inconsistent states. In this present theory, the concurrent application of two actions a and b is only possible when these actions are independent, i.e. when Π contains $\overline{\Box}(\neg(do(a) > \neg do(b)) \wedge \neg(do(b) > \neg do(a)))$.

Example 5 Consider a swinging door and two actions $push_in$ and $push_out$ the first one opening the door by pushing from out-side to open it and the second by pushing it in the opposite direction. We get the following formalization:

Π: $\overline{\Box}(do(push_in) > open_in)$
$\overline{\Box}(do(push_out) > open_out)$
$\overline{\Box}(open_in > \neg open_out)$
$\overline{\Box}(open_out > \neg open_in)$
$\overline{\Box}(\neg(do(push_in) > \neg do(push_out)))$
$\overline{\Box}(\neg(do(push_out) > \neg do(push_in)))$
$Frame_0 = \{(f, a) : a \in \Delta_0, f \in \mathcal{F}\}$.

Both actions are independent. But when trying to perform them at the same moment, nothing would happen, because there is a conflict between the effects of the two actions. The door stays in the same position. All the extensions of the theory contain the formulas:

(1) $do(push_in) > open_in$,
(2) $do(push_out) > open_out$,
(3) $do(push_in) \wedge do(push_out) > \bot$.

Let us reconsider example 1 modified such that the two switches form a two-way wiring.

Example 6 There is a two-way wiring circuit with two two-way switches and a lamp. There are two actions of toggling each of the switches and the lamp is alight when the two switches are in the same position. Toggling any

of the switches changes the light (from on to off or viceversa). We have, for $i = 1, 2$:

Π: $\overline{\Box}(\neg sw_i \rightarrow (do(tg_i) > sw_i))$
$\Box(sw_i \rightarrow (do(tg_i) > \neg sw_i))$
$\Box(sw_1 \leftrightarrow sw_2 > light)$
$\overline{\Box}(\neg(sw_1 \leftrightarrow sw_2) > \neg light)$
$\overline{\Box}(\neg(do(tg_1) > \neg do(tg_2)))$
$\overline{\Box}(\neg(do(tg_2) > \neg do(tg_1)))$
Obs: $do(\epsilon) > (sw_1 \wedge \neg sw_2 \wedge \neg light)$
$Frame_0 = \{(f, a) : a \in \Delta_0, f \in \mathcal{F}\}$.

As before, the first four rules in Π describe the immediate effects of the action of toggling one of the switches. All fluents are regarded as being persistent and the two toggling actions are independent. The domain description $(\Pi, Frame_0, Obs)$ has one extension E relative to the action sequence $\{do(tg_1)\}, \{do(tg_2)\}$ containing the frame axioms:

$$\neg sw_2 \rightarrow (do(tg_1) > \neg sw_2)$$
$$do(tg_1) > (\neg sw_1 \rightarrow (do(tg_2) > \neg sw_1)).$$

The following formulas are derivable in E:

(1) $do(tg_1) > light$,
(2) $do(tg_1) > (do(tg_2) > \neg light)$,
(3) $(do(tg_1) \wedge do(tg_2)) > \neg light$,
(4) $do(tg_1) > ((do(tg_1) \wedge do(tg_2)) > light)$,
(5) $\overline{\Box}(light \rightarrow (do(tg_1) \wedge do(tg_2) > light))$,
(6) $\overline{\Box}(\neg light \rightarrow (do(tg_1) \wedge do(tg_2) > \neg light))$.

We can see that the concurrent execution of both toggling actions never changes the status of the lamp, who remains alight (5) or not alight (6). This is true in the domain frame independently of specific observations. Moreover, (3), (5) and (6) are monotonically derivable from the domain description and they hold in all extensions.

4 Conclusion

We have presented a new logical approach to actions and causality which uses a single implication $>$ for causal consequence. Action execution and causal implication are represented uniformly. This makes it possible to integrate reasoning about mutual action dependence or independence into the language of the logic itself. This possibility distinguishes our approach from many other approaches, for example [14], who formulate dependencies outside the logic. Our action language can handle (co-operating, independent and conflicting) concurrent actions in a natural way without adding extra formal devices, and we believe that the language can be naturally extended to handle other boolean expressions concerning action performance.

References

[1] G. Brewka and J. Hertzberg, 'How to do things with worlds: on formalizing actions and plans', *Journal of Logic and Computation*, 3(5), (1993).

[2] Nonman Foo, Dongmo Zhang, Yan Zhang, Samir Chopra, and Bao Vo, 'Encoding solutions of the frame problem in dynamic logic', in *Logic Programming and Nonmonotonic Reasoning*, eds., T. Eiter, W. Faber, and M. Truszczynski, volume 2173 of *LNAI*, pp. 240–253. Springer, (2001).

[3] Norman Foo and Dongmo Zhang, *Dealing with the ramification Problem in extended Propositional Dynamic logic*, volume 3 of *CSLI*, 2000. to appear.

[4] Laura Giordano, Alberto Martelli, and Camilla Schwind, 'Dealing with concurrent actions in modal action logic', in *Proceedings of the 13th European Conference on Artificial Intelligence*, ed., H. Prade, pp. 537–541. John Wiley & Sons, Ltd, (1998).

[5] L. Giordano, A. Martelli and C. Schwind. Ramification and Causality in a Modal Action Logic. *Journal of Logic and Computation*, vol. 10 No. 5, pp. 625-662, 2000.

[6] Laura Giordano, Alberto Martelli, and Camilla Schwind, 'Reasoning about actions in dynamic linear time temporal logic.', in *Proceedings of the 3rd International Conference on Formal and Applied Practical Reasoning, FAPR 2000*, London, UK. Journal of the IGPL, Vol. 9, No. 2, pp. 289-303, March 2001.

[7] E. Giunchiglia and V. Lifschitz, 'Dependent fluents', In IJCAI'95 [9], pp. 1964–1969.

[8] Joseph Halpern and Judea Pearl, 'Causes and Explanations: A structural model Approach, Part I', In *Proceedings of the 17th International Joint Conference on Artificial Intelligence, IJCAI'01*, B. Nebel, ed., pp. 27-34, Seattle, USA, 2001, Morgan Kaufmann.

[9] *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, Montreal, Canada, 1995. Morgan Kaufmann.

[10] Frank Jackson, 'A causal theory of counterfactuals', *Australian Journal of Philosophy*, **55**(1), 3–21, (1977).

[11] David Lewis, 'Causation', *Journal of Philosophy*, **70**, 556–567, (1973).

[12] Fangzhen Lin, 'Embracing causality in specifying the indirect effects of actions', In IJCAI'95 [9], pp. 1985–1991.

[13] Fangzhen Lin, 'Embracing causality in specifying the indeterminate effects of actions', in *Proceedings of the 133th National Conference on Artificial Intelligence*, pp. 1985–1991. AAAI Press / The MIT Press, (August 1996).

[14] Andreas Herzig Marcos A. Castilho, Olivier Gasquet, 'Formalizing action and change in modal logic i: the frame problem', *Journal of Logic and Computation*, **9**(5), 701–735, (1999).

[15] N. McCain and H. Turner, 'A causal theory of ramifications and qualifications', In IJCAI'95 [9], pp. 1978–1984.

[16] Norman McCain and Hudson Turner, 'Causal theories of action and change', in *Proceedings of the 14th National Conference on Artificial Intelligence*, pp. 460–465. AAAI Press / The MIT Press, (July 1997).

[17] D. Nute, *Topics in Conditional Logic*, Reidel Publishing Company, Dordrecht, 1984.

[18] C. Schwind, 'Causality in action theories', *Electronic Articles in Computer and Information Science*, **3**(A), 27–50, (1999).

[19] Kurt Segerberg, 'Notes on conditional logic', *Studia Logica*, **48**, 157–168, (1989).

[20] J. R. Shoenfield, 'Mathematical Logic', Addison - Wesley Publishing Company, (1967).

[21] Michael Thielscher, 'Ramification and causality', *Artificial Intelligence*, **89**(1-2), 317–364, (1997).

[22] Hudson Turner, 'A logic of universal causation', *Artificial Intelligence*, **113**, 87–123, (1999).

[23] Dongmo Zhang and Norman Foo, 'Epld: A logic for causal reasoning', in *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pp. 131–136, Seattle, USA, (2001). Morgan Kaufmann.

Appendix

Proof of the completeness theorem

The completeness is shown by the construction of a canonical model. We construct a model such that for any consistent formula A (i.e. any formula A, such that $\nvdash \neg A$), there is a world in this model satisfying A. Moreover, we show that the semantic properties of AC (S-CV, S-OR, etc.) hold in the canonical model.

Definition 5

1. *A set of formulas* Γ *is called* inconsistent *iff there is a finite subset of* Γ, $\{F_1, \ldots F_n\}$ *such that* $\vdash \neg F_1 \vee \neg F_2 \vee \ldots \neg F_n$. Γ *is called* consistent *if it is not inconsistent. If* Γ *contains only one formula* F, *we say that* F *is (in)consistent.*
2. *A set of formulas* Γ *is called* maximal inconsistent *iff it is consistent and if for any formula* F *not in* Γ, $\Gamma \cup \{F\}$ *is inconsistent.*

We presuppose a number of properties of maximal consistent formula sets the proof of which can be found in most text books of formal logic (see e.g. [20].

The canonical model CM is defined by $CM =< W, f, [[]] >$ where

1. W is the set of all maximal consistent formula sets of AC.
   We set $\|A\| = \{w : w \in W \text{ and } A \in w\}$ for any formula A.
2. $f(A, w) = \{w' : w' \in W \text{ and } w^A \subseteq w'\}$,
   where, for any $w \in W$, $w^A = \{B : A > B \in w\}$
3. for any atom $p \in ATM$, $[[p]] = \{w : p \in w\}$

$[[]]$ is extended to formulas as usual.

We first will show that for any formula A, $\|A\| = [[A]]$. This is proven by induction on the degree of the conditional formulas. First we need the following lemma concerning conditionals.

Lemma 1 *For any conditional formula we have* $A > B \in w$ *iff for all* $w' \in f(A, w)$, $B \in w'$

Proof: The first half follows immediately from the definition of the selection function f. *For the second half, we first observe that* $w^A \cup \{\neg B\}$ *is an inconsistent formula set. Suppose for the contrary, that* $w^A \cup \{\neg B\}$ *is consistent. Then it is included in a maximal consistent formula set* $w' \in W$, $w^A \cup \{\neg B\} \subseteq w'$. *But then* $w^A \subseteq w'$, *which means that* $w' \in f(A, w)$. *From this follows by our precondition that* $B \in w'$. *This is a contradiction to* $\neg B \in w'$, *since* w' *is consistent. Since* $w^A \cup \{\neg B\}$ *is inconsistent, there are formulas* $\{F_1, \ldots F_n\} \subseteq w^A$ *such that* $\vdash \neg F_1 \vee \neg F_2 \vee \ldots \vee B$. *By the rules of propositional calculus and rule RCK, we conclude* $\vdash (A > F_1) \wedge (A > F_2) \wedge \ldots (A > F_n) \rightarrow (A > B)$. *But* $A > F_i \in w$ *for* $1 \leq i \leq n$, *hence* $A > B \in w$ *by the maximality of* w. Q.E.D.

We now proceed proving $\|A\| = [[A]]$ for arbitrary formula A. This is shown by induction on the degree of A.

- If A is a classical formula (degree(A) = 0), then by construction of the canonical model and the definition

of the valuation function $[[]]$, we get straightforwardly that $w \in \|A\|$ iff $w \in [[A]]$, which means that $\|A\| = [[A]]$.

- Suppose, that we have $\|F\| = [[F]]$ for all formulas F which have a degree less than n. Let be $A > B$ a formula of degree n. Let be $w \in \|A > B\|$. By the definition of $\|A > B\|$, this is equivalent to $A > B \in w$. By lemma 1, this is the case iff for all $w' \in f(A, w)$, $B \in w$. By the definition of $\|B\|$, we get equivalently $\forall w' \in f(A, w)$, $w \in \|B\|$. By induction hypothesis, since $degree(B) < n$, we can replace $\|B\|$ by $[[B]]$ and we get $\forall w' \in f(A, w)$, $w \in [[B]]$. And this is the case iff $f(A, w) \subseteq [[B]]$ which means that $w \in [[A > B]]$.

It remains to show that the canonical model CM has the properties required by our logic AC, provided the corresponding axioms belong to the logic (S-CV, S-CV, ...)

- S-RCEA if $[[A]] = [[B]]$ then $f(A, w) = f(B, w)$
  If $[[A]] = [[B]]$ by the maximality of w, we get that $A \leftrightarrow B \in w$. By RCEA, it follows that $A > C \leftrightarrow B > C \in w$, from which we get $f(A, w) = f(B, w)$.

- (S-CV) if $f(A, w) \cap [[C]] \neq \emptyset$ then $f(A \wedge C, w) \subseteq f(A, w)$
  Let be $w' \in f(A \wedge C, w)$ iff $w^{A \wedge C} \subseteq w'$. By precondition, we have $f(A, w) \cap [[C]] \neq \emptyset$ which means that $\neg(A > \neg C) \in w$. This yields using axiom CV, $(A > B) \rightarrow (A \wedge C > B) \in w$. From this we conclude $\{B : A > B \in w\} \subseteq \{B : A \wedge C > B \in w\}$ which means that $w^A \subseteq w^{A \wedge C}$. Hence we get $w^A \subseteq w'$, i.e. $w' \in f(A, w)$.

- (S-CA) $f(A \vee B, w) \subseteq f(A, w) \cup f(B, w)$
  Suppose for the contrary that there is $w_1 \in W$ such that $w_1 \notin f(A, w)$ and $w_1 \notin f(B, w)$. Then there are formulas F_1 and F_2 such that $A > F_1 \in w$ and $F_1 \notin w_1$ and $B > F_2 \in w$ and $F_2 \notin w_1$ by the definition of the selection function of the canonical model. Since w_1 is maximal consistent, we have that $\neg F_1 \in w_1$ and $\neg F_2 \in w_1$. By RCK and the maximality of w_1 , we get $A > F_1 \vee F_2 \in w_1$ and $B > F_1 \vee F_2 \in w_1$. By axiom CA this yields $A \vee B > F_1 \vee F_2 \in w_1$, from which follows that $F_1 \vee F_2 \in w^{A \vee B}$. Hence we cannot have $w^{A \vee B} \subseteq w_1$ because this would contradict $\neg F_1 \in w_1$ and $\neg F_2 \in w_1$ (maximality of w_1). Therefore $w_1 \notin f(A \vee B, w)$.

- (S-CHAINING) if $f(A, w) \subseteq [[B]]$ then $f(A, w) \subseteq f(B, f(A, w))$
  where $f(B, f(A, w))$ represent the set of worlds $\{z \in f(B, x) : x \in f(A, w)\}$.
  By the precondition, we have $A > B \in w$; by axiom CE, we then get $(A > (B > C)) \rightarrow (A > C) \in w$. But this means that $\{F : B > F \in w^A\} \subseteq \{F : A > F \in w\}$, where $\{F : A > F \in w\} = w^A$. Let be $w' \in f(A, w)$, i.e. $w^A \subseteq w'$. Then we have $\{F : B > F \in w^A\} \subseteq w'$. This means that $w' \in f(B, f(A, w))$.

- (S-MOD) if $f(B, w) \cap [[A]] \neq \emptyset$ then $f(A, w) \neq \emptyset$
  If $f(A, w) = \emptyset$ then $A > \bot \in w$, from which follows that $B > \neg A \in w$ by axiom MOD. This is equivalent to $f(B, w) \subseteq [[\neg A]]$, which gives equivalently $f(B, w) \not\subseteq [[A]]$, i.e. $f(B, w) \cap [[A]] = \emptyset$.

Proof of the completeness theorem: Proof: Let be A a formula not derivable in AC. Then $\not\vdash A$, i.e. $\{\neg A\}$ is consistent. Then there is a maximal set of formulas w such that $\neg A \in w$. And this means that the canonical model CM satisfies $\neg A$, i.e. $CM, w \not\models A$. Q.E.D.

It depends on the context! A decidable logic of actions and plans based on a ternary dependence relation

Marcos A. Castilho*
Depto. de Informática
Universidade Federal do Paraná
Curitiba – Brazil

Andreas Herzig†
IRIT
Université Paul Sabatier
Toulouse – France

Ivan J. Varzinczak‡
Depto. de Informática
Universidade Federal do Paraná
Curitiba – Brazil

Abstract

In this paper we argue for a weak form of causality in terms of a dependence relation involving actions, atoms and formulae in order to deal with the frame and ramification problems. This relation allows the atoms to change their value without forcing or causing it. Once integrated in the framework of the Logic of Actions and Plans $\mathcal{LAP}$, it gives us a simple and powerful formalism to reasoning about actions and a decision procedure in terms of tableau methods. We also show how to deal with scenarios involving indeterminate and indirect effects which no other causal framework can handle. **Keywords:** Reasoning about actions, causality, dependence relation, context.

1 INTRODUCTION

The reasoning about actions field has gained a very powerful framework to deal with the frame and ramification problems with the logic of actions and plans $\mathcal{LAP}$ [Castilho *et al.*, 1999].

$\mathcal{LAP}$ is a simple multimodal logic where formulae are constructed in the following way: We use an $S4$ operator $\Box$ to represent laws (static or dynamic ones) and a collection of K operators $[\alpha]$, one for each action α, in order to state the behavior of actions.

Given α an action and A, A' classical propositional formulae, the formula $\Box\langle\alpha\rangle\top$ is read as "α is executable". $\Box(A' \rightarrow [\alpha]A)$ as "if A', then after α A". $\Box[\alpha]A$ is an abbreviation for $\Box(\top \rightarrow [\alpha]A)$. For example, in the Yale shooting scenario (YSS) [Hanks and McDermott, 1986], the formula $\Box(\mathit{Loaded} \rightarrow [\mathit{shoot}]\neg\mathit{Alive})$ states that, in every Kripke world, shooting will kill the victim if the gun is loaded. For the same scenario, the formula $\Box(\mathit{Walking} \rightarrow \mathit{Alive})$ is a static law saying that it is always true that someone who is walking must also be alive.

In $\mathcal{LAP}$, every action α has the modal logic K, and the modal operator $\Box$ has logic $S4$. Actions and the $\Box$ operator are linked by an axiom $I(\Box, [\alpha])$ stating that $\Box A \rightarrow [\alpha]A$.

In order to solve the frame problem, in [Castilho *et al.*, 1999] $\mathcal{LAP}$ was augmented with a weak causal connection denoted by a dependence relation $\rightsquigarrow$ between *actions* and *literals*, giving the base logic $\mathcal{LAP}_{\rightsquigarrow}$. In that work, it was also proposed a sound and complete inference engine based on semantic tableau systems [Fitting, 1983], constituting a complete solution to representation and inference tasks. With $\mathcal{LAP}_{\rightsquigarrow}$, we can reason about actions without having to write down all the frame axioms and in a way not subjected to the same counter-examples that have invalidated most of the approaches in the literature along the history.

Nevertheless, even though with that formalism we do not have to write all the domain specific frame axioms, we are obliged to write down *conditional frame axioms* [Castilho *et al.*, 1999]. As an example, the formula below is needed for correctly dealing with the YSS:

$$\Box((\neg\mathit{Loaded} \wedge \mathit{Alive}) \rightarrow [\mathit{shoot}]\mathit{Alive}) \qquad (1)$$

Formulae of the type of (1) establish that if a given condition is true, then some literal persists along the execution of a given action. Without considering such conditional frame axioms, it is not possible to derive the intended conclusions in $\mathcal{LAP}_{\rightsquigarrow}$.

This point has constituted a criticism from the community, since a satisfactory solution to the frame problem

*http://www.inf.ufpr.br/~marcos
†http://www.irit.fr/~Andreas.Herzig
‡http://www.inf.ufpr.br/~ivan

is presumed not to have any kind of such axioms. Instead, it is desired to leave such information about persistence implicit in the domain description, although this is generally done by hiding this information in very complex semantics, whose consequence is the lack of real theorem provers for them.

Other causal approaches have been proposed in the literature in the last decade, in particular Thielscher's influence relation [Thielscher, 1995], Lin's *Caused* predicate [Lin, 1995] and McCain and Turner's causal laws [McCain and Turner, 1995]. However, as pointed out in [Castilho *et al.*, 2000], there are some subtleties that cannot be captured by these approaches without stating non-change information in the domain description.

More recently, Zhang and Foo [2001] have proposed a base logic similar to ours, but they define causality in terms of modal operators, instead of meta-logical information, like we do. In this way, despite the fact that they can deal with ramifications, their approach do not treat the frame problem and fails in deriving implicit qualifications [Ginsberg and Smith, 1988].

In this sense, we propose here a new dependence relation that does not oblige us to write down any kind of non-change statement in the domain description. We do this by adding a parameter to each tuple of the dependence relation with the aim of denoting the *context* (a particular condition) in which an action may influence a given atom. This makes the dependence relation more informative, eliminating the need for stating conditional frame axioms in the description of the domain. Once having done this, we augment the base logic $\mathcal{LAP}$ with the new dependence relation, giving a new approach to the frame and ramification problems.

The resulting logic $\mathcal{LAPD}$ can treat the same class of scenarios for which $\mathcal{LAP}_{\rightsquigarrow}$ does, but it is simpler than the latter in the elaboration of domain descriptions. The advantage of using modal logic appears in the proof procedure, for it is straightforward to adapt the tableau system of $\mathcal{LAP}$ to $\mathcal{LAPD}$.

The present paper is organized as follows: in Section 2, we introduce our new contextual dependence relation. In Section 3, we incorporate this new causal notion into the base logic $\mathcal{LAP}$, obtaining a new dependence-based formalism, and give its semantics. In Section 4, we show why conditional frame axioms are useless within our approach. In Section 5 we define its axiomatics, discuss its computational properties and also sketch our proof procedure. In Section 6, in order to show the use of our framework, we apply it to some scenarios in reasoning about actions. In the final section, we give some conclusions and future work.

2 CONTEXTUAL DEPENDENCE

In this section, we present a new dependence relation that is capable of capturing the *contexts* in which actions are executed.

Basically, we define a ternary dependence relation involving *actions*, *atomic formulae* and *formulae*[1]. The role of the latter formulae is to characterize the particular circumstance in which the actions may influence the truth values of the atomic formulae.

Saying that an atom P *depends* on a certain action α in a given *context* C means that if C is true, then, after the execution of α, P *may* have its truth value changed. We shall say that α may change P in the context C and this concept will be represented by the expression α influences P if C.

In order to better illustrate these ideas, consider the action *shoot*, the atom *Alive* and the formula *Loaded* $\wedge$ *Alive*. As the action *shoot* may change the truth value of *Alive* if *Loaded* and *Alive* are both true, then the expression

shoot influences *Alive* if *Loaded* $\wedge$ *Alive*

must be in our contextual dependence relation.

An important point to be highlighted here is the fact that the notion of dependence so introduced only *permits* change, and does not necessarily *cause* it. This can be better seen in non-deterministic domains, like the Russian turkey scenario [Sandewall, 1994], where, after shooting, the victim might die, and might as well keep on being alive.

We define $ACT = \{\alpha, \beta, \ldots\}$ as the set of *actions*, like *shoot*, *load*, etc. $ATM = \{P, Q, \ldots\}$ is the set of *atomic formulae*, or *atoms*, for short. Examples of atoms are *Loaded* and *Alive*. $LIT = ATM \cup \{\neg P : P \in ATM\}$ is the set of *literals*. If L is a literal, then $|L|$ will be used to denote its associated atom. The set of all formulae of Classical Propositional Logic will be denoted by *PFOR*.

Definition 2.1 A *contextual dependence relation* is a ternary relation $\mathcal{D} \subseteq ACT \times ATM \times PFOR$. ■

The triples (α, P, C) will be written α influences P if C and represent the fact that "α *may* change P in the context C". In other words, this means that "the execution of action α *may* change the truth value of the atom P, as long as the formula C is true".

[1] Formulae of Classical Propositional Logic, without modal operators.

3 A NEW LOGIC OF ACTIONS AND PLANS

In a similar way as done with the binary dependence relation $\rightsquigarrow$, we can combine the logic of actions and plans $\mathcal{LAP}$ with the information represented by the new dependence relation $\mathcal{D}$ defined so far, obtaining the new base formalism $\mathcal{LAPD}$. In this sense, the $\mathcal{LAP}$-models must satisfy the condition that whenever all the contexts in which an action may influence a given atom P are false, then the truth value of P must be preserved along the execution of that action.

As an example, consider the action *shoot* and the atom *Alive* in the YSS. We have that the only way of *shoot* affecting *Alive* is when *Loaded* $\wedge$ *Alive* is true. Thus, in a circumstance in which we have $\neg$*Loaded*, the persistence of *Alive* will be guaranteed by the falsehood of the context *Loaded* $\wedge$ *Alive*.

For the same scenario, considering the action *wait* and the literal *Loaded*, as *wait* does not affect *Loaded*, it does not matter the circumstance, there will not be in $\mathcal{D}$ any expression of the form *wait* influences *Loaded* if C, whatever the context C is. In this case, we guarantee the persistence of *Loaded* through the execution of *wait*.

With this dependence-based condition, one eliminates the $\mathcal{LAP}$-models in which non-intuitive changes occur in the following way: suppose that we are in a particular situation (a possible world in our Kripke semantics) w in which the atom P is false. First, imagine that the only element of $\mathcal{D}$ involving both α and P is α influences P if C. Thus, as long as C is true, the execution of α may change, or not, the truth value of P, since our causal notion only allows change, not forcing it. But surely α will not change the value of P if C is false. Suppose now that there is no $C \in PFOR$ such that α influences P if $C \in \mathcal{D}$. Then, the execution of α may never make P true, and hence P will still be false after α.

Definition 3.1 Let $\mathcal{D}$ be a ternary dependence relation. A model for $\mathcal{LAPD}$ is a $\mathcal{LAP}$-model $\mu = \langle W, \{R_\alpha : \alpha \in ACT\}, R_\Box, \tau\rangle$, such that whenever $wR_\alpha w'$ then for every α and for every $P \in ATM$, if, for all $C \in PFOR$ such that α influences P if C, $w \not\models C$, then $w \in \tau(P)$ if and only if $w' \in \tau(P)$. ■

Given a contextual dependence relation $\mathcal{D}$, we say that a formula A is true in a $\mathcal{LAPD}$-model $\mu = \langle W, \{R_\alpha : \alpha \in ACT\}, R_\Box, \tau\rangle$ if $w \models A$ for every $w \in W$. A is $\mathcal{LAPD}$-valid (noted $\models_{\mathcal{LAPD}} A$) if A is true in all $\mathcal{LAPD}$-models.

4 INFERRING CONDITIONAL FRAME AXIOMS

Once defined our contextual dependence relation, we immediately have an alternative to conditional frame axioms. This will be shown in this section. Due to space limitations, all the proofs of the results were ommited. For more details, refer to [Varzinczak, 2002].

Definition 4.1 Let $\alpha \in ACT$, $P \in ATM$ and $\mathcal{D}$ a contextual dependence relation. We define

$$Pre_{\mathcal{D}}(\alpha, P) = \bigvee_{(\alpha \text{ influences } P \text{ if } C) \in \mathcal{D}} C$$

■

In other words, $Pre_{\mathcal{D}}(\alpha, P)$ is the disjunction of all contexts in which α may affect the truth value of atom P, given a dependence relation $\mathcal{D}$.

Theorem 4.1 $\Box((\neg Pre_{\mathcal{D}}(\alpha, |L|) \wedge L) \rightarrow [\alpha]L)$ is $\mathcal{LAPD}$-valid. ■

With this result, we can see that in a domain description using a dependence relation $\mathcal{D}$ there is no need for a set of conditional frame axioms, since all the conclusions that are obtained with the aid of the latter can also be inferred with the former.

As an example, consider the action *shoot*, the atom *Alive* and the literal $\neg$*Loaded*, and suppose

$$\mathcal{D} = \{\textit{shoot} \text{ influences } \textit{Alive} \text{ if } \textit{Loaded} \wedge \textit{Alive}\}$$

Then, we have that the conditional frame axiom (1) is $\mathcal{LAPD}$-valid. In other words, the persistence of *Alive* follows from the dependence information in $\mathcal{D}$, making completely unnecessary, thus, the statement of the conditional frame axiom (1).

5 PROOF THEORY

5.1 AXIOMATICS AND COMPLEXITY

Given a ternary dependence relation $\mathcal{D}$, we axiomatize the class of $\mathcal{LAPD}$-models in the same way as done for $\mathcal{LAP}$ in [Castilho *et al.*, 1999, Section 4.2], adding an axiom scheme founded on the dependence relation:

- *Persist*$([\alpha])$: $\neg Pre_{\mathcal{D}}(\alpha, |L|) \wedge L \rightarrow [\alpha]L$, if α influences $|L|$ if C is in $\mathcal{D}$, $C \in PFOR$.

In order to show the soundness and completeness of $\mathcal{LAPD}$ with respect to its semantics, we have the following result:

Theorem 5.1 For all contextual dependence relation $\mathcal{D}$, the axiomatics of $\mathcal{LAPD}$ is sound and complete with respect to the class of $\mathcal{LAPD}$-models. ■

The theorem below shows that the complexity of $\mathcal{LAPD}$ is the same as that of $\mathcal{LAP}$.

Theorem 5.2 $\mathcal{LAPD}$ is decidable, and the satisfiability problem in $\mathcal{LAPD}$ is EXPTIME-complete. ■

This theorem guarantees that the inclusion of the ternary dependence relation $\mathcal{D}$ in $\mathcal{LAP}$ does not increase the computational complexity of the base logic.

5.2 PROOF PROCEDURE

In this section, we briefly sketch the tableau-based proof method for $\mathcal{LAPD}$. Here, we can assume w.l.o.g. that the context C is a set of literals. In essence, we replace the rules (SF) and (SB), which have to do with the binary dependence, in the method proposed in [Castilho *et al.*, 1999] by the following tableau rules in order to capture the semantical aspect of the dependence-based condition stated in Section 3 ($w :: L$ means "the literal L is true at world w", and $w \xrightarrow{\alpha} w'$ reads "w' is accessible from w by an execution of α"):

$$(RP)\ \frac{w :: L,\ w \xrightarrow{\alpha} w' \text{ and } C \not\subseteq w \quad \text{for all } \alpha \text{ influences } |L| \text{ if } C \in \mathcal{D}}{w' :: L}$$

$$(RB)\ \frac{w' :: L,\ w \xrightarrow{\alpha} w' \text{ and } C \not\subseteq w \quad \text{for all } \alpha \text{ influences } |L| \text{ if } C \in \mathcal{D}}{w :: L}$$

We call (RP) the *rule of persistence* and it captures the intuitive meaning of the semantics of dependence: all literals in a node of a tableau that depend on the execution of a given action α in a context that does not verify must be propagated to the node following the execution of that action.

Rule (RB) is the *rule of back-propagation* and expresses that if a given literal L is true in a node but it was not caused to change its truth value, then L must also be true in the antecedent node. This kind of rule is needed only to guarantee the completeness of the method, as shown in [Castilho *et al.*, 1999].

In order to get soundness, we have to define a particular strategy where rules (RP) and (RB) are applied at the end. Thus, the tableau can be proved sound and complete [Varzinczak, 2002]. Following the work in [Heuerding *et al.*, 1996], we can also show that it is a decision procedure. Moreover, its implementation is straightforward in Lotrec System [Fariñas del Cerro *et al.*, 2001].

6 SOME SCENARIOS IN $\mathcal{LAPD}$

In this section, we use $\mathcal{LAPD}$ as the base formalism for modeling and inferring in some typical scenarios. It is worthy noting that in none of the scenario descriptions that follow there is the need for stating conditional frame axioms, since their semantical aspects have already been implicitly captured by means of the contextual dependence relation $\mathcal{D}$.

In all scenario descriptions below, KB represents the set of observations and LAW that of static, effect and executability laws in $\mathcal{LAPD}$.

Example 6.1
(Forcing a door [Castilho *et al.*, 1999]) Consider a door that can be closed or barricaded (by a cupboard, for example), and two actions: *open*, that opens a door if it is not barricaded, and *force*, that is stronger and unconditionally opens it. In $\mathcal{LAPD}$ it is as follows:

$$LAW = \left\{ \begin{array}{l} \Box(\mathit{Barricaded} \rightarrow \mathit{Closed}), \\ \Box[\mathit{open}]\neg \mathit{Closed}, \\ \Box[\mathit{force}]\neg \mathit{Closed} \end{array} \right\}$$

$$\mathcal{D} = \left\{ \begin{array}{c} \mathit{open} \text{ influences } \mathit{Closed} \text{ if } \mathit{Closed}, \\ \mathit{force} \text{ influences } \mathit{Closed} \text{ if } \mathit{Closed}, \\ \mathit{force} \text{ influences } \mathit{Barricaded} \text{ if } \mathit{Barricaded} \end{array} \right\}$$

$$KB = \{\mathit{Closed}, \mathit{Barricaded}\}$$

With this modeling, we can conclude that the formula

$$(KB \wedge LAW) \rightarrow [\mathit{force}](\neg \mathit{Closed} \wedge \neg \mathit{Barricaded})$$

is $\mathcal{LAPD}$-valid, which means that in $\mathcal{LAPD}$, like in $\mathcal{LAP}_{\rightsquigarrow}$, it is possible to derive the indirect effect of action *force*. Notice, however, that as with $\mathcal{LAP}_{\rightsquigarrow}$, we also have to write down the indirect dependence between *force* and *Barricaded*. Nevertheless, we argue that all existing approaches that do not do things this way fail in solving this example.

Another important conclusion one can obtain with our approach is the implicit qualification [Ginsberg and Smith, 1988] for action *open*, that is, such action cannot be executed if the door is barricaded:

$$\models_{\mathcal{LAPD}} (KB \wedge LAW) \rightarrow [\mathit{open}]\bot$$ ■

Remark 6.1 We consider that for a given effect law $\Box(A \rightarrow [\alpha]L)$ it is reasonable to have α influences $|L|$ if $A \wedge \neg L$ in $\mathcal{D}$, otherwise it would seem a bit strange. So, we assume that every effect law has a corresponding element in the dependence relation.

Example 6.2
(Toulouse suitcase [Castilho *et al.*, 1999]) Consider an extension of Lin's suitcase [Lin, 1995] where we want to express that if both of its locks are linked (by a rigid metal bar, for example), then they are always in the same position. The representation of this situation in $\mathcal{LAPD}$ is given below ($i = 1, 2$):

$$LAW = \left\{ \begin{array}{l} \Box\langle toggle_1\rangle\top, \\ \Box(Linked \rightarrow (Up_1 \leftrightarrow Up_2)), \\ \Box((Up_1 \wedge Up_2) \rightarrow Opened), \\ \Box(\neg Up_i \rightarrow [toggle_i]Up_i, \\ \Box(Up_i \rightarrow [toggle_i]\neg Up_i \end{array} \right\}$$

$$\mathcal{D} = \left\{ \begin{array}{c} toggle_i \text{ influences } Up_i \text{ if } \top, \\ toggle_1 \text{ influences } Up_2 \text{ if } Linked, \\ toggle_2 \text{ influences } Up_1 \text{ if } Linked, \\ toggle_1 \text{ influences } Opened \text{ if } Up_2 \wedge \neg Opened, \\ toggle_2 \text{ influences } Opened \text{ if } Up_1 \wedge \neg Opened \end{array} \right\}$$

This example is problematic for causal approaches that are not action indexed. ■

More generally, we claim that all the existing approaches fail to handle scenarios having actions with both indeterminate and indirect effects, as exemplified in the following scenario:

Example 6.3
(The Mailboxes scenario) Suppose $Mbox_1$ means "the message is in mailbox 1", and $Mbox_2$ "the message is in mailbox 2", and consider the actions $save_1$ and $save_2$, that always save an email message in $Mbox_1$ and in $Mbox_2$, respectively. Suppose too we have a non-deterministic *save* action that saves the email in one of the two mailboxes or in both. We represent the fact that the email is saved in $Mbox_1$ or in $Mbox_2$ or in both by the atom *Saved*. The representation of this scenario in $\mathcal{LAPD}$ is as follows ($i = 1, 2$):

$$LAW = \left\{ \begin{array}{l} \Box\langle save\rangle\top, \Box\langle save_i\rangle\top, \\ \Box(Saved \leftrightarrow Mbox_1 \vee Mbox_2), \\ \Box[save]Saved, \\ \Box[save_i]Mbox_i \end{array} \right\}$$

$$\mathcal{D} = \left\{ \begin{array}{c} save \text{ influences } Saved \text{ if } \neg Saved, \\ save \text{ influences } Mbox_i \text{ if } \neg Mbox_i, \\ save_i \text{ influences } Mbox_i \text{ if } \neg Mbox_i, \\ save_i \text{ influences } Saved \text{ if } \neg Saved \end{array} \right\}$$

$$KB = \{\neg Saved, \neg Mbox_1, \neg Mbox_2\}$$

From this representation, we can conclude

$$\models_{\mathcal{LAPD}}(KB \wedge LAW) \rightarrow [save](Mbox_1 \vee Mbox_2)$$

It seems to us that this example is problematic for all the existing proposals (i.e. approaches allowing the representation of actions with indirect and indeterminate effects). We think that the problem is the following: if we have indirect nondeterministic effects, then

1. we must exempt these effects from minimization of change (in order to avoid exclusive interpretation of inclusive disjunctions, cf. Reiter's "dropping a coin on a chess-board" example);
2. as the effect is indirect, this exemption must be specified indirectly: mentioning the context where it applies, but without mentioning the action.

Then it is difficult to exclude that another action applies in exactly the same context, but without the indirection. As an example, in Lin's approach [Lin, 1996], in order to obtain the intended conclusions, we are obliged to state the formula

$$Poss(save_1, s) \rightarrow Caused(Mbox_2, false, do(save_1, s)),$$

which is unintuitive. ■

7 CONCLUSIONS AND FURTHER WORK

In this work, we have presented a new causal dependence that combined with the logic of actions and plans $\mathcal{LAP}$ results in a powerful framework to reasoning about actions.

In essence, our method is a modification of that in [Castilho *et al.*, 1999], which defines a binary relation between actions and literals as a way of expressing causal information. In our approach, however, we consider a ternary one, since we argue that with it we can get a more intuitive domain description. In this sense, we have seen that our definition of a contextual dependence allows a more economic representation of the domain under consideration, for the presence of contexts establishes a more informative causal notion, eliminating the need for stating conditional frame axioms in the set of action laws.

We have also briefly sketched our tableau method, that constitutes a decision procedure. This gives to our formalism a higher degree of practical applicability and makes it a more complete solution compared to other approaches in the literature.

We have made an analysis of typical scenarios constituting good instances of the frame and ramification problems. In our approach we can treat them in a more economic way than using $\mathcal{LAP}_{\rightsquigarrow}$, without increasing the complexity of the solution. In fact, $\mathcal{LAP}_{\rightsquigarrow}$ deals

with the same class of problems than $\mathcal{LAPD}$, but at the price of writing conditional frame axioms.

We have also investigated the behavior of a ternary dependence in scenarios involving actions with indirect effects and our solution permits us to obtain the desired conclusions. In this sense, we believe our approach unifies a more robust solution to the frame and ramification problems with a higher degree of representational parsimony [Shanahan, 1997].

In the forcing a door scenario, we were able to capture implicit qualifications. However, in the same way as with $\mathcal{LAP}_{\rightsquigarrow}$, we have to state indirect dependences in $\mathcal{D}$. An argument in response to possible criticisms is the fact that there is no approach capable of solving the forcing a door scenario without explicitly stating in one way or another the relation between the action and the indirect effects — which is what these approaches would like to avoid.

With respect to situations involving non-determinism, our method applies too, as shown in the mailboxes scenario. In this case, it suffices to state in $\mathcal{D}$ the correct dependences relating an action and each atom it can affect. In the same way, as long as we concern, there is no approach in the literature capable of solving this example in a better way.

We plan to pursue our work by extending our solution by an account of sensing and communication actions. This will be done by integrating the approach in [Herzig *et al.*, 2000a, Herzig *et al.*, 2000b], the long-term aim being a unified logic of actions and beliefs.

References

[Castilho *et al.*, 1999] M. A. Castilho, O. Gasquet, and A. Herzig. Formalizing action and change in modal logic I: the frame problem. *Journal of Logic and Computation*, 9(5):701–735, 1999.

[Castilho *et al.*, 2000] M. A. Castilho, A. Herzig, and C. Schwind. Raisonnement sur les actions: les approches basées sur la causalité et la dépendance. In *Nouveaux Défis en Sciences de l'Information: Documents et Evolution*, Toulouse, France, Sep 2000.

[Fariñas del Cerro *et al.*, 2001] L. Fariñas del Cerro, D. Fauthoux, O. Gasquet, A. Herzig, D. Longin, and F. Massacci. Lotrec: the generic tableau prover for modal and description logics. In *Proc. IJCAR*, LNCS, page 6. Springer Verlag, Jun 2001.

[Fitting, 1983] M. Fitting. *Proof Methods for Modal and Intuitionistic Logics*. D. Reidel, 1983.

[Ginsberg and Smith, 1988] M. L. Ginsberg and D. E. Smith. Reasoning about actions II: The qualification problem. *Art. Intelligence*, 35(3):311–342, 1988.

[Hanks and McDermott, 1986] S. Hanks and D. McDermott. Default reasoning, nonmonotonic logics and the frame problem. In *AAAI'86*, pages 328–333, 1986.

[Herzig *et al.*, 2000a] A. Herzig, J. Lang, D. Longin, and T. Polacsek. A logic for planning under partial observability. In *AAAI'2000*, Aug 2000.

[Herzig *et al.*, 2000b] A. Herzig, J. Lang, and T. Polacsek. A modal logic for epistemic tests. In *Proc. ECAI'2000*, Berlin, August 2000.

[Heuerding *et al.*, 1996] A. Heuerding, M. Seyfried, and H. Zimmermann. Efficient loop-check for backward proof search in some non-classical propositional logics. *Tableaux 96*, 1(1071):210–225, 1996.

[Lin, 1995] Fangzhen Lin. Embracing causality in specifying the indirect effects of actions. In *Proc. IJCAI'95*, pages 1985–1991, 1995.

[Lin, 1996] Fangzhen Lin. Embracing causality in specifying the indeterminate effects of actions. In *AAAI'96*, pages 670–676, 1996.

[McCain and Turner, 1995] N. McCain and H. Turner. A causal theory of ramifications and qualifications. In *Proc. IJCAI'95*, pages 1978–1984, 1995.

[Sandewall, 1994] E. Sandewall. *Features and Fluents: the Representation of Knowledge about Dynamical Systems*, volume 1. Oxford University Press, 1994.

[Shanahan, 1997] M. Shanahan. *Solving the Frame Problem — A Mathematical Investigation of the Common Sense Law of Inertia*. The MIT Press, Cambridge, Massachusetts, USA, 1997.

[Thielscher, 1995] M. Thielscher. Computing ramifications by postprocessing. In *Proc. IJCAI'95*, pages 1994–2000, 1995.

[Varzinczak, 2002] I. J. Varzinczak. Causalidade e dependência em raciocínio sobre ações. Master's thesis, Depto. de Informática, UFPR, 2002.

[Zhang and Foo, 2001] D. Zhang and N. Y. Foo. EPDL: A logic for causal reasoning. In *Proc. IJCAI'2001*, pages 131–138, 2001.

Learning non-monotonic causal theories from narratives of actions

David Lorenzo
Computer Science Dept.
Univ. Coruña
15071 A Coruña (Spain)

Abstract

Non-monotonic formalisms for reasoning about actions and change have become a whole subfield of Artificial Intelligence. Current implementations allow to work with very expressive action theories involving ramifications, concurrent actions, complex qualifications and so on. A natural question that can be posed is whether this kind of declarative knowledge can be learned from observed time traces of property values from an existing dynamic system. For this task we consider a *narrative*-based logical theory of change in the form of Extended Logic Programs where logic-based learning methods can be applied effectively. The use of a narrative formalism provides more expressivity on the theories that are learned, for instance, to learn the effects of *concurrent* actions.

1 Introduction

Non-monotonic formalisms for reasoning about actions and change have become a whole subfield of Artificial Intelligence. These formalisms serve as an unifying approach that subsumes the many special purpose mechanisms that have been developed in other approaches, and that accommodates all the features of dynamical systems [24]. In these formalisms, system's behaviors are naturally viewed as appropriate logical consequences of the domain's description, hence the specification of actions and their effects is made intuitive and natural. Current implementations like VITAL [7], CCALC [17], PAL [4], among others, allow to work with very expressive action theories involving ramifications, concurrent actions, complex qualifications and so on.

A natural question that can be posed is whether this kind of declarative knowledge can be learned from observed time traces of property values from an existing dynamic system. Any attempt to learn such theories must cope with various facets such as inertia (and the associated Frame Problem), constraints and indirect effects (and the associated Ramification problem), non-deterministic and other complex effects of actions. Machine Learning methods have been used to learn planning operators [20, 5, 27], however, in most cases the inferred model corresponds to a set of STRIPS-like operators [8]. Some extensions have been considered since then, that are able to cope with more complex effects [3], however, they rely on special purpose formalisms, which makes the results of learning very dependent on the particular formalism and difficult to transfer.

Our work differs from previous approaches also in the use of Inductive Logic Programming [23] methods which allow a natural integration with implementations of Action Theories based on Logic Programming. Unlike classical inductive learning, ILP uses Logic Programming as the representational mechanism for hypotheses and observations. We adopt Extended Logic Programs as the form of programs to be learned, where two kinds of negation–negation as failure and classical negation–are effectively used in the presence of incomplete information.

This paper is organized as follows. Section 2 briefly describes logic-based formalisms for reasoning about actions and how these can be learned by adapting current *ILP* methods for static domains. In section 3 we consider the use of ramification constraints to learn the indirect effects of actions. In section 4, we describe a prototype based on conventional ILP methods and some experimental results. In section 5 we extend the framework to learn the effects of concurrent actions. Finally, in section 6 we present some conclusions and outlook future work.

2 Learning Action Theories from narratives

A fundamental problem in machine learning is the representation language used for the learning process. This is specially important when dealing with temporal data, where most of the previous methods treat temporal data as an unordered collection of events, ignoring its temporal information or need extensive transformations. In this paper, we consider the use of Action Languages [10] as the representational method underlying the learning process, which benefits from the declarative nature and the inferential capacity of these logic-based formalisms. The most classical formalisms for reasoning about actions and change are the Situation Calculus [18] and the Event Calculus [13]. In both cases, logic Programs can be used to represent the effects of actions by importing the corresponding ontology, where effect axioms are logic programs with a fixed clausal structure. In [16] we used an implementation of the Situation Calculus similar to [9], because of its simpler formalization with respect to the Event Calculus. However, Situation Calculus has several problems that limit its applicability [11], hence we have used a narrative-based version with a predicate *happens/2* to mean for the execution of actions, apart from the predicate *holds/2* that is used for recording the values of fluents at each situation ($s \geq 0$). This Logic Programming implementation uses an special predicate *caused/3* [14] (or *abnormal/3* [2]) and negation as failure to implement inertia. Formally we have that an action theory is a conjunction of:

- A finite set of general clauses $[\neg]Holds(f, 0)$ where 0 denotes the initial situation.
- A finite set of clauses in the form

$$Caused(f, v, t) \leftarrow Happens(a, t), \pi \quad (1)$$

  where π consists of literals in the form $[\neg]Holds(f', t\text{-}1)$. The description states that, in any situation, if the precondition holds then the effect will hold in the resulting situation. These axioms are called *effect axioms* or *action laws*.
- The universal frame axiom describes how the world stays the same (as opposed to how it changes).

$$Holds(f, t) \leftarrow Happens(a, t), Holds(f, t\text{-}1), \text{not } Caused(f, v, t) \quad (2)$$

$$\neg Holds(f, t) \leftarrow Happens(a, t), \neg Holds(f, t\text{-}1), \text{not } Caused(f, v, t) \quad (3)$$

- Some clauses that propagate caused values to *Holds*.

$$Holds(f, t) \leftarrow Caused(f, \text{true}, t) \quad (4)$$

$$\neg Holds(f, t) \leftarrow Caused(f, \text{false}, t) \quad (5)$$

Inertia is tackled in the following way: a fluent may only change if a cause for the change can be derived from the theory. Thus, we can add as many effect axioms as we like and rely upon axioms (2) and (3) to implement inertia. Other formalizations define a predicate *Pos/2* apart from *holds/2* and *caused/3*, such that for any action a and any situation s, $Poss(a, s)$ is true if a is executable in s. Thus, *caused/3* represents the *conditional effects* and *Poss* would represent the *executability* of the action. A similar shape for the programs to be learned based on the Event Calculus could also be used.

Given the description of an initial state of the world, and a set of observations of a dynamic system, the learning task we consider is to find a theory that correctly predicts the state of the world after the execution of any sequence of actions starting from any initial state. Let us consider the blocks world. Observations are in the form of *narratives* of actions (Fig. 1).

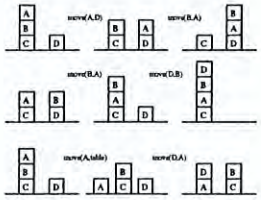


Figure 1: Narratives in the Blocks world

Each narrative consists of the values of fluents at the initial situation represented with the predicate *holds/2*, the actions executed and the effects produced by the actions, represented with the predicate *happens/2* and *caused/3* respectively. For instance, the first narrative of Fig. 1 is represented as follows:

holds($clear(a), 0$)	happens($move(a, d), 1$)	happens($move(b, a), 2$)
holds($clear(d), 0$)	caused($on(a, b), f, 1$)	caused($on(b, c), f, 2$)
holds($on(a, b), 0$)	caused($on(a, d), t, 1$)	caused($on(b, a), t, 2$)
holds($on(b, c), 0$)	caused($clear(b), t, 1$)	caused($clear(c), t, 2$)
holds($on(c, table), 0$)	caused($clear(d), f, 1$)	caused($clear(a), f, 2$)
...		

so that *caused/3* represents the truth value and the causality that were represented separately by *holds/2* and *affects/3* in [16]. Each narrative may correspond to a different scenario or a different initial situation, which makes that examples covering a wide range of situations can be used for learning. In the Situation Calculus, multiple narratives of events are represented in a single model, however, they are now considered for learning as separate models, one for each narrative. This fits in the "learning from interpretations" paradigm of ILP where examples consist of multiple interpretations and not just one interpretation as it is usual in ILP. In our case, an interpretation corresponds to a narrative.

These observations can be provided directly to a ILP algorithm, however, the learner needs a full description

of each narrative, hence if a fluent does not change after executing an action, e.g., $holds(on(b,c),1)$, its truth value must be explicitly asserted in the input data. In previous approaches, both kinds of observations had to be provided as ground facts so that non-affected values of fluents must be part of the information given to the learning algorithm, thus having to provide the learner with large datasets for each narrative from which only a small fraction corresponds to effects of actions. In some cases, explicit *frame axioms* are obtained as part of the learning results [25]. This can be avoided if we include axioms (2 and 3) in the input data, so that observations need only be explicitly given for the effects of actions, whereas the inertia axiom propagates non-affected truth values from one situation to the next one, completing every situation. This is possible because of the explicit separation between the predicates *caused/3* and *holds/2*.

With respect to negative examples for each fluent, it is assumed that an action is not successful in the situations where the fluent did not change. For this reason, we build negative examples for the predicate *caused/3* for those fluents not affected by actions. By doing so, applying the induced rules on the observations will not result in postulating new effects not considered initially in the narratives used for learning. This CWA makes an inductive leap of a different kind, in the sense that it postulates that the induced rules will be valid on unseen narratives. However, if narratives are partially specified, additional methods are needed for learning to succeed [15].

Once narratives are encoded as shown above, conventional ILP methods can be applied to learn a definition for the positive and negative values of fluents. Positive examples E^+ consist of ground facts $caused(f,v,t)$ that correspond to the effects of actions, both for the positive value and for the negative value. Negative examples E^- consist of ground facts $caused(f,v,t)$ representing observations where a fluent f was not affected by any action. Apart from positive and negative examples, ILP uses so-called *Background Knowledge* (B), consisting basically of predicate definitions, and builds a definition of a target predicate in terms of itself and the background predicates. In our case, background knowledge includes a set of ground facts $happens(a,t)$ for every situation in a narrative, representing that an action a was executed at time t, *holds/2* ground facts for fluents at the initial situation of each narrative, the universal *inertia axiom* (eq. 2,3), axioms (4 and 5), and domain-dependent static predicates common to all narratives. The objective is to find a theory H composed of effect axioms in the form (1), such that:

$$(\forall e^+ \in E^+) \quad B \cup H \models e^+ \tag{6}$$

$$(\forall e^- \in E^-) \quad B \cup H \not\models e^- \tag{7}$$

In order to satisfy the completeness requirement, the learned rules for the positive and the negative concept will entail all positive examples (eq. 6). The consistency requirement is satisfied when the induced rules do not entail any of the negative examples (eq. 7).

3 Learning ramifications of actions

Previous approaches to learning action models are restricted to predicting a single outcome or effect of an action. This forces the explicit representation of all the effects of an action as direct effects, producing the so-called *Ramification problem*. In some cases, it is difficult to express some action preconditions explicitly as a condition on the starting state. This makes effect axioms increasingly complex and consequently harder to learn, since they need to anticipate the *ramifications* of the executed action. Most solutions that have been proposed for the ramification problem require effect axioms to specify the most significant effects of an action and rely on *ramification constraints*, i.e., general laws describing dependences between fluents, for specifying additional changes that are due to the action. A ramification constraint is a formula

$$caused(f,v,t) \leftarrow \pi \tag{8}$$

where the $Holds$ literals in π are only of the form $[\neg]Holds(f',t)$. According to this, indirect effects are derived from the state of fluents, while direct effects come from the execution of actions.

The learner must now infer how properties of a domain are (directly/indirectly) affected by the execution of actions. In general, actions can have multiple direct and indirect effects, and fluents can be a direct (resp. indirect) effect of multiple actions. In practice, we allow the learner to determine at each step whether a fluent should be learned as a direct or an indirect effect, thus learning possibly a mix of axioms based on a compression-based measure so that smaller theories are preferred. The insight is that indirect effects can make programs sensibly shorter, given that a ramification constraint may subsume the direct effects of several actions, which have a positive influence in their learnability, as the difficulty of learning a logic program is very much related to its length.

Indirect effects actually represent a *propagation of changes*, hence ramification constraints are allowed to cover only those examples where the action executed produced several simultaneous effects, and at least one of the fluents in the body must have changed simultaneously with the fluent in the head. Note also that the predicate *caused/3* cannot be substituted by

holds/2 in (8) because a fluent that has been initiated/terminated directly through an effect axiom cannot be terminated/initiated indirectly through such a ramification constraint, unless it is released from inertia beforehand, otherwise it would lead to contradiction. However, a number of "useless" (even if correct) instances of *caused/3* are derived even for those situations where a fluent does not change. This contradicts the observations given that negative examples for a fluent are given for those situations where the fluent does not change. To avoid this, the consistency of ramification constraints is not tested on those negative examples where a fluent does not change its value.

4 Learning with conventional ILP methods

We have developed a prototype *LRAC* consisting of an "ordinary" ILP algorithm. We do not assume any particular learning algorithm, however, in the prototype we used the notion of Inverting Entailment (IE) that is the technique used in the system Progol [22]. The IE method computes the most specific generalization (also called bottom clause) from a positive example with respect to the background knowledge. For instance, the most specific generalization of $caused(on(c, table), t, 4)$ is[1]:

```
caused(on(A,B),t,C) :- happens(move(A,B),C), diff(A,B), diff(B,A),
       -holds(on(A,B),C-1), -holds(on(B,A),C-1), holds(on(A,E),C-1),
       holds(on(B,F),C-1), holds(clear(A),C-1), holds(clear(B),C-1),
       -holds(on(A,F),C-1), -holds(on(B,E),C-1), -holds(on(F,A),C-1),
       -holds(on(F,B),C-1), -holds(clear(F),C-1), holds(on(F,E),C-1),
       diff(A,E), diff(A,F), diff(B,E),diff(B,F), diff(E,A), diff(E,B),
       diff(E,F), diff(F,A),diff(F,B), diff(F,E), table(E).
```

Effect axioms introduce a bias for the clauses to be learned, where *holds/2* literals in the body can only refer to the previous situation. In general, the bottom clause can have infinite cardinality, however some biases are considered to keep it manageable, e.g., type information for fluent and action literals, bounds on the length of hypotheses and the depth of variables, and other syntactic restrictions. *LRAC* searches in a top-down fashion to find a subset of literals of the bottom clause that explains as many of the positive examples as possible, so that the search space is bounded by this most specific clause and the most general clause (with an empty body). Then the explained examples are separated and the algorithm recursively conquers the remaining positive examples.

The rule generation phase will be called twice for each fluent, once for the positive concept and once for the negative concept, and will be repeated searching for effect axioms and for ramification constraints. For testing the coverage of hypotheses, these are evaluated by performing a derivation of each example from a program composed by the hypothesis, the background theory and the clauses previously learned, so that the coverage test corresponds to a *temporal projection* problem for each hypothesis generated by *LRAC*, which is a high overload for medium-sized domains. However, the common approach in ILP is to use θ-subsumption which is computationally more efficient so that a clause covers *extensionally* an example e, i.e., there exists a ground instance of the clause $e \leftarrow l_1, \ldots, l_n$ where each l_i belongs to $B \cup \{e\}$. However, θ-subsumption needs some assumptions to guarantee completeness and soundness [15]. For instance, care must be taken to ensure that the addition of ramification constraints to a theory does not cause a *non-finite recursion*, where fluents may use one another in their definitions. If a cycle is identified, the last clause is removed and remembered, so that in the case in which it is generated again in the specialization loop, it is immediately discarded. This is done in order to avoid that the system goes into a loop of continuously generating and retracting the same clause.

For the experimental part we have considered several challenge problems in the literature of reasoning about actions, as well as other domains used in AIPS planning competitions [19]. The objective is to show not only if a sufficiently general theory can be learned from observations but also if the learned theory for each domain matches the description found in the literature. Let us consider a prototypical problem of the literature [26], that consists of a circuit that includes a lamp, a relay, and three switches sw_1, sw_2 and sw_3, together with actions in the form t_i $(i = 1 \ldots 3)$.

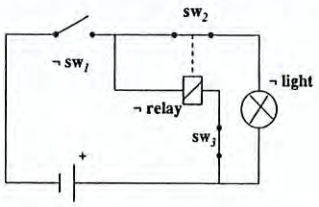


Figure 2: An electric circuit

We used a simple *wander* program that collects data about its actions and the fluents that changed while exploring the environment. Input data consists of 21 narratives from which *LRAC* found:

```
caused(sw1,t,A) :- happens(t1,A),-holds(sw1,A-1).
caused(sw1,f,A) :- happens(t1,A),holds(sw1,A-1).
caused(sw2,t,A) :- happens(t2,A),-holds(sw2,A-1),
                   -holds(relay,A-1).
caused(sw2,f,A) :- happens(t2,A),holds(sw2,A-1).
```

[1] Background for the blocks world consists of domain predicates *on*/2, *clear*/1, *block*/1, *table*/1, and a predicate $diff/2$ to represent $X \neq Y$.

```
caused(sw2,f,A) :- holds(relay,A).
caused(sw3,f,A) :- happens(t3,A),holds(sw3,A).
caused(sw3,t,A) :- happens(t3,A),-holds(sw3,A).
caused(light,t,A) :- holds(sw1,A), holds(sw2,A).
caused(light,f,A) :- -holds(sw1,A).
caused(light,f,A) :- -holds(sw2,A).
caused(relay,t,A) :- holds(sw1,A), holds(sw3,A).
caused(relay,f,A) :- -holds(sw1,A).
caused(relay,f,A) :- -holds(sw3,A).
```

According to the rules, action t_1 (resp. t_3) toggles switch sw_1 (resp. sw_3). The *relay* is controlled by switches sw_1 and sw_3, i.e., the relay is active whenever both sw_1 and sw_3 hold simultaneously, so that only ramification constraints are learned for the relay instead of the corresponding effect axioms for all the actions that affect it, i.e., t_1 and t_3, thus producing a shorter theory. Similarly for *light* with sw_1 and sw_2. With respect to sw_2, the negative value is in some cases a direct effect of t_2 so that sw_2 can be closed only when the relay was previously inactive, and in other cases an indirect effect of the actions that activate the relay. The use of the ramification constraint for the negative value also produces a shorter theory, otherwise *LRAC* would need to learn an effect axiom for the actions that activate the relay.

In the blocks world, from 48 narratives of length 6, *LRAC* learned the following clauses:

```
caused(on(A,B),t,C) :- happens(move(A,B),C),
                       holds(clear(A),C-1),holds(clear(B),C-1).
caused(on(A,B),t,C) :- happens(move(A,B),C),
                       holds(clear(A),C-1),table(B).
caused(on(A,B),f,C) :- holds(on(A,D),C),diff(B,D).
caused(clear(A),t,B) :- happens(move(C,D),B),
                        holds(clear(D),B-1),holds(clear(C),B-1),
                        holds(on(C,A),B-1),diff(D,A).
caused(clear(A),t,B) :- happens(move(C,D),B),holds(clear(C),B-1),
                        holds(on(C,A),B-1),table(D).
caused(clear(B),f,A) :- holds(on(_,B),A).
```

LRAC learned two effect axioms for the positive value of $on/2$ that deal separately with the cases where a block is moved onto other block or onto the table [2]. The use of ramification constraints produces a shorter theory for the negative value of $on/2$, even with a single action $move/2$, and it represents that $on/2$ is a *functional fluent*. The ramification constraint for the negative value of $clear/1$ does not produce a clear benefit mainly because there are no other actions that affect it apart from $move/2$. However, as the ramification constraint is shorter than the corresponding effect axiom, *LRAC* is biased to prefer it. Actually the positive value of $clear/2$ is also an indirect effect, given that a block is clear if there is no another block on it, however, *LRAC* learned two effect axioms for the $move/2$ action due to the limited quantification of logic programs.

[2]The table is an special location that is always clear.

5 Concurrent actions

We have also assumed that actions are atomic. If *concurrent actions* are allowed, then an effect may depend on a particular combination of actions, an action may qualify another action's effects, effects can be cancelled, and so on [1]. As a consequence, learning about concurrent actions even in simple domains, is a complex task because the theory to be learned must cope with every possible interaction between actions. In this case, negative examples correspond to those situations where an effect is not caused because there are missing preconditions, missing actions or actions that are executed additionally.

Concurrent actions are introduced by allowing multiple instances of the predicate *happens/2* in the body of rules to be learned. Default negation is used to cope with the non-execution of conflicting actions. We explicitly use positive and negative *happens/2* literals so that at least a positive *happens/2* literal must be added to the body, whereas the negative counterpart of *happens/2* will include any potentially cancelling action, which are taken from those actions executed in the negative examples. By doing so, *LRAC* can specialize an overgeneral effect axiom by adding positive and negative *holds/2* and *happens/2* literals. Effects of concurrent actions can be also seen in terms of inheritance [1], i.e., compound actions normally inherit effects from their subactions, and cancelling actions cancel inheritance of the effects of atomic actions. However, we use positive and negative *happens/2* literals instead of cancelling and inheritance axioms, given that the latter express separately the positive occurrences of actions and their preconditions from the cancelling actions and their preconditions, and it is not obvious how they can be learned separately.

Let us consider again the circuit of Figure 2. Fluents sw_1 and sw_3 are independent effects, hence the effect axioms learned for them do not differ from those learned when a single action is executed. With respect to the state of *light*, the consequences of executing, e.g. t_1, depend on the previous value of sw_2 in a different way depending on whether t_2 is also executed or not simultaneously. For instance, when both switches are closed simultaneously, they produce independent effects and the accumulative effect of activating the light. As a consequence, many more clauses have to be learned to cope with every possible interaction between actions. However, *LRAC* learned the same definition for *light*, because the ramification constraints express a dependence between the switches and *light* without making reference to the actions executed. Similarly for the *relay* that is controlled by

sw_1 and sw_3. With respect to sw_2, *LRAC* found the following definition:

```
caused(sw2,f,A) :- happens(t2,A),holds(sw2,A-1).
caused(sw2,f,A) :- holds(relay,A).
caused(sw2,t,A) :- happens(t2,A),
                   not happens(t1,A),not happens(t3,A),
                   -holds(sw2,A-1),-holds(relay,A-1).
caused(sw2,t,A) :- happens(t2,A),not happens(t1,A),
                   -holds(sw2,A-1),-holds(sw1,A-1).
caused(sw2,t,A) :- happens(t2,A),not happens(t3,A),
                   -holds(sw2,A-1),-holds(sw3,A-1).
caused(sw2,t,A) :- happens(t2,A),happens(t1,A),happens(t3,A),
                   -holds(sw2,A-1),-holds(relay,A-1),
                   holds(sw1,A-1).
caused(sw2,t,A) :- happens(t2,A),happens(t1,A),happens(t3,A),
                   -holds(sw2,A-1),-holds(relay,A-1),
                   holds(sw3,A-1).
```

When only atomic actions are executed, *LRAC* learned two effect axioms and one ramification constraint. However, if we execute t_2 and other action that activates the *relay* at the same time (e.g., t_3), sw_2 may be closed for an instant but it will be definitely open when the relay becomes active, so that the effect axiom for the positive value of sw_2 needs to consider all subsets of cancelling actions that could activate the relay, together with their preconditions. This makes the descriptions of actions cumbersome and difficult for complex domains, and consequently harder to learn.

In most current action formalisms (VITAL [7], CCALC [17], etc.), such complex *qualifications* to actions are often expressed apart from the effect axioms in the form of *integrity constraints* that forbid some next states produced by effect axioms, that correspond to non-allowed states of a domain [12]. For instance, in the previous circuit, we can extract a set of constraints that correspond to those states not allowed for the switches, for instance, that sw_2 is never closed when the relay is active. From such constraint, we have the following alternative definition for the positive value of sw_2:

```
caused(sw2,t,A) :- happens(t2,A),-holds(sw2,A-1).
:- holds(sw2,A),holds(relay,A).
```

In this case, the constraint specializes the overgeneral effect axiom for sw_2, so that only valid and legal resulting states are allowed. By doing so, the state of sw_2 does not depend on what other actions are executed apart from t_2, but on the state of the *relay* in the resulting situation, which provides with a much simpler definition. A consequence is that these constraints can make programs sensibly shorter and thus easier to learn. The interest of such qualification constraints is clearer when learning the effects of concurrent actions, since we may need to anticipate the effects of the co-occurring actions. We are currently considering to learn such qualification constraints to specialize overgeneral effect axioms instead of fully specialize them by adding preconditions. The choice of the constraints is not independent from the rules of the theory, and the difficulty lies in finding the "relevant" constraints that would compensate correctly for the rules of the theory [6].

6 Conclusions

A logic programming formalization of action domains has been well-studied, yet not much previous work [21] exists on the combination with learning methods. The presented approach improves on previous ones by providing more expressivity, so that it can solve some problems that were unsolvable before. Further work needs to be done to show the adaptation to more and increasingly more complex scenarios, and with different noise levels, e.g., to improve its adequation for dealing with real robot's environments. More complex phenomena can be dealt with in an homogeneous way, by slightly changing the form of the rules to be learned or building special fluents, e.g., for durative actions or delayed effects. Further work also includes the management of continuously varying parameters as a consequence of a process execution.

Acknowledgements

This research was supported in part by the Government of Spain, grant TIC2001-0393.

References

[1] C. Baral and M. Gelfond. Reasoning about effects of concurrent actions. *Journal of Logic Programming*, 31(1–3):85–117, 1997.

[2] C. Baral and J. Lobo. Defeasible specifications in action theories. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI-97)*, pages 1441–1446, San Francisco, 1997. Morgan Kaufmann Publishers.

[3] S. Benson. Inductive learning of reactive action models. In *Proc. 12th International Conference on Machine Learning*, pages 47–54. Morgan Kaufmann, 1995.

[4] P. Cabalar, M. Cabarcos, and R. P. Otero. PAL: Pertinence action language. In *Proceedings of the 8th Intl. Workshop on Non-Monotonic Reasoning NMR'2000 (Collocated with KR'2000)*, Breckenridge, Colorado, USA, 2000.

[5] J. G. Carbonell and Y. Gil. Learning by experimentation: The operator refinement method. In

R. S. Michalski and Y. Kodratoff, editors, *Machine Learning: An Artificial Intelligence Approach, Volume III*. Morgan Kaufmann, San Mateo, California, 1990.

[6] Y. Dimopoulos, S. Džeroski, and A. Kakas. Integrating explanatory and descriptive learning in ILP. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI-97)*, pages 900–907, San Francisco, 1997. Morgan Kaufmann Publishers.

[7] P. Doherty and J. Kvarnstrom. Tackling the qualification problem using fluent dependency constraints: Preliminary report. In *International Workshop on Temporal Representation and Reasoning*, pages 97–104, 1998.

[8] R. E. Fikes and N. J. Nilsson. STRIPS: A new approach to the application of theorem proving to problem solving. In D. C. Cooper, editor, *Proceedings of the 2nd International Joint Conference on Artificial Intelligence*, pages 608–620, London, UK, 1971. William Kaufmann.

[9] M. Gelfond and V. Lifschitz. Representing action and change by logic programs. *Journal of Logic Programming*, 17:301–321, 1993.

[10] M. Gelfond and V. Lifschitz. Action languages. *Electronic Transactions on AI*, 3(16), 1998.

[11] M. Gelfond, V. Lifschitz, and A. Rabinov. What are the limitations of the situation calculus? In *Automated Reasoning: Essays in Honor of Woody Bledsoe*, pages 167–179. Kluwer Academic Publishers, Dordrecht, 1986.

[12] Matthew L. Ginsberg and David E. Smith. Reasoning about action II: the qualification problem. In Frank M. Brown, editor, *The frame problem in artificial intelligence: proc. of 1987 workshop*. Morgan Kaufmann, 1987.

[13] R. Kowalski and M. Sergot. A logic-based calculus of events. *New Generation Computing*, 4:67–95, 1986.

[14] F. Lin. Embracing causality in specifying the indirect effects of actions. In C. S. Mellish, editor, *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. Morgan Kaufmann, 1995.

[15] D. Lorenzo. *Learning non-monotonic Logic Programs to Reason about Actions and Change*. PhD thesis, Departamento de Computación, Facultade de Informática, Univ. A Coruña, Spain, 2001.

[16] D. Lorenzo and R.P. Otero. Learning to reason about actions. In *W. Horn (Ed.), Proceedings of the 14th European Conference on Artificial Intelligence*, pages 435–439. IOS Press, Amsterdam, 2000.

[17] N. McCain and H. Turner. A causal theory of ramifications and qualifications. In *Proc. of the Intl. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 1978–1984, 1995.

[18] J. McCarthy and P.J. Hayes. Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence*, 4:463–502, 1969.

[19] D. McDermott. The 1998 ai planning systems competition. *AI Magazine*, 21(2):35–55, 2000.

[20] T. M. Mitchell, P. E. Utgoff, and R. Banerji. Learning by experimentation: acquiring and refining problem-solving heuristics. In R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach*, pages 163–190. Morgan Kaufmann, San Mateo, California, 1983.

[21] S. Moyle and S. Muggleton. Learning programs in the event calculus. In N. Lavrač and S. Džeroski, editors, *Proceedings of the Seventh Inductive Logic Programming Workshop (ILP97)*, LNAI 1297, pages 205–212, Berlin, 1997. Springer-Verlag.

[22] S. Muggleton. Inverse entailment and Progol. *New Generation Computing, Special issue on Inductive Logic Programming*, 13(3-4):245–286, 1995.

[23] S. Muggleton and L. De Raedt. Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19/20:629–679, 1994.

[24] R. Reiter. *Knowledge in action: Logical Foundations for describing and implementing dynamical systems*. MIT Press, 1998.

[25] W. M. Shen. *Autonomous Learning from the Environment*. Computer Science Press, 1994.

[26] M. Thielscher. Ramification and causality. *Artificial Intelligence Journal*, 1-2(89):317–364, 1997.

[27] X. Wang. Learning by observation and practice: an incremental approach for planning operator acquisition. In *Proc. 12th International Conference on Machine Learning*, pages 549–557. Morgan Kaufmann, 1995.

Planning with Complex Actions

Sheila McIlraith
Knowledge Systems Laboratory
Department of Computer Science
Gates Building, 2A wing
Stanford University
Stanford, CA 94305, USA.
sam@ksl.stanford.edu

Ronald Fadel
Knowledge Systems Laboratory
Department of Computer Science
Gates Building, 2A wing
Stanford University
Stanford, CA 94305, USA.
rfadel@ksl.stanford.edu

Abstract

In this paper we address the problem of planning with complex actions. We are motivated by the problem of automated Web service composition, in which planning *must* be performed using predefined complex actions or services as the building blocks of a plan. Planning with complex actions is also compelling in primitive action planning domains because it enables the exploitation of reusable subplans, potentially improving the efficiency of planning. This paper provides a formal, semantically-justified account of how to plan with complex actions using operator-based planning techniques. A key contribution of this work is the definition, characterization, and computation of preconditions and conditional effects for complex actions. While we use the situation calculus and Golog to formalize the task and our solution, the results in this paper are directly applicable to most action theories and planning systems. In particular, we have developed a PDDL-equivalent compiler that computes the preconditions and effects of complex actions, thus enabling wide-spread use of these results. Finally we provide an approach to planning that enables us to exploit deductive plan synthesis or alternatively ADL planners to plan with complex actions. Our approach to complex-action planning is sound and complete relative to the corresponding primitive action domain.

1 Introduction

Given a description of an initial state, a goal state, and a set of actions, the planning task is to generate a sequence of actions that, when performed starting in the initial state, will terminate in a goal state. Typically, actions are primitive and are described in terms of their precondition, and (conditional) effects. Our interest is in planning with complex actions as the building blocks for a plan. Complex actions are actions composed of primitive actions using typical programming language constructs. E.g.,complex actions move(obj,orig,dest) and goToAirpt(loc) are defined as:

move(obj,orig,dest) $\doteq$[1] pickup(obj,orig);putdown(obj,dest)
goToAirpt(loc) $\doteq$ **if** loc=Univ **then** shuttle(Univ,PA);
train(PA,MB);shuttle(MB,SFO) **else** taxi(loc,SFO)

Our primary motivation for investigating complex action planning is to automate *Web service composition* (e.g., [13]). Web services are self-contained Web-accessible computer programs, such as the airline ticket service at www.ual.com, or the weather service at www.weather.com. These services can be conceived as complex actions. Consider ual.com's buyAirTicket($\vec{x}$) service. This service can be described as the complex action locateFlight($\vec{x}$); **if** Available($\vec{x}$) $\wedge$ OKPrice($\vec{x}$) **then** buyAirTicket($\vec{x}$);...[2]. The task of automated Web service composition is to automatically sequence together Web services such as buyAirTicket($\vec{x}$) or getWeather($\vec{y}$) into a composition that achieves some user-defined objectives. The task of automated Web service composition is, by necessity, a problem of planning with complex actions. But how do we represent these complex actions (Web services) and how do we plan with them?

What makes planning with complex actions difficult is that the traditional characterization of actions as operators with preconditions and effects does not apply, making operator-based planning techniques such as Blackbox, FF, GraphPlan, BDDPlan, etc., inapplicable, at least at face value. In this paper we provide a formal, semantically-justified account of how to characterize, represent and precompile the preconditions and effects of complex actions, such as buyAirTicket($\vec{x}$), under a frame assumption [16]. This enables us to treat complex actions such as buyAirTicket($\vec{x}$) as planning operators and to apply standard planning tech-

[1] Denotes "defined as."
[2] Example is simplified for illustration purposes.

niques to planning with complex actions. Planning results in a plan in terms of complex actions from which a plan in terms of primitive actions is easily expanded, if desired[3].

A secondary motivation for this work is to improve the efficiency of planning by representing useful (conditional) plan segments as complex actions. As we show, our approach to planning with complex actions can dramatically improve the efficiency of plan generation by reducing the search space size and the length of a plan.

The idea of planning with some form of abstraction or aggregation is not new, and there has been a variety of work in this area including ABStrips (e.g., [17]), planning with macro-operators (e.g., [11] and [6]), and most notably HTN planning (e.g., [5]). Our work is fundamentally different from these approaches, and in particular from HTN planning, both in terms of i) the representation of complex actions (aka HTN *non-primitive tasks*), and ii) the method of planning. In this paper we precompile complex actions into planning operators described in terms of preconditions and effects that embody all possible evolutions of the complex action. In contrast, HTN planners do not use a declarative representation of the preconditions and effects of tasks. Rather, methods are associated with tasks, and tasks are pre-arranged into a network of compositions, without the full programming constructs we use to describe complex actions [18]. Further, HTN planners operate by searching for plans that accomplish task networks using task decomposition and conflict resolution. In contrast, having precompiled our complex actions, we can apply standard operator-based planning techniques to generate a plan, followed by plan expansion.

Our work is somewhat similar in methodology to [2], which proposes to encode planning constraints by compiling the constraints together with the original planning problem into a new unconstrained problem. The resultant planning problem can be solved using classical planning methods, and the resultant plan *decompiled* to provide a solution in the original problem domain. The general methodology of compilation and subsequent expansion is similar to what we propose. Nevertheless, the general problem is different. We are compiling complex actions into new plan operators. These complex actions represent Web services that we wish to reason with as black-box components. The constraints used in [2] are constraints upon the domain, and thus capture different types of planning information than our more procedural complex actions. Further the formal treatment and results are different.

We also contrast our work to the use of Golog (e.g., [12]) in planning. In this paper we use Golog as the formal language to describe complex actions, however the role these actions play in planning is very different. Golog complex actions are traditionally used to specify non-deterministic programs. In combination with deductive plan synthesis [7], a Golog program expands to a situation calculus formula which constrains the search space for a plan. This is similar to the role of domain-specific knowledge, as exemplified by systems such TALPlanner [4], BDDPlan [10] and ASP [18]. In all these systems, complex actions constrain the search space, but are not used as operators in plan construction.

The research presented in this paper is of both theoretical and practical significance. From a theoretical standpoint, we provide a semantically-justified means of characterizing the preconditions, effects and successor situations of complex actions under a frame assumption, that embodies all possible trajectories of a complex action. This enables us to not only use operator-based planning methods to plan with complex actions, but also to prove formal properties of our approach. In particular, we prove that our approach to planning is sound and complete relative to corresponding primitive action domains. From a practical perspective, analysis shows a significant increase in the efficiency of planning with complex actions, relative to primitive action planning. We illustrate potential speedup with some experiments on the briefcase domain, using the FF planner ([9]). Finally, this paper provides a principled approach to automating Web service composition, that has far-reaching application to automated component-based software composition

2 Background: Situation Calculus & Golog

We use the situation calculus and Golog to formalize the task and our solution. The expressive power and formal semantics of the situation calculus provide the theoretical foundations for our work, and for the later translation to PDDL.

Briefly, the situation calculus is a logical language for specifying and reasoning about dynamical systems [16]. In the situation calculus, the state of the world is expressed in terms of functions and relations (fluents) relativized to a particular situation s, e.g., $F(\vec{x}, s)$. A situation s is a history of the primitive actions, e.g., a, performed from an initial, distinguished situation S_0. The function $do(a, s)$ maps a situation and an action into a new situation. A situation calculus theory $\mathcal{D}$ comprises the following sets of axioms:

- domain independent foundational axioms, Σ.
- successor state axioms, $\mathcal{D}_{SS}$, one for every fluent F.
- action precondition axioms, $\mathcal{D}_{ap}$, one for every action a in the domain,which define $Poss(a, s)$.
- axioms describing the initial situation, $\mathcal{D}_{S_0}$.

[3] For many Web service applications, expansion is not relevant.

• unique names axioms for actions, $\mathcal{D}_{una}$.

Successor state axioms, originally proposed [15] to address the frame problem, are created by compiling effect axioms into axioms of this form[4]: $F(\vec{x}, do(a,s)) \equiv \Phi_F(\vec{x}, a, s)$ where $\Phi_F(\vec{x}, a, s) = \gamma_F^+(\vec{x}, a, s) \vee (F(\vec{x}, s) \wedge \neg\gamma_F^-(\vec{x}, a, s))$. (See [16, pg.28-35] for details.)

Example: In the interest of simplicity, we illustrate concepts in this paper in terms of an action theory with three actions $pickup(x), putdown(x)$ & $drop(x)$, and three fluents $holding(x)$, $broken(x)$ & $hot(x)$. (1)-(3) comprise $\mathcal{D}_{ap}$, and (4)-(6) comprise $\mathcal{D}_{SS}$[5].

$$Poss(pickup(x), s) \equiv \neg holding(x, s) \quad (1)$$
$$Poss(drop(x), s) \equiv holding(x, s) \quad (2)$$
$$Poss(putdown(x), s) \equiv holding(x, s) \quad (3)$$
$$holding(x, do(a, s)) \equiv a = pickup(x) \vee holding(x, s) \wedge a \neq putdown(x) \wedge a \neq drop(x) \quad (4)$$
$$broken(x, do(a, s)) \equiv a = drop(x) \vee broken(x, s) \quad (5)$$
$$hot(x, do(a, s)) \equiv hot(x, s) \quad (6)$$

Golog (e.g., [12, 16, 3]) is a high-level logic programming language for the specification and execution of complex actions in dynamical domains. It builds on top of the situation calculus by providing extralogical constructs for assembling primitive situation calculus actions, into complex actions δ. [3] shows how these complex actions can be considered to be first-class objects in the language. $Do(\delta, s, s')$ is an abbreviation that macro-expands into a situation calculus formula, as defined inductively below. The formula says that it is possible to reach s' from s by executing a sequence of actions specified by δ [16].

Prim. action: $Do(a, s, s') \doteq Poss(a, s) \wedge s' = do(a[s], s)$
Test: $Do(\phi?, s, s') \doteq \phi[s] \wedge s = s'$
Seq.: $Do(\delta_1; \delta_2, s, s') \doteq \exists s''.Do(\delta_1, s, s'') \wedge Do(\delta_2, s'', s')$
Nondet. act.: $Do(\delta_1 \mid \delta_2, s, s') \doteq Do(\delta_1, s, s') \vee Do(\delta_2, s, s')$
Nondet. arg.: $Do((\pi x)\delta(x), s, s') \doteq \exists x.Do(\delta(x), s, s')$

The construct, **if** ϕ **then** δ_1 **else** δ_2 **endIf** is defined as $[\phi?; \delta_1] \mid [\neg\phi?; \delta_2]$. The Golog language also includes nondeterministic iteration, δ^*, which executes δ zero or more times. The while-loop construct,**while** ϕ **do** δ **endWhile** is defined in terms of nondeterministic iteration as $[\phi? : \delta]^*; \neg\phi?$. For now, we exclude nondeterministic iteration, and while-loops, whose macro-expansions are second order, and which may be non-terminating. Instead, we define a bounded notion of while, **while** $_k(\phi)$ δ, which is guaranteed to terminate, and is commonly used in Web services. **while** $_k(\phi)$ δ executes like the original while-loop except that it loops at most k times, even if ϕ still holds after the k^{th} iteration. Formally, **while** $_k(\phi)$ δ corresponds to k conditional branchings as follow:

$$\textbf{while}\ _1(\phi)\ \delta \doteq \textbf{if}\ \phi\ \textbf{then}\ \delta\ \textbf{endIf}\ [6] \quad (7)$$
$$\textbf{while}\ _k(\phi)\ \delta \doteq \textbf{if}\ \phi\ \textbf{then}\ [\delta; \textbf{while}\ _{k-1}(\phi)\delta]\ \textbf{endIf} \quad (8)$$

[4] For space, we will only consider relational fluents here.

[5] Notation: formulae are universally quantified with maximum scope unless noted. Action arguments suppressed.

A deterministic version of the choice construct (π') is defined in a longer paper. These constructs are used to specify complex actions such as buyAirTicket($\vec{x}$) or goToAirpt(loc). Traditional usage of Golog is to apply deductive plan synthesis to find a sequence of actions $\vec{a} = [a_1, \ldots, a_n]$ that realizes a Golog program, δ relative to domain theory, $\mathcal{D}$. I.e., $\mathcal{D} \models Do(\delta, S_0, do(\vec{a}, S_0))$. $Do(\delta, S_0, do(\vec{a}, S_0))$ denotes that the Golog program δ, starting execution in S_0 will legally terminate in situation $do(\vec{a}, S_0)$, where $do(\vec{a}, S_0)$ is an abbreviation for $do(a_n, do(a_{n-1}, \ldots, do(a_1, S_0)))$.

3 Problem: Planning with Complex Actions

Given a set of primitive actions, $\mathcal{A}$ together with an associated set of complex actions, $\Delta_{\mathcal{A}}$, our objective is to use an operator-based planner to compose complex and primitive actions to achieve some goal. To do this, we must characterize the preconditions, effects, and the situation resulting from performing a complex action.

3.1 Preconditions, Effects, Resulting Situations

For analysis, our actions $\mathcal{A}$ are axiomatized in a situation calculus action theory $\mathcal{D}$, and our complex actions $\Delta_{\mathcal{A}}$ are described in Golog. For now, we restrict our focus to terminating complex actions described in Section 2.

Resulting Situation: We wish to characterize the situation resulting from performing the complex action δ. Observe that many complex actions are nondeterministic. They may have several different executions, each terminating in a different situation. As such, we can't define a function analogous to $do(a, s)$. Instead, we introduce the abbreviation $do_{ca}(\delta, s)$ to denote a situation resulting from performing complex action δ in s. $do_{ca}(\delta, s)$ ranges over the set of *executable* situations and corresponds to a so-called *ghost situation* [16, pg.52-53], when δ is not physically realizable. The interpretation of $do_{ca}(\delta, s)$ is constrained by the following axiom, which is added to $\mathcal{D}$ producing theory $\mathcal{D}_{ca}$.

For all complex actions δ and situations s:

$$Do(\delta, s, do_{ca}(\delta, s)) \vee (\neg\exists s''.Do(\delta, s, s'') \wedge \neg executable(do_{ca}(\delta, s))) \quad (9)$$

where $executable(s)$ denotes a situation, all of whose actions in the situation action history are $Poss$-ible [16]. I.e., $executable(s) \doteq (\forall a, s^*).do(a, s^*) \sqsubseteq {}^7 s \rightarrow Poss(a, s^*)$ It follows that:

$$\mathcal{D}_{ca} \models \forall s.executable(s) \wedge Do(\delta, s, do_{ca}(\delta, s)) \rightarrow executable(do_{ca}(\delta, s)) \quad (10)$$

[6] **if -then -endIf** is the obvious variant of **if -then -else -endIf**.

[7] The order relation on situations in the situation tree [16].

Preconditions: $Poss_{ca}(\delta, s)$ denotes the preconditions of complex action δ. Intuitively, the preconditions of a complex action are the preconditions of all the actions that make up the execution of δ. E.g., for $a_1; a_2$,

$$Poss_{ca}(a_1; a_2, s) \equiv Poss(a_1, s) \wedge Poss(a_2, do(a_1, s)).$$

This is captured tidily in the inductive definition of Do. We define the precondition of complex action δ , $Poss_{ca}(\delta, s)$ as:

$$Poss_{ca}(\delta, s) \equiv \Pi^*_\delta(s) \tag{11}$$

where $\Pi^*_\delta(s) \equiv \exists s'.Do(\delta, s, s')$. These are *intermediate* action precondition axioms.

Proposition 1 (Properties of $Poss_{ca}(\delta, s)$)
These axioms follow from $\mathcal{D}_{ca} \cup (11)$.

$$\begin{aligned} Poss_{ca}(\delta, s) &\equiv Do(\delta, s, do_{ca}(\delta, s)) \\ executable(s) \wedge Poss_{ca}(\delta, s) &\equiv executable(do_{ca}(\delta, s)) \end{aligned}$$

Effects: Intuitively the effects of a complex action are the effects of each action in the execution of δ, modulo the effects of subsequent actions. We assume that fluents whose truth value is not changed by an action, persist. $F(\vec{x}, do_{ca}(\delta, s))$ denotes that fluent F is true in the situation resulting from performing complex action δ in s. We capture the effects of complex actions as successor state axioms. Since all but trivial complex actions involve multiple intermediate situations, strictly speaking, we cannot define successor state axioms for complex actions. Rather, we define the notion of a pseudo-successor state axiom. Here we define *intermediate* pseudo-successor state axioms, making them "Markovian" in the section to follow via regression.

$$Poss_{ca}(\delta, s) \rightarrow [F(\vec{x}, do_{ca}(\delta, s)) \equiv \Phi^*_F(\vec{x}, \delta, s)], \text{ where,}$$
$$\Phi^*_F(\vec{x}, \delta, s) \equiv \exists s'.Do(\delta, s, s') \wedge F(\vec{x}, s') \wedge s' = do_{ca}(\delta, s). \tag{12}$$

We need the $s' = do_{ca}(\delta, s)$ since some complex actions are nondeterministic. This enables us to identify the particular sequence of actions in the instantiation of the complex action that leads to the truth/falsity of the fluent F.

3.2 Pseudo-Markovian Complex Actions

In order to plan with complex actions as operators, we must make our characterization *pseudo-markovian*. That is, we wish to characterize the preconditions strictly in terms of the situation in which the complex action execution is initiated, and the effects, strictly in terms of the initiating and terminating situations of the complex action. To do so we appeal to regression rewriting [19], regressing over the successor state axioms for the *primitive actions* in our domain theory $\mathcal{D}$. Unfortunately, the formulae over which we need to regress are not, by definition, regressable using $\mathcal{R}$ [16, pg.62], since we are not regressing to S_0, and since the macro-expansion of $Do(\delta, s, s')$ does not yield a nested representation of situations. Since regression is a syntactic rewriting, this is problematic. We define a suitable (small) variant of Reiter's regression operator, $\mathcal{R}^s$, that first rewrites the macro-expansion of Do so that situations are expressed as nested do's, and that enables regression to an arbitrary situation s, rather than to S_0. We define the preconditions and effects of δ in terms of a set of action precondition axioms, $\mathcal{D}_{caap}$, of the form of (13) and a set of pseudo-successor state axioms, $\mathcal{D}_{caSS}$, of the form of (15).

Preconditions:
Action Precondition Axioms, $\mathcal{D}_{caap}$, one for every $\delta \in \Delta$:

$$Poss_{ca}(\delta, s) \equiv \Pi_\delta(s) \tag{13}$$

where $\Pi_\delta(s) \equiv \mathcal{R}^s[\Pi^*_\delta(s)]$ from (11), i.e. $\mathcal{R}^s[\exists s'.Do(\delta, s, s')]$.

Example (continued): Consider the complex action $pickup(x)$; **if** $hot(x)$ **then** $drop(x)$ **else** $putdown(x)$ **endIf**, which we denote as δ_1 for parsimony. Its action precondition axiom is defined as follows.

$$\begin{aligned} &Poss_{ca}(\delta_1, s) \equiv \\ &\quad \mathcal{R}^s[\exists s', s''.Poss(pickup(x), s) \wedge s'' = do(pickup(x), s) \wedge \\ &\quad ((hot(x, s'') \wedge Poss(drop(x), s'') \wedge s' = do(drop(x), s'')) \\ &\quad \vee (\neg hot(x, s'') \wedge Poss(putdown(x), s) \wedge \\ &\quad s' = do(putdown(x), s''))))] \end{aligned} \tag{14}$$

Following our regression, $Poss_{ca}(\delta_1, s) \equiv \neg holding(x, s)$.

Successor State Axioms: Observe that while a situation calculus axiomatization has one successor state axiom for every fluent, we currently define one pseudo-successor state axiom for every fluent-complex action pair.

Pseudo-Successor State Axioms, $\mathcal{D}_{caSS}$, one for every fluent-complex action pair:

$$Poss_{ca}(\delta, s) \rightarrow [F(\vec{x}, do_{ca}(\delta, s)) \equiv \Phi_F(\vec{x}, \delta, s)] \tag{15}$$

where $\Phi_F(\vec{x}, \delta, s) \equiv \mathcal{R}^s[\Phi^*_F(\vec{x}, \delta, s)]$, $\mathcal{R}^s[\Phi^*_F(\vec{x}, \delta, s)] \equiv \mathcal{R}^s[\exists s'.Do(\delta, s, s') \wedge F(\vec{x}, s') \wedge do_{ca}(\delta, s) = s']$

Example (continued): The pseudo-successor state axiom for fluent $broken(x, do_{ca}(\delta_1, s))$ is:

$$\begin{aligned} &Poss_{ca}(\delta_1, s) \rightarrow [broken(x, do_{ca}(\delta_1, s)) \equiv \\ &\quad \mathcal{R}^s[\exists s', s''.Poss(pickup(x), s) \wedge s'' = do(pickup(x), s) \wedge \\ &\quad ((hot(x, s'') \wedge Poss(drop(x), s'') \wedge s' = do(drop(x), s'')) \\ &\quad \vee (\neg hot(x, s'') \wedge Poss(putdown(x), s) \wedge \\ &\quad s' = do(putdown(x), s''))) \wedge \\ &\quad broken(x, s') \wedge do_{ca}(\delta_1, s) = s'] \end{aligned} \tag{16}$$

Applying our $\mathcal{R}^s$ regression operator, (16) becomes:

$$\begin{aligned} &Poss_{ca}(\delta_1, s) \rightarrow (\ broken(x, do_{ca}(\delta_1, s)) \equiv \\ &\quad \neg holding(x, s) \wedge [hot(x, s) \wedge \\ &\quad do_{ca}(\delta_1, s) = do(drop(x), do(pickup(x), s)) \\ &\quad \vee \neg hot(x, s) \wedge broken(x, s) \wedge \\ &\quad do_{ca}(\delta_1, s) = do(putdown(x), do(pickup(x), s))]\ ) \end{aligned}$$

Though the computation looks complex, regression rewriting is a powerful tool and the final pseudo-successor state axiom is simple. Observe that a pseudo-successor state axiom not only defines the conditions under which fluent F is true after performing complex action δ, but it also defines the action trajectory upon which the truth of F is predicated. This is most valuable with nondeterministic actions.

Note that when the definition of $Poss_{ca}(\delta, s)$ and the intermediate pseudo-successor state axiom, ((11) and (12), respectively) are conjoined to $\mathcal{D}_{ca}$, they entail the complex action precondition axioms and the complex action pseudo-successor state axioms.

Proposition 2 $\mathcal{D}_{ca} \cup (11) \cup (12) \models \mathcal{D}_{caap} \cup \mathcal{D}_{caSS}$

Effect axioms: While we have encoded the effects of our complex actions, together with a solution to the frame problem in terms of pseudo-successor state axioms, many planners use effect axioms, rather than successor state axioms, solving the frame problem in the procedural code of their planner, rather than representationally. Hence, for completion we define effect axioms for complex actions, $\mathcal{D}_{caef}$.

Effect Axioms $\mathcal{D}_{caef}$, up to one positive effect axiom and one negative effect axiom for every fluent - complex action pair, where the execution of δ can potentially change the truth value of fluent $F \in \mathcal{F}$:

$$Poss_{ca}(\delta, s) \wedge \epsilon_F^+(\vec{x}, s) \rightarrow F(\vec{x}, do_{ca}(\vec{x}, \delta, s)) \quad (17)$$

$$Poss_{ca}(\delta, s) \wedge \epsilon_F^-(\vec{x}, s) \rightarrow \neg F(\vec{x}, do_{ca}(\vec{x}, \delta, s)) \quad (18)$$

Proposition 3 (Effect Axioms Entailment)

$$\mathcal{D} \cup \mathcal{D}_{caap} \cup \mathcal{D}_{caSS} \models \mathcal{D}_{caef}$$

I.e., the positive and negative complex action effect axioms are entailed by the pseudo-successor state axioms. Hence, we can easily extract effect axioms for complex actions from our pseudo-successor state axioms.

In this section we have provided a representation of the preconditions, successor state axioms and effects of complex actions under a frame assumption. They are characterized in terms of $\mathcal{D}_{caap}$, and $\mathcal{D}_{caSS}$, and follow from the semantically-justified account of actions in the situation calculus. In the section to follow, we show how these representations of complex actions lead to a simple approach to planning with complex actions.

4 Complex Actions Planning

Given our operator-based characterization of complex actions in terms of their preconditions and effects, we turn to the problem of operator-based planning with these complex actions. For now, we restrict our consideration to the subset of complex actions that are deterministic, I.e., primitive actions a, sequences $\delta_1; \delta_2$, conditional **if** ϕ **then** δ_1 **else** δ_2 **endIf**, and **while** $_k(\phi)$ δ, plus others described in a longer paper.

Following the problem statement in Section 3, our approach is to take as input $[\mathcal{T}_\mathcal{A}, \Delta_\mathcal{A}]$ – an action theory $\mathcal{T}_\mathcal{A}$ and a set of complex actions $\Delta_\mathcal{A}$, both defined in terms of actions in $\mathcal{A}$. Following the results in the previous section, we **COMPILE** $[\mathcal{T}_\mathcal{A}, \Delta_\mathcal{A}]$ into a new theory $\mathcal{T}_{\mathcal{A}'}$, in terms of actions $\mathcal{A}'$ (generally $\mathcal{A} \subseteq \mathcal{A}'$), where each complex action in $\Delta_\mathcal{A}$ corresponds to a new primitive action in $\mathcal{A}'$. Next, **PLAN**ning is performed in $\mathcal{T}_{\mathcal{A}'}$ to produce a plan in terms of $\mathcal{A}'$. To extract a plan in terms of the primitive actions, we **REWRITE** the theory, replacing primitive actions from $\mathcal{A}'$ by their corresponding complex actions, $\Delta_\mathcal{A}$. Finally, using $\mathcal{T}_\mathcal{A}$, the resulting sequence of primitive actions is **EXPAND**ed from the plan in $\mathcal{A}'$ into a plan in terms of $\mathcal{A}$.

Next, we show how this approach is realized, first using the situation calculus and deductive plan synthesis, and then using an arbitrary operator-based planning system that allows conditional effects of actions in PDDL.

4.1 Deductive Plan Synthesis and Expansion

The following is the theory with primitive actions $\mathcal{T}_\mathcal{A}$.

$\mathcal{T}_\mathcal{A} = \Sigma \cup \mathcal{D}_{ap} \cup \mathcal{D}_{SS} \cup \mathcal{D}_{una} \cup \mathcal{D}_{S_0}$.

(1) COMPILE$[\mathcal{T}_\mathcal{A}, \Delta_\mathcal{A}] \rightarrow \mathcal{T}_{\mathcal{A}'}$**:**
- Define $\mathcal{D}_{caap}$ and $\mathcal{D}_{caSS}$ as described in Section 3.2.
- $\mathcal{D}'_{ap} \leftarrow \mathcal{D}_{caap} \cup \mathcal{D}_{ap}$. $\mathcal{D}'_{SS} \leftarrow \mathcal{D}_{caSS}$. $\mathcal{A}' \leftarrow \mathcal{A}$.
- $\forall \delta_i \in \Delta_A$: Create a primitive action a'_i. Substitute "a'_i" for "δ_i" in $\mathcal{D}'_{ap}$& $\mathcal{D}'_{SS}$. $\mathcal{A}' \leftarrow \mathcal{A}' \cup \{a'_i\}$.
- $\mathcal{D}'_{SS} \leftarrow$ MERGE$(\mathcal{D}'_{SS}, \mathcal{D}_{SS})$. Update $\mathcal{D}_{una}$ to $\mathcal{D}'_{una}$.

COMPILE produces a situation calculus theory in actions $\mathcal{A}'$, comprising all the original primitive actions $\mathcal{A}$ plus new primitive actions corresponding to each complex action in $\Delta_\mathcal{A}$. $\mathcal{T}_{\mathcal{A}'} = \Sigma \cup \mathcal{D}'_{ap} \cup \mathcal{D}'_{SS} \cup \mathcal{D}'_{una} \cup \mathcal{D}_{S_0}$

(2) PLAN$[\mathcal{T}_{\mathcal{A}'}$**,goal]** $\rightarrow$ **plan**$[\mathcal{A}']$**:** Given a goal formula, $Goal(s)$ in the language of $\mathcal{T}_\mathcal{A}$, planning can be achieved via deductive plan synthesis in $\mathcal{T}_{\mathcal{A}'}$. Following [7, 16], $\mathcal{T}_{\mathcal{A}'} \vdash \exists s.Goal(s)$. From the binding of s, we can read off a plan $[a'_1 \ldots, a'_n], a'_i \in \mathcal{A}'$, a plan in $\mathcal{A}'$. [16] describes a variety of situation calculus planners implemented in Prolog.

(3) REWRITE[plan$[\mathcal{A}']]$ $\rightarrow$ **plan**$[\mathcal{A}, \Delta_\mathcal{A}]$**:** Rewrite the plan $[a'_1 \ldots, a'_n], a'_i \in \mathcal{A}'$ as a plan $[\alpha_1, \ldots, \alpha_n]$ in $(\mathcal{A}, \Delta_\mathcal{A})$, where $\alpha_i = a'_i$, for all $a'_i \in \mathcal{A}$, otherwise α_i equals the corresponding δ_i from the compilation in Step (1).

(4) EXPAND[plan$[\mathcal{A}, \Delta_\mathcal{A}]$**,**$\mathcal{T}_\mathcal{A}] \rightarrow$ **plan**$[\mathcal{A}]$: Use our same deductive machinery to extract a final plan in $\mathcal{A}$ from our plan in $(\mathcal{A}, \Delta_\mathcal{A})$, by expanding the complex actions in $[\alpha_1, \ldots, \alpha_n]$. We do so by trivially rewriting our plan as a

sequence of complex actions in Golog $\delta_G = \alpha_1; \alpha_2; \ldots; \alpha_n$. A Golog interpreter, written in Prolog will return a binding for situation s' where $\mathcal{T}_{\mathcal{A}} \vdash (\exists s').Do(\delta_G, s, s') \wedge Goal(s')$. From the situation s' we can read off a plan $[a_1 \ldots, a_m], a_j \in \mathcal{A}$.

Note that every plan our approach finds is also a plan in the original primitive action theory, and vice-versa.

Theorem 1 *$\mathcal{T}_{\mathcal{A}'}$ and $\mathcal{T}_{\mathcal{A}}$ are defined as in Section 4.1. Let $Goal(s)$ be a formula uniform in s such that $Goal(s) \in \mathcal{L}(\mathcal{T}_{\mathcal{A}}) \cap \mathcal{L}(\mathcal{T}_{\mathcal{A}'})$, the intersection of the languages of $\mathcal{T}_{\mathcal{A}}$ and $\mathcal{T}_{\mathcal{A}'}$ respectively. For all ground situations σ' of $\mathcal{T}_{\mathcal{A}'}$, $\mathcal{T}_{\mathcal{A}'} \models executable(\sigma') \wedge Goal(\sigma')$ iff there exists a ground situation σ of $\mathcal{T}_{\mathcal{A}}$ such that $\mathcal{T}_{\mathcal{A}} \models executable(\sigma) \wedge Goal(\sigma)$ and EXPAND[REWRITE[seq(σ')],$\Delta_{\mathcal{A}}$] = seq(σ), where $seq(do(\vec{a}, s)) = \vec{a}$.*

Proof Sketch: First, by construction of $\mathcal{T}_{\mathcal{A}'}$, $\mathcal{L}(\mathcal{T}_{\mathcal{A}}) \subset \mathcal{L}(\mathcal{T}_{\mathcal{A}'})$, and, for any action a in $\mathcal{T}_{\mathcal{A}}$, $\mathcal{T}_{\mathcal{A}'}$ contains the successor state and action precondition axioms of a in $\mathcal{T}_{\mathcal{A}}$. It follows that, for any term s which denotes a situation in the language of $\mathcal{T}_{\mathcal{A}}$, $\mathcal{T}_{\mathcal{A}} \models Goal(s) \wedge executable(s)$ iff $\mathcal{T}_{\mathcal{A}'} \models Goal(s) \wedge executable(s)$. Second, since in any situation s, the expansion of an executable complex action is also executable and has the same effects, for any executable complex plan $[a'_1, a'_2, ..., a'_n]$ in $\mathcal{T}_{\mathcal{A}'}$, EXPAND[REWRITE$[a'_1, a'_2, ..., a'_n]$,$\Delta_{\mathcal{A}}$]= $[a_1, a_2, ..., a_m]$ is an executable plan in $\mathcal{T}_{\mathcal{A}'}$, and $do([a'_1, a'_2, ..., a'_n], S_0)$ and $do([a_1, a_2, ..., a_m], S_0)$ are the same *states* in $\mathcal{T}_{\mathcal{A}'}$(i.e. fluents has the same truth value in both situations). Finally, by definition of the REWRITE and EXPAND steps, $a_1, a_2, ..., a_m$ are actions in $\mathcal{T}_{\mathcal{A}}$. It follows that $do([a_1, a_2, ..., a_m], S_0)$ is a term in the language of $\mathcal{T}_{\mathcal{A}}$ which denotes a situation, and thus $\mathcal{T}_{\mathcal{A}} \models Goal(do([a_1 ... a_m], S_0)) \wedge executable(do([a_1 ... a_m], S_0))$ if and only if $\mathcal{T}_{\mathcal{A}'} \models Goal(do([a'_1 ... a'_n], S_0)) \wedge executable(do([a'_1 ... a'_n], S_0))$.

Planning in $\mathcal{T}_{\mathcal{A}'}$ is sound and complete with respect to planning in $\mathcal{T}_{\mathcal{A}}$. Thus our approach to complex action planning via transformation of the theory is well-founded.

4.2 Exploiting Existing Operator-Based Planners

Our approach is not limited to planners realized in the situation calculus. Most popular planners don't use a successor state axioms representation of the effects of actions. E.g., all of the planners that participate in the AIPS Planning competition use PDDL as an initial specification of the action theory. In this section we show how to exploit an arbitrary operator-based planner that accepts PDDL planning domains with conditional effects [14], in order to plan with complex actions.

(1) COMPILE[$\mathcal{T}_{\mathcal{A}}$,$\Delta_{\mathcal{A}}$]: Rather than employing successor state axioms, PDDL describes the effects of actions in terms of (conditional) effects without a solution to the frame problem. Section 3.2 provides a semantic justification for an intuitive algorithm that compiles a PDDL representation of the preconditions and effects of actions in $\mathcal{T}_{\mathcal{A}}$, together with complex actions $\Delta_{\mathcal{A}}$ into a new PDDL representation of preconditions and effects in $\mathcal{T}_{\mathcal{A}'}$, without going through the intermediate stage of creating successor state axioms. (We have such an algorithm, but space precluded its inclusion in this paper.) Intuitively, the effects of a complex action are the effects of each action in the execution of δ, modulo subsequent effects.

(2) PLAN[$\mathcal{T}_{\mathcal{A}'}$,goal]: Given a compiled PDDL representation $\mathcal{T}_{\mathcal{A}'}$, we can generate a plan with any planner that accepts PDDL with conditional effects. (We used FF [9].)

(3) REWRITE & (4) EXPAND: We can use STEP (3)-(4) from Section 4.1. Alternatively, we can write a (fairly straightforward) algorithm to expand the final plan in $\mathcal{A}'$. For maximal efficiency, we would cache the conditions that uniquely determine the expansion of each complex action in a situation.

5 Elaborations on Complex Action Planning

In this section we examine elaborations on complex action planning. In particular, we examine the conditions underwhich adding complex actions to a theory causes other actions to be redundant and thus removable. Removing redundant actions is desirable because it reduces the plan search space. In an extended version of this paper, we discuss concurrency in complex action planning.

5.1 Removing Weaker Actions

When a complex action δ_1 is compiled into a primitive action theory as a new primitive action a_1, another primitive action, a_2 may become redundant in the sense that in any situation s, if a_2 is possible, a_1 is also possible and has exactly the same effects as a_2. More generally, we define the notion that primitive action a_1 is *stronger* than primitive action a_2, $a_1 \succeq a_2$ (and conversely that a_2 is weaker than a_1) as follows:

$$a_1 \succeq a_2 \leftrightarrow [\, Poss(a_2, s) \rightarrow Poss(a_1, s) \wedge SS(do(a_1, s), do(a_2, s))\,] \quad (19)$$

where $SS(s, s')$ is an abbreviation for the first-order formulae that is true iff situations s and s' have the same state. The relation $\succeq$ is a preorder (it is reflexive and transitive). It follows that for any situation calculus theory T and goal formula $G(s)$, $T \models \exists s.G(s)$ iff $T' \models \exists s.G(s)$, where T' is T with all weaker actions removed.

Note that removal of weaker primitive actions may result in removal of the optimal plan. In particular, if $a_1 \succeq a_2$ and a_1

is a compiled complex action that can expand into multiple primitive actions, then by removing a_2, we may lose the optimal plan with respect to the number of primitive actions in our initial domain. Also note that the notion of stronger actions does not capture all the conditions underwhich an action is redundant. In particular, a_2 may be conditionally redundant, or it might be redundant relative to a_1 in some situation, and redundant relative to a_3 in others.

Example: Let a_1 and a_2 be primitive actions in $\mathcal{T}_\mathcal{A}$, let $a_2$achieve the preconditions for a_1, and let $Poss(a_1)$ be the situation suppressed expression [16, pg.112] for $Poss(a_1, s)$. Define complex action δ_3 as **if** $\neg Poss(a_1)$ **then** a_2 **endIf** ; a_1. If we compile a_1, a_2 and δ_3 in $\mathcal{T}_\mathcal{A}$ into primitive actions a_1', a_2' and a_3' in $\mathcal{T}_{\mathcal{A}'}$, following Section 4.1, then it follows that $a_3' \succeq a_1'$.

5.2 Irrelevant Actions with Respect to a Goal

Let $G(s)$ be a goal predicate that is true iff s satisifies the goal formula. If the direct effect of an action a can never make $G(s)$ true, and if a cannot directly achieve the preconditions of any of the actions, then a is irrelevant with respect to goal predicate $G(s)$ and can be removed. Formally, given a primitive action a_1 and goal predicate G, we consider a_1 as G-irrelevant in $\mathcal{T}$ if and only if, for a_2 ranging over all actions in $\mathcal{T}$ except a_1, it follows from T that:

$$executable(do(a_1, s)) \leftrightarrow [\,(G(do(a_1, s)) \rightarrow G(s)) \wedge (Poss(a_2, do(a_1, s)) \rightarrow Poss(a_2, s))\,] \quad (20)$$

If a_1 is G-irrelevant, then a_1 will not be in any optimal successful plan to achieve G, and can be removed from the set of actions when planning to achieve G.

Example (continued): In the previous example, we showed that a_1' could be removed from $\mathcal{T}_{\mathcal{A}'}$. It then follows that, a_2' will never be needed to make a_1' $Poss$-ible. If a_2' can never directly achieve the preconditions for any other actions in the theory, then for all goal predicate $G(s)$ which are not among the effects of a_2', a_2' is G-irrelevant.

6 Web Service Composition

The primary motivation for our work was to be able to compose Web services using operator-based planning techniques. With the results of Sections 3 and 4, we have addressed a fundamental barrier to automated Web service composition. Service providers such as Amazon or United Airlines will describe their Web services (Web-accessible programs) as processes. In our vision of the Semantic Web, this will be done using the DAML+OIL Web service ontology, DAML-S [1], whose process description constructs are similar to Golog. (The relationship between DAML-S and the situation calculus is well-defined and has been used to define the semantics of DAML-S.) To produce black-box or compiled representations of Web services for automated composition, we can exploit the compilation techniques described in this paper. Using them, we compile process-oriented program descriptions of services into black-box component descriptions. Once Web services process descriptions have been compiled, we can use standard operator-based planning techniques to automatically compose Web services.

7 Efficiency of Complex Action Planning

A secondary motivation for our work was to potentially improve the efficiency of planning (e.g., [11, 8]) through our operator-based approach to complex action planning. We restrict our attention to complex action planning with the deterministic actions listed in Sect. 4. Compiling a complex action δ is polynomial in the number of primitive action occurrences in its definition. Note that this step can be performed offline, and is amortized over multiple planning runs. The expansion step is itself linear in the length of the plan, and in the number of branchings in the complex action definition. Of no surprise, plan generation dominates the computational cost. In particular: i) complex action operators tend to have more complex preconditions and effects than primitive actions, and ii) the size of the search space will be changed. However, i) causes only a linear slowdown and thus, the crucial point is ii).

Although the following analysis can be adapted to almost any classical planner, for simplicity of the argument, let's consider a breadth-first search forward planner. Given n ground actions, if the shortest successful plan is of length l, the size of the primitive action domain search space is $O(n^l)$. Adding d ground complex actions yields $n' = n+d$ ground actions in the compiled domain. [8] claims that adding actions that correspond to compositions of other actions will yield a larger search space. We identify conditions under which this is false.

Suppose the use of complex actions results in successful plans of length l/k, $k \geq 1$. One way to ensure this is by requiring complex actions to correspond to non-overlapping subplans in the shortest plan. In this case, the number of states visited to find a plan of length l/k will be $O(n'^{l/k})$ and the difference between the search spaces will be $O(n^l - n'^{l/k})$. If $(n^k - n')$ has a strictly positive lower bound for any problem, the new search space will be exponentially smaller than the old, as problem complexity increases. Informally, complex action planning reduces the planning search space when the complex actions significantly shorten the smallest successful plan relative to the increase they cause in the breadth of the search space.

Finally, in addition to this potential search space reduction,

some complex actions remove conflicts between the goals. This results in less backtrackings and enables the use of very efficient hill-climbing techniques (e.g., [9]).

8 Experimental Results

The techniques of Section 4.2 were implemented using the operator-based breadth-first search forward planner, FF [9]. FF supports conditional effects and uses its "enforced hill-climbing" whenever possible. We tested our approach on the ADL BRIEFCASE domain (BCD) [8]. This domain moves objects between locations using a briefcase. Three experiments were run on multiple instances of the problem, varying numbers of locations (#l) and portables (#p)[9].

The first experiment was simply BCD alone. Note that FF struggles as we increase (#p) and (#l). The next experiments involved the addition of the complex action Move-object. Move-object MO(locInit, locObj, Obj, locFinal) takes as input the location of the briefcase locInit, an object Obj, its location locObj, and a destination locFinal. It moves the briefcase to locObj, puts the object in the briefcase, moves the briefcase to locFinal, and removes Obj. MO is not a subplan of the shortest plan, so we would not necessarily expect it to do well. Further, it does *not* reduce the search space as it does not shorten the successful plan enough to compensate for the number of ground complex actions ($(n^k - n')$ is not positive). Nevertheless, adding the complex action move-object (BCD+MO) turns on FF's hill-climbing techniques, which reduce the number of nodes considered.

Finally, we designed a complex action that does correspond to subplans of the shortest successful plan and thus reduces the search space. The complex action LOC(loc-bc, loc), takes as input the location of the briefcase loc-bc, moves the briefcase to location loc, removes all the objects in the briefcase that should be at loc, and puts all other objects at loc in the briefcase. The goal defines where an object should be. To encode this complex action in PDDL, the action must know the goal at the time it executes. Hence, we added a persisting predicate $ShouldBeAt(Obj, Loc)$ to the domain. This predicate always has the same values as the $At(Obj, Loc)$ predicate in the goal statement. This complex action reduces the search space ($k \simeq \#p/\#l, d \simeq \#l$) and allows the use of hill-climbing techniques. Of no surprise, (BCD+LOC) presented the best results of all three experiment runs.

	#l:5, #p:20	#l:6, #p:30	#l:7, #p:42
BCD	5549 (1.39)	201006 (2261)	? (> 40h)
BCD+MO	859 (11.83)	2345 (201.47)	5195 (2211)
BCD+LOC	75 (.08)	139 (.27)	260 (.85)

Number of nodes (and time of run in seconds).

[8] http://rakaposhi.eas.asu.edu/domain-syntax.html

[9] Experiments run on Sun Sparc v9, 2×750GHz, 4GB of mem.

9 Discussion and Summary

The work in this paper was motivated by the problem of automating Web service composition. In particular, we posed the problem of composing Web services such as UAL's buyAirTicket($\vec{x}$) or CNN's getWeather($\vec{y}$) in order to achieve a user-defined goal. These Web services are describable as simple programs, using typical programming language constructs. We conceived this task as the problem of planning with complex actions, with the restriction that the complex actions *had* to be the primitive building blocks of a plan. Consequently, we posed the problem of how to represent and plan with complex actions, using operator-based planning techniques. To this end, we embarked upon a theoretical analysis of the problem of how to represent complex actions as operators. The situation calculus provided the formal foundation for our work, enabling us to provide a formal definition of the preconditions, successor state axioms, and effects of complex actions under a frame assumption.

With this representational problem addressed we turned to the practical matter of how to plan. We proposed a method of planning that produced sound and complete plans relative to a corresponding primitive action domain. We showed how to use our results to plan via deductive plan synthesis as well as using an arbitrary operator-based planning system that accepts ADL as input.

We are currently incorporating these representation and compilation results into DAML-S [1], an AI-inspired markup language ontology for Web services. We're also incorporating the results into ongoing Web service composition work [13].

Finally, the second motivation for this work was to potentially improve either the efficiency of planning or the quality of the plans generated, by exploiting complex actions that capture some preferred subplans. We have shown how, in some domains, using relevant complex actions will result in a dramatic speedup of the planning process. We discussed the impact of our approach on the planning search space and illustrated predicted speedup with experiments.

Acknowledgements

We would like to thank Srini Narayanan for conversations related to this work. We gratefully acknowledge the financial support of the US Defense Advanced Research Projects Agency DARPA Agent Markup Language (DAML) Program #F30602-00-2-0579-P00001.

References

[1] A. Ankolekar, M. Burstein, J. Hobbs, O. Lassila, D. Martin, S. McIlraith, S. Narayanan, M. Paolucci, T. Payne, K. Sycara, and H. Zeng. Daml-s: Semantic markup for web services. In *Proc. International Semantic Web Working Symposium (SWWS)*, 2001.

[2] M. Baioletti, S. Marcugini, and A. Milani. Encoding planning constraints into partial order planning domains. In *Proc. 6th Conference on Knowledge Representation and Reasoning*, pages 608 – 616, 1998.

[3] G. De Giacomo, Y. Lespérance, and H. Levesque. *ConGolog*, a concurrent programming language based on the situation calculus. *Artificial Intelligence*, 121(1-2):109–169, 2000.

[4] P. Doherty and J. Kvarnstrom. TALplanner: A temporal logic based forward chaining planner. *Annals of Mathematics and Artificial Intelligence*, 30:119–169, 2001.

[5] K. Erol, J. Hendler, and D. Nau. HTN planning: Complexity and expressivity. In *Proc. of the Twelfth National Conference on Artificial Intelligence (AAAI-94)*, volume 2, pages 1123–1128, 1994.

[6] R.E. Fikes, P.E Hart, and N.J Nilsson. Learning and executing generalized robot plans. *Artificial Intelligence*, 3:251–288, 1972.

[7] C. C. Green. Theorem proving by resolution as a basis for question-answering systems. In B. Meltzer and D. Michie, editors, *Machine Intelligence 4*, pages 183–205. American Elsevier, New York, 1969.

[8] Patrik Haslum and Peter Jonsson. Planning with reduced operator sets. In *Artificial Intelligence Planning Systems*, pages 150–158, 2000.

[9] Jörg Hoffmann and Bernhard Nebel. The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, 14:253–302, 2001.

[10] S. Hölldobler and H.-P. Störr. BDD-based reasoning in the fluent calculus – first results. In *8th. Intl. Workshop on Non-Monotonic Reasoning (NMR'2000)*, 2000.

[11] R. E. Korf. Planning as search: A quantitative approach. *Artificial Intelligence*, 33(1):65–88, 1987.

[12] H. Levesque, R. Reiter, Y. Lesperance, F. Lin, and R. Scherl. GOLOG: A logic programming language for dynamic domains. *Journal of Logic Programming*, 31(1-3):59–84, April-June 1997.

[13] S. McIlraith, T. Son, and H. Zeng. Semantic Web services. In *IEEE Intelligent Systems (Special Issue on the Semantic Web)*, March/April 2001.

[14] E. Pednault. ADL: Exploring the middle ground between STRIPS and the situation calculus. In *Proc. KR'89*, pages 324–332, 1989.

[15] R. Reiter. The frame problem in the situation calculus: A soundness and completeness result, with an application to database updates. In *Proceedings First International Conference on AI Planning Systems*, College Park, Maryland, June 1992.

[16] R. Reiter. *KNOWLEDGE IN ACTION: Logical Foundations for Specifying and Implementing Dynamical Systems*. The MIT Press, 2001.

[17] E. Sacerdoti. Planning in a hierarchy of abstraction spaces. *Artificial Intelligence*, 5:115–135, 1974.

[18] T. C. Son, C. Baral, and S. McIlraith. Planning with different forms of domain-dependent control knowledge – an answer set programming approach. In *6th International Conference on Logic Programming and Nonmonotonic Reasoning*, Programming Approach Proceedings, pages 226–239, 2001.

[19] R. J. Waldinger. Achieving several goals simultaneously. In E. W. Elcock and D. Michie, editors, *Machine Intelligence 8: Machine Representations of Knowledge*, pages 94–136. Ellis Horwood, Chichester, 1977.

The Mutual Exclusion Problem in Reasoning about Action and Change

Jan Broersen
Faculty of Sciences
Vrije Universiteit Amsterdam
De Boelelaan 1081a
1081 HV Amsterdam

John-Jules Meyer
Department of Computer Science
Universiteit Utrecht
Padualaan 14
3584 CH Utrecht

Roel Wieringa
Faculty of Computer Science
University of Twente
P.O. Box 217
7500 AE Enschede

Abstract

We describe a new aspect of the qualification problem that arises in the context of concurrent action. The qualification problem is typically approached by formulating principles of maximal qualification. However, for concurrent action the qualification problem is exponentionally harder, because we have to specify all mutual exclusions explicitly. We identify the problem encountered as 'the mutual exclusion problem'. We argue that its solution may involve a default position regarding the qualification of concurrent actions relative to the qualification of constituent concurrent parts of an action. We describe the mutual exclusion problem in detail and propose a possible solution in terms of intended modal action models.

1 Introduction

Concurrency embodies a new challenge for the area of reasoning about action and change. In relation to the frame problem, concurrency has been studied quite intensively in the last decade [7, 12, 3]. The central question in these studies is how to define the relation between effects of concurrent actions and effects of their constituent parts, and how to do this under a condition of minimal change. Effects of constituent concurrent action parts might (1) add up unless overridden by explicitly specified concurrent effects [5], or (2) add up and lead to non-determinism in case of conflict [3], or (3) add up and lead to non-qualification in case of conflict [12].

But in relation to the qualification problem [15, 8, 13], to our knowledge no studies have yet been conducted. It might be that the overall opinion is that the non-concurrent solutions for the qualification problem generalize smoothly to the concurrent case. However, in this paper we observe that the standard semantic solution to the qualification problem, that is, to maximize qualifications and specify only the conditions under which actions cannot take place (in the form of necessary preconditions), still requires the specification of an unreasonable amount of (non-)qualification information in the case of concurrent action. For each concurrent action that is a concurrent composition of atomic actions appearing in an action description, we have to specify explicitly which conditions prevent its possible occurrence. We argue that this problem calls for a new description default in the semantics of action descriptions: one that maximizes concurrent qualifications relative to qualifications of concurrent constituent parts. What we mean by this, is explained in detail in section 3 of the paper. But first, in section 2, we introduce the modal action description language (ADL) we work with. Sections 4, 5 and 6 respectively describe the semantic solutions for the frame problem, the qualification problem and the mutual exclusion problem for the ADL defined in section 2. Section 7 discusses that the orderings associated with the three problems are orthogonal, which implies that they can be freely combined. Section 8 gives the conclusions.

2 A modal description language for concurrent action

Given a set $\mathcal{A}$ of action symbols, and a set $\mathcal{P}$ of proposition symbols, the syntax of an action description formula φ is defined as:

α	$::=$	$a \mid -\alpha \mid \alpha \cup \alpha' \mid \alpha \cap \alpha'$	a ranging over $\mathcal{A}$
φ	$::=$	$P \mid \neg\varphi \mid \varphi \wedge \varphi' \mid [\alpha]\varphi$	P ranging over $\mathcal{P}$ and no nesting of the operator $[\alpha](.)$

An atomic proposition $p \in \mathcal{P}$ corresponds to what by other authors is sometimes called a *fluent*. A set of action description formulas of the form φ is called an 'action description', and is typically denoted by AD. We assume all the customary syntactic abbreviations, such as $\neg[\alpha]\neg\varphi \equiv_{def} \langle\alpha\rangle\varphi$. The restriction to formulas without nestings of modalities is not strong for an action description language. Action description formulas typically have the form $\varphi \rightarrow [\alpha]\varphi'$ with intended meaning: *φ is a condition under which performing an action*

α *brings about* φ'. This form subsumes the expression of *necessary preconditions*, which are expressed by $\neg\varphi \rightarrow [\alpha]\bot$, with intuitive meaning: *action* α *can only take place if the condition* φ *is satisfied.* The action connectives $\cup$, $\cap$ and $-$ stand for non-deterministic choice, concurrent composition and action complement respectively. We use the standard modal semantics for the ADL. A modal action model $\mathcal{M} = (S, R^{\mathcal{A}}, V)$ is defined as:

- S is a nonempty set of possible states
- $R^{\mathcal{A}}$ is an action interpretation function $R^{\mathcal{A}} : \mathcal{A} \rightarrow 2^{(S \times S)}$, assigning a binary relation over $S \times S$ to each atomic action a in $\mathcal{A}$.
- V is a valuation function $V : \mathcal{P} \rightarrow 2^S$ assigning to each proposition P of $\mathcal{P}$ the subset of states in S for which P is valid.

The *standard semantics* of ADL-formulas is determined by the notion of validity of a formula φ in a state s of a model $\mathcal{M}$, denoted $\mathcal{M}, s \models \varphi$. Compound actions are interpreted by extending the interpretation function $R^{\mathcal{A}}$ to a function R:

$$
\begin{array}{lll}
R(\alpha) & = & R^{\mathcal{A}}(\alpha) \text{ for } \alpha \in \mathcal{A} \\
R(\alpha \cup \alpha') & = & R(\alpha) \cup R(\alpha') \\
R(\alpha \cap \alpha') & = & R(\alpha) \cap R(\alpha') \\
R(-\alpha) & = & (\bigcup_{a \in \mathcal{A}} R(a)) \setminus R(\alpha) \\
\mathcal{M}, s \models P & \text{iff} & s \in V(P) \\
\mathcal{M}, s \models \neg\varphi & \text{iff} & \text{not } \mathcal{M}, s \models \varphi \\
\mathcal{M}, s \models \varphi \wedge \varphi & \text{iff} & \mathcal{M}, s \models \varphi \text{ and } \mathcal{S}, s \models \varphi' \\
\mathcal{M}, s \models [\alpha]\varphi' & \text{iff} & \text{for all } s', (s, s') \in R(\alpha) \text{ implies } \mathcal{M}, s' \models \varphi
\end{array}
$$

Validity on a model ($\mathcal{M} \models \varphi$), general validity and standard semantic entailment ($\Sigma \models \psi$, with Σ a set of action description formulas), are defined as usual. According to the definition for modal languages given by van Benthem [16], it is not justified to call this action description language 'modal', because it does not preserve validity under the standard notion of bisimulation. This is due to the relational intersection operation $\cap$ and the action negation $-$, that prevent that the standard modal logic tree property is obeyed. Yet we refer to the language as 'modal' because of its modal-type truth conditions.

As argued elsewhere [4], for the definition of intended models for modal action languages, as solutions to the frame and qualification problem, it is important first to establish a semantic equivalence notion for the language. The second step is then to adapt, for both the frame problem and the qualification problem, the notion of semantic equivalence in a *minimal* way, such that it is turned into an ordering of modal action models. In case of the frame problem, the adaptation implements the criterion of minimal change. In case of the qualification problem, the adaptation implements maximal qualification (actions are possible, unless the action description contains explicit information to the contrary). Then the justification for calling the orderings that are obtained in this way *intuitively correct* is this: the fact that the resulting change ordering, as a solution to the frame problem, is a *minimal* adaptation of the notion of semantic equivalence ensures that the ordering compares models *only* on the aspect of minimal change, while *all other* information contained in models is kept invariant modulo the expressive power of the modal action logic. And the fact that the resulting qualification ordering, as a solution to the qualification problem, is a *minimal* adaptation of the notion of semantic equivalence ensures that the ordering compares models *only* on the aspect of action qualification (possibility) in certain states. This strategy of adapting modal semantic equivalence notions in a minimal way thus provides a general approach to the problem of anomalous models for reasoning about action and change. If we can be sure that the orderings compare models *exclusively* on the relevant aspects (minimal change, maximal qualification, etc.), no unintended properties for intended models can sneak into the minimization (or maximization) strategy, and thus, no anomalous intended models can result.

For modal action logics with intersection in general, we need the notion of a 'well-nested graph' [6] in order to define a semantic equivalence notion. But for this relatively simple action language without sequence or iteration, the notion of well nested graph reduces to that of a 'step', which is a set of atomic actions performed concurrently.

Definition 1 (step relatedness) *Given a model* $\mathcal{M} = (S, R^{\mathcal{A}}, V)$, *and a set of atomic actions* T. *Now it holds that states* s *and* t *are* T*-related relative to* $\mathcal{M}$, *notation '*sTt *relative to* $\mathcal{M}$*' iff* $T = \{a \mid (s, t) \in R^{\mathcal{A}}(a)\}$.

Now we can define semantic equivalence.

Definition 2 (total surjective step-1-bisimulation) *Let* $\mathcal{M}_1 = (S_1, R_1^{\mathcal{A}}, V_1)$ *and* $\mathcal{M}_2 = (S_2, R_2^{\mathcal{A}}, V_2)$ *be two modal action models. Then* $\mathcal{M}_1 \simeq \mathcal{M}_2$ *if and only if there is a total surjective relation* $H \subseteq S_1 \times S_2$, *such that for all* s_1 *and* s_2 *such that* $(s_1, s_2) \in H$ *it holds that:*

1. $s_1 \in V_1(P)$ *iff* $s_2 \in V_2(P)$ *for all* P
2. *if there is a* t_1 *such that for some step* T *it holds that* $s_1 T t_1$ *relative to* $\mathcal{M}_1$, *then there is a* t_2 *such that* $s_2 T t_2$ *relative to* $\mathcal{M}_2$ *and* $t_1 \in V_1(P)$ *iff* $t_2 \in V_2(P)$ *for all* P
3. *if there is a* t_2 *such that for some step* T *it holds that* $s_2 T t_2$ *relative to* $\mathcal{M}_2$, *then there is a* t_1 *such that* $s_1 T t_1$ *relative to* $\mathcal{M}_1$ *and* $t_2 \in V_2(P)$ *iff* $t_1 \in V_1(P)$ *for all* P

Theorem 1 *$\mathcal{M}_1 \simeq \mathcal{M}_2$ if and only if for all formulas φ of the ADL it holds that $\mathcal{M}_1 \models \varphi$ iff $\mathcal{M}_2 \models \varphi$*

3 The mutual exclusion problem

The mutual exclusion problem, that we describe in this section, is strongly related to the standard qualification problem. The qualification problem reads as follows:

The qualification problem: when specifying under which conditions actions are possible, we do not want to involve ourselves in describing *exhaustively* what circumstances prevent the possibility of an action.

We may describe the qualification problem as the problem how to avoid having to specify sufficient preconditions for actions. That φ is a sufficient precondition for the possibility of an action α is typically expressed as $\psi \rightarrow \langle\alpha\rangle\top$. But obtaining correct information of this type, i.e. establishing conditions φ for a specific action domain, is usually very hard for two reasons. First of all, when making action descriptions, it is very difficult to foresee all conditions that prevent the possibility of an action. A standard example is from Mc Carthy [13] who describes what conditions may prevent the possibility of a boat crossing a river. It may be leak, the weather may be bad, the peddles may be broken, etc. But the difficulty may also come from 'inside' an action description. It may for instance be the case that the specifier of an action domain does not notice that effect information and static information, specified in separate formulas of an action description, conspire to (logically) imply that α cannot take place under the condition ψ. Then addition of a formula $\psi \rightarrow \langle\alpha\rangle\top$ results in an inconsistent action description.

A semantic solution is provided by the principle of maximal qualification that provides a description default with respect to qualification information: an action is possible in a certain situation whenever it does *not* follow from the action description that it is *not* possible. So, under an intended maximal qualification interpretation of action descriptions, we may only provide necessary preconditions for the possibility of actions. We call this description default 'maximal qualification'. Such a description default is also assumed in the languages $\mathcal{A}$ and $\mathcal{C}$ developed by Gelfond and Lifschitz [7], where it is called 'qualification completeness'.

In case the action description language enables the description of concurrent action the qualification problem becomes considerably harder. This is because the number of possible different concurrent actions grows exponentially with the number of atomic actions used for action description. This means that for concurrent actions the qualification problem is exponentially intensified: for any possible concurrent composition of actions we have to decide on the qualification information when making an action description.

Partially, this task is facilitated by the qualification variant of the add-up principle for concurrent actions. That is, in the ADL we defined, it is the case that if φ is a necessary precondition for a, then it is also a necessary precondition for $a \cap b$. This follows from the standard modal semantics we gave for the ADL: the reader may check that $\neg\varphi \rightarrow [a]\bot \models \neg\varphi \rightarrow [a \cap b]\bot$ is a valid global entailment relation. Yet, this only gives a partial facilitation of the problem. For all concurrent subsets that do not subsume one another, we still have to decide on the qualification. So we still may have to provide unacceptably many formulas of the form $\neg\varphi \rightarrow [a_1 \cap a_2 \cap \ldots \cap a_n]\bot$. Such formulas say that in states where $\neg\varphi$ holds it is not possible to perform the atomic actions $a_1 \cap a_2 \cap \ldots \cap a_n$ concurrently. Following standard terminology from concurrency theory, we refer to such formulas as 'mutual exclusions'. The problem of having to specify all mutual exclusions is thus 'caused' by the principle of maximal qualification in combination with the acceptance of concurrent actions in the description language. We argue that this problem of having to specify all mutual exclusions is an important problem in itself, that requires a solution independent from the qualification problem.

The mutual exclusion problem: when specifying mutual exclusions for concurrent actions, we do not want to involve ourselves in describing *exhaustively* together with which other actions a given action may or may not occur concurrently.

We argue that preferably one would like to assume a description default with respect to the issue raised by the mutual exclusion problem, a default of the same type as the minimal change default for the frame problem and the maximal qualification default for the qualification problem. We propose the following description default: if actions are both possible concurrently and in isolation, we only assume the concurrent possibility, at the cost of the isolated possibilities. That is, we want to maximize concurrency, or equivalently, minimize mutual exclusion, and only specify the exceptions as explicit mutual exclusions.

One might argue that the mutual exclusion problem is simply the concurrent form of the qualification problem, since both problems deal with possible occurrence: maximal reachability with possible occurrence in general, and maximal concurrency with possible occurrence of concurrent actions. But this misses the point. Maximal concurrency deals with the problem whether isolated and concurrent execution of an action can both be qualified or that only one of the two is (preferably) qualified, and it decides this choice in favor of concurrent executions. Thus, maximal concurrency is not a description default about individual actions as such, but about mutual concurrent in- or exclusions of actions. Maximal reachability maximizes all possible occurrences, either isolated or concurrent.

The problem of wanting to engage in a description default with respect to possibilities of concurrent actions relative to constituent concurrent parts also arises in the context of the semantics of statecharts, a graphical specification formalism for reactive systems [1, 14, 9]. A basic problem in assigning intuitive semantics to statecharts is how to deal with the concurrent executions of parallel components of the statechart. Often a criterion of 'maximal steps' or 'maximal parallelism' is assumed ([11]). Another argument for adopting this particular description default is related to the problem of the state space explosion that is encountered when we try to apply model checking techniques to the verification of concurrent system specifications [10]. The state space explosion is a name for the problem that the number of concurrent actions grows exponentially with the number of atomic actions. Now the state space is most drastically reduced if we adopt a description default of maximal concurrency. It is perceivable that by doing so, we can derive interesting properties of the concurrent system that would otherwise haven taken too much computation time to calculate.

4 Minimal change

In this section we define minimal change models. We do not elaborate on the exact details of the change ordering we define here. Some background information is given elsewhere [4]. As argued, the most important aspect of the ordering is that it is obtained as a minimal adaptation of the semantic equivalence notion for the ADL, as given in definition 2. First we need a notion of valuation change, which is the set of atomic propositions that changes value when going from one state to the other.

Definition 3 (valuation change) *The change $\delta(s_1, s_2)$ between two states, relative to a valuation V of propositional atoms $\mathcal{P}$, is defined as the set of propositions $\{P \mid (s_1 \in V(P)$ and $s_2 \notin V(P))$ or $(s_1 \notin V(P)$ and $s_2 \in V(P))\}$.*

Now we can add the change condition to the equivalence notion, thereby forming a change ordering $\sqsubseteq^{ch}$ ('ch' for 'change') of modal action models.

Definition 4 (change ordering) *Let $\mathcal{M}_1 = (S_1, R_1^A, V_1)$ and $\mathcal{M}_2 = (S_2, R_2^A, V_2)$ be two modal action models. Then $\mathcal{M}_1 \sqsubseteq^{ch} \mathcal{M}_2$ if and only if there is a total surjective relation $H \subseteq S_1 \times S_2$, such that for all s_1 and s_2 such that $(s_1, s_2) \in H$ it holds that:*

1. *$s_1 \in V_1(P)$ iff $s_2 \in V_2(P)$ for all P*
2. *if there is a t_1 such that for some step T it holds that $s_1 T t_1$ relative to $\mathcal{M}_1$, then there is a t_2 such that $s_2 T t_2$ relative to $\mathcal{M}_2$ and $\delta(s_1, t_1) \subseteq \delta(s_2, t_2)$*
3. *if there is a t_2 such for some step T it holds that $s_2 T t_2$ relative to $\mathcal{M}_2$, then there is a t_1 such that $s_1 T t_1$ relative to $\mathcal{M}_1$ and $\delta(s_2, t_2) \supseteq \delta(s_1, t_1)$*

We define $\mathcal{M} \sqsubset^{ch} \mathcal{M}'$ as $\mathcal{M} \sqsubset^{ch} \mathcal{M}'$ and $\mathcal{M}' \not\sqsubset^{ch} \mathcal{M}$, and $\mathcal{M} \equiv^{ch} \mathcal{M}'$ as $\mathcal{M} \sqsubseteq^{ch} \mathcal{M}'$ and $\mathcal{M}' \sqsubseteq^{ch} \mathcal{M}$. Then, $\equiv^{ch}$-equivalence classes are sets of models that are $\equiv^{ch}$-equivalent. If $\mathcal{M}$ is a model, then $|\mathcal{M}|^{ch}$ is its $\equiv^{ch}$-equivalence class. The $\sqsubseteq^{ch}$-ordering is a pre-order on models: reflexifivity and transitivity are immediate from the definition. We extend this pre-order on models to a partial order over the $\equiv^{ch}$-equivalence classes in the standard way. Anti-symmetry of this extended $\sqsubseteq^{ch}$-ordering, follows directly from the extension: if $|\mathcal{M}|^{ch} \sqsubseteq^{ch} |\mathcal{M}'|^{ch}$ and $|\mathcal{M}'|^{ch} \sqsubseteq^{ch} |\mathcal{M}|^{ch}$, then $|\mathcal{M}'|^{ch} = |\mathcal{M}|^{ch}$. Now the following proposition says that semantic equivalent models are indistinguishable in the $\equiv^{ch}$-ordering.

Proposition 2 *For any two models $\mathcal{M}$ and $\mathcal{M}'$ it holds that $\mathcal{M} \simeq \mathcal{M}'$ implies $\mathcal{M} \equiv^{ch} \mathcal{M}'$.*

This follows from the structural correspondence of the definitions for the $\sqsubseteq^{ch}$-ordering and $\simeq$-equivalence. When we replace the $\subseteq$ and $\supseteq$ symbols in the second and third condition of definition 4 by the equal sign '=', we arrive exactly at an alternative formulation of the semantic equivalence notion of definition 2. This proofs the proposition, since absence of change is encompassed by the expressions $\delta(s_1, t_1) \subseteq \delta(s_2, t_2)$ and $\delta(s_2, t_2) \supseteq \delta(s_1, t_1)$. This shows that the change ordering stays very close to the equivalence notion. But the property is important also for another reason. It implies that models that are not equivalent in the $\equiv^{ch}$-ordering, i.e. models that have different change properties, are not semantic equivalent. And since from proposition 1 it follows that semantic inequivalence implies modal inequivalence[1], we have that the models can be distinguished by formulas of the ADL. This means that we do not have to increase expressiveness of the ADL to express persistency properties: if models differ in the change ordering, there must be formulas in the ADL (that corresponds to the semantic equivalence relation the ordering is based on) that separates them.

Definition 5 *If we use the notation $Int([[AD]])$ to denote the set of intended models, and $[[AD]]$ to denote the set of classical models of an action description AD, we call a set of completion formulas C* intention correct *if and only if $Int([[AD]]) \subseteq [[AD \cup C]]$, and* intention sufficient *if and only if $[[AD \cup C]] \subseteq Int([[AD]])$.*

Theorem 3 *For any finite action description AD in an action description language for which $\simeq$-equivalence holds, there is a finite set F of conditional frame formulas of the form $\psi \wedge \varphi \rightarrow [\alpha]\varphi$, that is intention correct and sufficient with respect to the $\sqsubseteq^{ch}$-minimal models of AD.*

The minimal elements in the ordering define an 'intended' minimal change semantics of action descrip-

[1] For standard bisimulation, this relation between semantic equivalence and modal equivalence is known as the Hennessy-Milner property [2].

tions. The minimal change semantics of an action description AD is the set of $\sqsubseteq^{ch}$-minimal models of AD. Preferential entailment is defined as follows.

Definition 6 (preferential entailment) *An action description AD preferentially entails ψ under minimal change, notation $AD \models^{ch} \psi$, if and only if all $\sqsubseteq^{ch}$-minimal valid models for AD are also valid models for ψ.*

5 Maximal qualification

For the concept of maximal qualification we follow the same approach as for minimal change: we take the equivalence notion for the ADL and adapt it such that it compares models exclusively on the aspect of qualification. For minimal change, we did not want to compare models on action possibilities, which meant that we needed to leave the back and forth structure of the equivalence notion in tact, and only focus on the change of valuations in states.

For the concept of maximal qualification we follow the same approach as for minimal change: we take the equivalence notion for a class of logics and adapt it such that it compares models on the aspect of qualification. For minimal change, we did not want to compare models on action possibilities, which meant that it was necessary to leave the back and forth structure of the equivalence notion in tact, and only focus on the change of valuations in states. For maximal qualification we have the opposite. We want to compare models only on the possibility to do certain actions in states. This means we have to eliminate either the forth or back clause in equivalence notions, such that action possibilities are preserved only in one 'direction'. Furthermore, we do not have to look at valuations in states reached by actions, since we are only concerned with possibility of action, not with effects.

We start with the equivalence notion of definition 2. Elimination of the 'back' condition and the comparison of the postconditions of actions in terms of valuations of atomic propositions, results in the qualification ordering $\sqsubseteq^{ql}$ ('ql' for 'qualification') for non-sequential actions.

Definition 7 (qualification ordering) *Let $\mathcal{M}_1 = (S_1, R_1^A, V_1)$ and $\mathcal{M}_2 = (S_2, R_2^A, V_2)$ be two models. Then $\mathcal{M}_1 \sqsubseteq_1^{ql} \mathcal{M}_2$ if and only if there is a total surjective relation $H \subseteq S_1 \times S_2$, such that for all s_1 and s_2 such that $(s_1, s_2) \in H$ it holds that:*

1. *$s_1 \in V_1(P)$ if and only if $s_2 \in V_2(P)$ for all P*
2. *if there is a t_1 such that for some step T it holds that $s_1 T t_1$ relative to $\mathcal{M}_1$, then there is a t_2 such that $s_2 T t_2$ relative to $\mathcal{M}_2$*

Analogous to the change ordering, we define $\mathcal{M} \equiv^{ql} \mathcal{M}'$ as $\mathcal{M} \sqsubseteq^{ql} \mathcal{M}'$ and $\mathcal{M}' \sqsubseteq^{ql} \mathcal{M}$, and $\equiv^{ql}$-equivalence classes as sets of models that are $\equiv^{ql}$-equivalent. If $\mathcal{M}$ is a model, then $|\mathcal{M}|^{ql}$ is its $\equiv^{ql}$-equivalence class. The $\sqsubseteq^{ql}$-ordering is a pre-order on models: reflexifivity and transitivity are immediate from the definition. As for the change ordering, we extend this pre-order on models to a partial order over the $\equiv^{ql}$-equivalence classes in the standard way. Preferential entailment is defined as in definition 6. The following proposition says that semantic equivalent models are indistinguishable in the $\equiv^{ql}$-ordering.

Proposition 4 *For any two models $\mathcal{M}$ and $\mathcal{M}'$ it holds that $\mathcal{M} \simeq \mathcal{M}'$ implies $\mathcal{M} \equiv^{ql} \mathcal{M}'$.*

This follows directly from the structural correspondence of the $\sqsubseteq^{ql}$-ordering of definition 7 with $\simeq$-equivalence of definition 2. So, models that are not equivalent in the $\equiv^{ql}$-ordering, i.e. models that have different qualification properties, are not semantic equivalent. Then, it follows from proposition 1 that the models can be distinguished by formulas of the ADL. The following theorem states that we can always define a completion of a finite action description AD that selects exactly the models maximal under the $\sqsubseteq^{ql}$-ordering.

Theorem 5 *For any finite action description AD in an action description language for which $\simeq$-equivalence holds, there is a finite set Q of precondition formulas of the form $\psi \rightarrow \langle\alpha\rangle\top$, that is intention correct and sufficient with respect to the $\sqsubseteq^{ql}$-maximal models of AD.*

6 Minimal mutual exclusion

In section 3 we introduced the mutual exclusion problem. We adapt the equivalence notion of definition 2 such that it compares models on the aspect of mutual exclusions of actions. However, the resulting ordering is not called 'mutual exclusion ordering', but 'concurrency ordering', because the notion of 'minimal mutual exclusion' is clearer the equivalent 'maximal concurrency'. The forth condition of definition 2 is changed such that steps of actions that are qualified in 'lower' models are either (1) also qualified in 'higher' models, or (2) are concurrently subsumed by steps that are qualified in 'higher' models. The back condition guarantees that (concurrent) actions that are qualified in higher models are either (1) qualified in lower models, or are a concurrent composition of actions that are qualified in lower models.

Definition 8 (concurrency ordering) *Let $\mathcal{M}_1 = (S_1, R_1^A, V_1)$ and $\mathcal{M}_2 = (S_2, R_2^A, V_2)$ be two models. Then $\mathcal{M}_1 \sqsubseteq^{cr} \mathcal{M}_2$ if and only if there is a total surjective relation $H \subseteq S_1 \times S_2$, such that for all s_1 and s_2 such that $(s_1, s_2) \in H$ it holds that:*

1. *$s_1 \in V_1(P)$ if and only if $s_2 \in V_2(P)$ for all P*
2. *if there is a t_1 such that for some step T it holds that $s_1 T t_1$ relative to $\mathcal{M}_1$, then there is a t_2 such*

that for some step T' it holds that $T \subseteq T'$, and $s_2 T' t_2$ relative to $\mathcal{M}_2$

3. *if there is a t_2 such that for some step T' it holds that $s_2 T' t_2$ relative to $\mathcal{M}_2$, then there are states $t_1^1, t_1^2, \ldots, t_1^n$ and there are steps $T^1, T^2, \ldots, T^n$ such that $s_1 T^1 t_1^1, s_1 T^2 t_1^2, \ldots, s_1 T^n t_1^n$ relative to $\mathcal{M}_1$, and $T' = T^1 \cup T^2 \cup \ldots \cup T^n$*

Metaphorically speaking, going up in the mutual exclusion ordering, actions look around for concurrent partners. When an action has found a partner, it sticks to it in the models higher up in the ordering. And concurrent parts of actions in models higher in the ordering always originate from isolated actions in lower models. To give some evidence that this ordering does what it promises to do, we consider two examples where models are compared under the $\sqsubseteq^{cr}$-ordering.

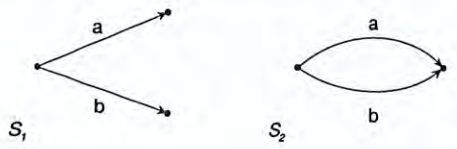


Fig 1. models for which $\mathcal{M}_1 \sqsubset^{cr} \mathcal{M}_2$

Assuming that the relation H relates corresponding positions in the models, it follows that $\mathcal{M}_1 \sqsubseteq^{cr} \mathcal{M}_2$ and $\mathcal{M}_2 \not\sqsubseteq^{cr} \mathcal{M}_1$. Thus, if two different actions performed in isolation can also be performed concurrently, models with the concurrent possibility are preferred.

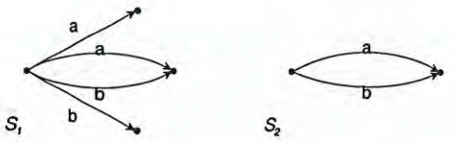


Fig 2. models for which $\mathcal{M}_1 \sqsubset^{cr} \mathcal{M}_2$

Again it holds that $\mathcal{M}_1 \sqsubseteq^{cr} \mathcal{M}_2$ and $\mathcal{M}_2 \not\sqsubseteq^{cr} \mathcal{M}_1$. Thus, situations where actions can only be performed concurrently are favored over situations where there is a choice to perform actions concurrently or in isolation. The examples make it clear that whenever an action description specifies compatible postconditions for two actions, in $\sqsubseteq^{cr}$-maximal models for the description they are only possible concurrently.

The $\sqsubseteq^{cr}$-ordering is reflexive and transitive. In the same way as for the other orderings, we may define $\equiv^{cr}$-equivalence classes and a generalized partial $\sqsubseteq^{cr}$-ordering over these classes. From the construction of the $\sqsubseteq^{cr}$-ordering as an adaptation of $\simeq$-equivalence, it is not difficult to conclude that $\mathcal{M} \simeq \mathcal{M}'$ if and only if models $\mathcal{M}$ and $\mathcal{M}'$ belong to the same $\equiv^{cr}$-equivalence class. The minimal mutual exclusion semantics of an action description AD is the set of $\sqsubseteq^{cr}$-maximal models of AD. Preferential entailment under minimal mutual exclusion is defined analogous to preferential entailment under minimal change, and maximal qualification.

That the mutual exclusion problem is more then the just the qualification problem for concurrent action follows from the observation that we cannot use the completion formulas $\psi \rightarrow \langle\alpha\rangle\top$ to define intention correct and sufficient completions. If for instance we want to enforce that in the situations where ψ, atomic actions a and b are only possible concurrent but not in isolation, it is not sufficient to specify $\psi \rightarrow \langle a \cap b\rangle\top$: this formula ensures that action $a \cap b$ is possible, but does not exclude that a and b are not also possible in isolation. But we also cannot add $\psi \rightarrow \langle a\rangle\bot$ and $\psi \rightarrow \langle b\rangle\bot$, since then the action is not longer possible concurrently. What we need, to ensure that the action is only possible concurrently is the following formula $\psi \rightarrow [-(a \cap b)]\bot$. This formula excludes models where from states where ψ holds only an a, only a b or no a or b are possible: it only allows a and b concurrent. Here we see that we need the extra expressiveness of the action negation as defined in section 2 to define completions: standard modal expressiveness does not suffice.

7 Orthogonality of the problems

In this section we argue that the three mentioned problems, and the semantic solutions we proposed for them are orthogonal. That is, we can adopt each of the three discussed descriptions defaults independently.

Theorem 6 *Given an action description AD, and given that $MinCh([[AD]])$ is the set of $\sqsubseteq^{ch}$-minimal models, $MaxQl([[AD]])$ is the set of $\sqsubseteq^{ql}$-maximal models, and $MaxCr([[AD]])$ is the set of $\sqsubseteq^{cr}$-maximal models among the standard models $[[AD]]$ of an action description AD, it holds that $MaxCr(MaxQl(MinCh([[AD]]))) = MinCh(MaxCr((MaxQl([[AD]])))) = MaxQl(MinCh((MaxCr([[AD]]))))$.*

This follows from the that for each of the descriptions defaults we can define completions. No special conditions on the form of formulas in action descriptions were assumed. This means that for instance the models $MaxQl(MinCh([[AD]]))$ and $MinCh(MaxQl([[AD]]))$ are selected by exactly the same sets of completion formulas for minimal change and maximal qualification. Thus, the sets $MaxQl(MinCh([[AD]]))$ and $MinCh(MaxQl([[AD]]))$ are equal.

8 Conclusion

In this paper we argued that minimal mutual exclusion, like minimal change and maximal reachability, can be seen as a useful description default for action descriptions. We defined intended modal action models that obey the principle of maximal concurrency (= minimal

mutual exclusion), and argued that this default in for the interpretation of action descriptions is independent of the frame problem and the qualification problem.

Intuitiveness of the semantic solution was not demonstrated using specific reasoning examples, but was 'derived' from the closeness of the orderings to the semantic equivalence relation of the modal action description language. It was argued that this strategy provides a general approach to the problem of anomalous models for reasoning about action and change. Intuitiveness was not shown with the help of examples. In the planned extended version of this paper, we do discuss several specific examples, one of which is that of the 'bowl of soup' [12].

References

[1] M. von der Beeck. A comparison of Statecharts variants. In H. Langmaack, W.P. de Roever, and J. Vytopil, editors, *Formal Techniques in Real-Time and Fault-Tolerant Systems*, Lecture Notes in Computer Science 863, pages 128–148. Springer, 1994.

[2] Patrick Blackburn, Maarten de Rijke, and Yde Venema. *Modal Logic.* Cambridge tracts in theoretical computer science 53. Cambridge University Press, 2001.

[3] Sven-Erik Bornscheuer and Michael Thielscher. Representing concurrent actions and solving conflicts. *Journal of the Interest Group in Pure and Applied Logics (IGPL). Special issue of the ESPRIT project MEDLAR*, 4(3):355–368, 1996.

[4] J.M. Broersen, R.J. Wieringa, and J.-J.Ch. Meyer. A semantics for persistency in propositional dynamic logic. In John Lloyd, Veronica Dahl, Ulrich Furbach, Manfred Kerber, Kung-Kiu Lau, Catascia Palamidessi, Luis Moniz Pereira, Yehoshua Sagiv, and Peter J. Stuckey, editors, *Proceedings First International Conference on Computational Logic (CL2000)*, Lecture Notes in Artificial Intelligence 1861, pages 912–925. Springer, 2000.

[5] Michael Gelfond Chitta Baral. Reasoning about effects of concurrent actions. *Journal of Logic Programming*, 12:1–199, 1993.

[6] S. Danecki. Nondeterministic propositional dynamic logic with intersection is decidable. Lecture Notes in Computer science 208, pages 34–53, 1984.

[7] Michael Gelfond and Vladimir Lifschitz. Action languages. *Electronic Transaction on AI*, 16(3), 1998.

[8] Matthew L. Ginsberg and David E. Smith. Reasoning about action II: The qualification problem. *Artificial Intelligence*, 35:311–342, 1988.

[9] D. Harel, A. Pnueli, J.P. Schmidt, and R. Sherman. On the formal semantics of statecharts. In *Proceedings, Symposium on Logic in Computer Science*, pages 54–64. Computer Science Press, June 22–25 1987.

[10] David Harel, Orna Kupferman, and Moshe Y. Vardi. On the complexity of verifying concurrent transition systems. In Józef Winkowski Antoni W. Mazurkiewicz, editor, *CONCUR '97: Concurrency Theory, 8th International Conference, Warsaw, Poland, July 1-4, 1997, Proceedings*, Lecture Notes in Computer Science 1243, pages 258–272. Springer, 1997.

[11] C. Huizing, R. Gerth, and W.P. de Roever. Modeling Statecharts in a fully abstract way. In M. Dauchet and M. Nivat, editors, *13th Colloquium on Trees in Algebra and Programming (AAP'88)*, Lecture Notes in Computer Science 299, pages 271–294. Springer, 1988.

[12] Camilla Schwind Laura Giordano, Alberto Martelli. Dealing with concurrent actions in modal action logic. In Henri Prade, editor, *Proceedings ECAI98, 13th European Conference on Artificial Intelligence*, 1998.

[13] John McCarthy. Circumscription—a form of non-monotonic reasoning. *Artificial Intelligence*, 13:27–39, 1980.

[14] A. Pnueli and M. Shalev. What is in a step: on the semantics of statecharts. In T. Ito and A.R. Meyer, editors, *Theoretical Aspects of Computer Software*, Lecture Notes in Computer Science 526, pages 244–264. Springer, 1991.

[15] Michael Thielscher. The Qualification Problem: A solution to the problem of anomalous models. *Artificial Intelligence*, 131(1–2):1–37, 2001.

[16] Johan van Benthem. *Modal Correspondence Theory.* PhD thesis, Universiteit van Amsterdam, Instituut voor Logica en Grondslagenonderzoek van Exacte Wetenschappen, 1-148, 1976.

Intensionality and Circumscription

Dr. G. White
Department of Computer Science
Queen Mary, University of London
graham@dcs.qmul.ac.uk
http://www.dcs.qmul.ac.uk/~graham

1 Introduction

This paper examines the semantics of a group of logical techniques, used by the Artificial Intelligence community, and which are fundamental for circumscription-based approaches to the situation calculus. At first sight, this calculus appears to be an operationalised version of David Lewis' account of causal reasoning: there are items – here called 'situations' – which are analogous to Lewis' possible worlds, and one finds the situation which is the result of an action by minimising difference: that is, if an action is performed in a situation s, then the situation resulting from the action is that situation which, among all of those in which the effects of the action hold, is minimally different from s.

I will argue, however, that appearances deceive: that, whereas Lewis' theory is extensional (in the sense that equal possible worlds can always be substituted for each other without harm), the situation calculus is strongly intensional. Consequently, the underlying ontologies of the two theories are strikingly different. The culprit here is the way in which the situation calculus measures difference: whereas Lewis' measure is required to be extensional (that is, to depend only on the identities of the worlds involved), the measure used in the situation calculus depends strongly on the way in which the worlds are described: we can have two languages which are logically equivalent, but which, nevertheless, yield distinct difference measures, and which, consequently, give different results for the same problem when used in the situation calculus.

The situation calculus, then, is not logically innocuous: if it can be given any principled semantics, then that semantics must involve an extremely intensional ontology.

2 The Situation Calculus and an Example

2.1 The Situation Calculus

I shall describe the situation calculus rather generically; the phenomena that I am describing apply to a great number of versions of the situation calculus, and consequently a rather more abstract approach has the benefit of clarity.

Situations are described by *fluents*: a fluent is a linguistic item – typically a propositional atom – that can be true or false in a given situation. So there are two sides to fluents: on the one hand, since they can be assigned truth values, they have a propositional role. On the other hand, they are tied to a particular language, and, specifically to a particular vocabulary, in which they figure as atoms.

For brevity, I shall be making my arguments informally, in terms of finding situations which minimise change, and, correspondingly, in terms of the partial order on the sets of models given by inclusion of the sets of changed fluents: the arguments are easy enough to formulate in terms of circumscription, in the style of (Shanahan, 1997, pp. 49ff) or of (Lifschitz, 1994, p. 329).

2.2 The Example

Consider the following scenario:

> Two balls are linked by a string of length 1. Initially (at $t = 0$) they are lying next to one another on a table, with the string slack. At $t = 1$, one ball is raised to height 2; this tightens the string and raises the other ball to height 1.

Now let us consider formulations of this scenario in several different vocabularies. In all cases the balls will be b_1 and b_2, and the positions will simply be numbered 0, 1, 2, 3, starting from the table and working upwards, so that the two balls start off in position 0.

With some trivial work, we can show that all of our vocabularies are logically equivalent: any assertion made in one of them can be expressed without loss in any of the others. The constraints on the fluents are, likewise, equally easy to formulate in any vocabulary: they simply say that, if one of the balls is at position i, then the other ball must either be at positions $i-1$ (if this is positive) or $i+1$, with the string taut, or at i, with the string slack.

Because of the constraints, ball 2 must either end up at positions 1 or 3, with the string taut, or at 2, with the string slack. So there are basically three situations to consider; I will call them m_1, m_2, and m_3 in what follows. I tabulate the order relations induced by a selection of vocabularies in Table 1

3 Formal Results on Interpretation

As we can see from the tables, we appear to be able to obtain a wide range of partial orders merely by change of vocabulary. As the results of this section show, this variability is quite typical.

We first need the notion of an interpretation of one language in another. Notice that, even though our languages in the preceding section was ostensibly first-order, its variables only take on a finite range of values: we can, thus, for simplicity, assume that our languages are propositional and have countable sets of atoms. They will have the classical connectives, $\wedge$, $\vee$, $\rightarrow$, and $\neg$.

Definition 1. *Let $\mathfrak{L}$ and $\mathfrak{L}'$ be two propositional languages. An* interpretation, *α, of $\mathfrak{L}$ in $\mathfrak{L}'$ is a map from the atoms of $\mathfrak{L}$ to the propositions of $\mathfrak{L}'$; it may be extended to all of $\mathfrak{L}$ inductively.*

Suppose we have theories, τ and τ', formulated respectively in the languages $\mathfrak{L}$ and $\mathfrak{L}'$. We will identify theories with deductively closed sets of sentences.

Definition 2. *An interpretation α is* sound *(with respect to τ and τ') if, whenever T is a theorem of τ, $\alpha(T)$ is a theory of τ'; it is* conservative *(again, with respect to τ and τ') if, given a proposition p of $\mathfrak{L}$, if $\alpha(p)$ is a theorem of τ', then p is a theorem of τ. α is* essentially surjective *(with respect to τ') if, for all $q \in \mathfrak{L}'$, there is a $p \in \mathfrak{L}$ such that $\tau \vdash \alpha(p) \leftrightarrow q$.*

We are concerned with these theories mainly for the sake of their effect on models:

Definition 3. *A* valuation *of a language $\mathfrak{L}$ is an assignment $\nu(\cdot)$ of truth values to its atoms; this can be extended to the propositions of $\mathfrak{L}$ in the usual way. Such an assignment is a* model *of τ if all theorems of τ are assigned* t.

Suppose, then, that we have a sound interpretation α of $\mathfrak{L}$ in $\mathfrak{L}'$; if ν' is a model of τ', then, by soundness, $\nu' \circ \alpha$, the composition of α with ν, is a model of τ. We thus have a mapping, α^*, from models of τ' to models of τ.

The proofs of the following results are quite straightforward.

Proposition 1. *If α is sound, then α is conservative iff α^* is surjective.*

Proposition 2. *If α is sound, then α is essentially subjective iff $\alpha*$ is injective.*

3.1 Applications

Our applications of these results will be in very simple cases; we will typically start with two different languages, $\mathfrak{L}$ and $\mathfrak{L}'$, and an interpretation α of $\mathfrak{L}$ in $\mathfrak{L}'$. We will also start with a theory in one language or the other, and will define a theory in the other to correspond to it, either by starting with a theory $\tau \subset \mathfrak{L}$ and defining τ' to be the deductive closure of $\alpha(\tau)$, or by starting with a theory $\tau' \subset \mathfrak{L}'$ and defining τ to be $\alpha^{-1}(\tau')$. In either case, soundness is a triviality. We will also be working in situations where it is easy to establish both conservativity and essential surjectivity; so, in consequence, α^* will be a bijective correspondence on models.

We need, however, some more elaborate notation for theories. We are concerned with situations which evolve: for simplicity, our systems will only evolve from time 0 to time 1. However, this means that we have three theories to deal with: we will have a *base theory*, $\underline{\tau}$, consisting of the constraints which are valid atemporally, and we will also have two extensions of this, τ_0 and τ_1, which consist of $\underline{\tau}$ together with the constraints which are valid at times 0 and 1. We will have to verify, rather tediously, our conditions for the correspondences between models of all three theories; we can save some work because, if α is essentially surjective with respect to $\underline{\tau}$, it will be so with respect to τ_0 and τ_1.

We are interested in the $\preceq$ relation between models at time 1 (i.e. models of τ_1): in our situation, this is defined as follows. Suppose we have a language, theories in that language as above, and a model ν_0 of

Vocabulary	Fluents	Order
A	$\mathsf{at}(b_i, j)\|_{i=1}^{2}{}_{j=0}^{3}$ taut	$m_1 \quad m_2 \quad m_3$
B	$\mathsf{above}(b_i, j)\|_{i=1}^{2}{}_{j=0}^{3}$ taut	m_3 | $m_1 \quad m_2$
C	$\mathsf{above}(b_i, j)\|_{i=1}^{2}{}_{j=0}^{3}$	m_3 | m_2 | m_1
D	$\mathsf{above}(b_1, j)\|_{j=0}^{2}$ $\mathsf{above}(b_1, b_2)$, $\mathsf{above}(b_2, b_1)$	$m_1 \quad m_3$ $\diagdown \quad \diagup$ m_2
E	$\mathsf{above}(b_1, j)\|_{j=0}^{2}$ $\mathsf{above}(b_2, b_1)$ taut	m_3 | m_1 | m_2
F	$\mathsf{above}(b_1, j)\|_{j=0}^{2}$ $\mathsf{above}(b_1, b_2)$ taut	m_1 | m_3 | m_2
G	$\mathsf{above}(b_1, j)\|_{j=0}^{2}$ $\mathsf{at}(b_2, 2)$, $\mathsf{at}(b_2, 3)$	$m_2 \quad m_3$ $\diagdown \quad \diagup$ m_1
H	$\mathsf{above}(b_1, j)\|_{j=0}^{2}$ $\mathsf{at}(b_2, 1)$, $\mathsf{at}(b_2, 2)$	$m_1 \quad m_2$ $\diagdown \quad \diagup$ m_3
I	$\mathsf{above}(b_1, j)\|_{j=0}^{2}$ $\mathsf{above}(b_2, 1)$ $\mathsf{at}(b_2, 2)$	m_2 | m_3 | m_1

Table 1: Vocabularies and Their Orders

τ_0. Then we can define a relation as follows:

Definition 4. *If ν is a model of τ_1, define the difference set of ν to be*

$$\delta_{\mathfrak{L},\nu_0}(\nu) = \{p \in \mathfrak{L} | p \text{ an atom}, \nu(p) \neq \nu_0(p)\}$$

We then define the order relation, $\preceq_{\mathfrak{L},\nu_0}$ by inclusion of difference sets:

$$\mu \preceq_{\mathfrak{L},\nu_0} \nu \Leftrightarrow \delta_{\mathfrak{L},\nu_0}(\mu) \subseteq \delta_{\mathfrak{L},\nu_0}(\nu)$$

Notice that we have carefully made explicit the dependence of $\preceq_{\mathfrak{L},\nu_0}$ on both ν_0 and on $\mathfrak{L}$.

Now we have the following

Proposition 3. *Suppose that we have a language $\mathfrak{L}$ together with a theory τ. Then, given a language $\mathfrak{L}'$ containing $\mathfrak{L}$, and a theory τ' such that the inclusion is sound, conservative and essentially surjective, we have*

$$\preceq_{\mathfrak{L}',\nu_0} \subseteq \preceq_{\mathfrak{L},\nu_0}$$

i.e. as we enlarge the language we restrict the accessibility relation on models.

Proof. Clear

□

Example Language B is such an extension of Language C. This example shows that the containment is, in general, strict.

This example prompts the question of how far we can go: whether, given any language, we can extend it so as to make the order relation discrete. It turns out that we can; however, we need another result first.

Proposition 4. *Suppose we have a language, $\mathfrak{L}$, and a theory τ. Suppose that we are given a model, ν_0, of τ, and an order relation $\preceq$ on the models of τ for which ν_0 is the unique least element. Then there is a language $\mathfrak{L}'$, a theory τ', and a sound, conservative and essentially surjective interpretation α of $\mathfrak{L}$ in $\mathfrak{L}'$ such that $\preceq_{\mathfrak{L}',\nu_0} = \preceq$.*

Proof. Before proving the proposition, we first standardise some notation. Let $\mathfrak{M}$ be the set of models of $\mathfrak{L}, \tau$. For each $\nu \in \mathfrak{M}$, take a propositional atom p_ν; let $\mathfrak{L}'$ be the language generated by these propositional atoms. Then define an interpretation α of $\mathfrak{L}$ in $\mathfrak{L}'$ by, firstly, defining, for each model ν of τ, a formula of $\mathfrak{L}'$

$$\vartheta_\nu = \left(\bigwedge_{\nu' \preceq \nu} p_{\nu'} \wedge \bigwedge_{\nu' \not\preceq \nu} \neg p_{\nu'} \right)$$

and then defining the interpretation α of an atom q of $\mathfrak{L}$:

$$\alpha(q) = \bigvee_{\nu(q)=t} \vartheta_\nu \tag{1}$$

Let also τ' be given by the single proposition

$$\alpha(\top) = \bigvee_{\nu} \theta_\nu$$

We can then show that α and τ' give the desired interpretation. □

Given a language, and a desired partial order, this result gives us a *disjoint* language together with an interpretation which realises the partial order in terms of model similarity. The following result shows us what we can do if we want an *enlargement* of the original language: in this case, Proposition 3 shows us that all we can ask for is a subrelation of the original relation $\preceq_{\mathfrak{L},\nu_0}$. But we can achieve any subrelation:

Proposition 5. *Suppose we have a language, $\mathfrak{L}$, a theory τ, and a model ν_0 of $\mathfrak{L}$; we thus have a relation $\preceq_{\mathfrak{L},\nu_0}$ on models of τ. Then, given any reflexive transitive relation $\preceq$ contained in $\preceq_{\mathfrak{L},\nu_0}$, we can find a sound, complete and essentially surjective inclusion of languages $\mathfrak{L} \hookrightarrow \mathfrak{L}'$ such that $\preceq_{\mathfrak{L}',\nu_0} = \preceq$.*

Proof. Suppose given $\mathfrak{L}$, τ and ν_0, together with a desired relation $\preceq \subseteq \preceq_{\mathfrak{L},\nu_0}$. By Proposition 4, we can find a language $\mathfrak{L}'$ together with an interpretation α which realises $\preceq$. Now let $\mathfrak{L}''$ be the language whose set of atoms is the disjoint union of the sets of atoms of $\mathfrak{L}$ and $\mathfrak{L}'$, and expand τ with the definitions $p \leftrightarrow \alpha(p)$ for all atoms p of $\mathfrak{L}$; call this theory τ''. Then $(\mathfrak{L}'', \tau'')$ is the required extension of $(\mathfrak{L}, \tau)$:

$$\preceq_{\mathfrak{L}'',\nu_0} = \preceq_{\mathfrak{L},\nu_0} \cap \preceq,$$

and, since $\preceq \subseteq \preceq_{\mathfrak{L},\nu_0}$. we have the result. □

3.2 Interpretations of Our Vocabularies

We have still to justify our informal description of the mutual interpretations between our different vocabularies. First notice that all of the vocabularies are embedded in a single vocabulary (call it $\hat{\mathfrak{L}}$) which is generated by the union of all of their atoms. There is also a theory, $\hat{\tau}$, which expresses the constraints between these atoms; this theory will have precisely three models, corresponding to our three situations. Given one of our vocabularies, $\mathfrak{L}_i$, we can embed it in $\hat{\mathfrak{L}}$ in the obvious way; let τ_i be $\hat{\tau} \cap \mathfrak{L}_i$. The inclusion is clearly sound and complete. To show that it is essentially surjective, we have to show that each atom of $\hat{\mathfrak{L}}$ can be defined in $\mathfrak{L}_i$; and that can be checked case by case.

4 Inference Relations

What is interesting here is not so much the correctness, or otherwise, of the answers given by particular vocabularies as the fact that the various vocabularies give *different* answers.

We might think that this is nothing to worry about; that the differences are, to be sure, syntactic rather than model-theoretic, but that they would appear natural given a more syntactic approach to nonmonotonic reasoning, such as the inference relations of Kraus, Lehmann and Magidor (Gabbay, 1985; Makinson, 1989; Makinson, 1994; Kraus *et al.*, 1990). In this approach, we typically have two inference relations: a classical relation, $\vdash$, and a non-monotonic one, $\mid\!\sim$; the non-monotonic one is supposed to satisfy the rules in Table 2.

We need an alternative description of such inference relations: first, some notation.

Definition 5. *Suppose that $\mathfrak{L}$ is a propositional language, and that $\mathfrak{M}$ is its set of models. Then, for $p \in \mathfrak{L}$, let $\mu(p) = \{\nu \in \mathfrak{M} | \nu(p) = \mathfrak{t}\}$.*

Then we have

Proposition 6. *Given an inference relation $\mid\!\sim$ satisfying the rules in Table 2, there is exactly one map $\eta : \mathfrak{P}(\mathfrak{M}) \to \mathfrak{P}(\mathfrak{M})$ such that*

$$p \mid\!\sim q \text{ iff } \eta(\mu(p)) \subseteq \mu(q). \tag{2}$$

Conversely, any such η gives rise to exactly one inference relation $\mid\!\sim$.

Proof. Suppose given an inference relation $p \mid\!\sim q$; by Left Logical Equivalence, for any q, if $\mu(p) = \mu(\tilde{p})$, then $p \mid\!\sim q$ iff $\tilde{p} \mid\!\sim q$. Similarly, by Right Weakening, if $\mu(q) = \mu(\tilde{q})$, then, for any p, $p \mid\!\sim q$ iff $p \mid\!\sim \tilde{q}$. $\mid\!\sim$ may thus be regarded as a relation – $\mathcal{R}$, let us say – on $\mathfrak{M} \times \mathfrak{M}$. Now fix $A \subseteq \mathfrak{M}$ (i.e. $A \in \mathfrak{P}(\mathfrak{M})$). Right Weakening entails that, if $A\mathcal{R}B$ and if $B \subseteq \tilde{B}$, then $A\mathcal{R}\tilde{B}$. Similarly, if $A\mathcal{R}B$ and $A\mathcal{R}\tilde{B}$, the And rule tells us that $A\mathcal{R}B \cap \tilde{B}$. Because we have only a finite number of models, any descending chain $\{B_i\}$ of sets of models with $A\mathcal{R}B_i$ must terminate. Thus, for our fixed A, there is a unique minimal B_0 such that $A\mathcal{R}B_0$; thus, by Right Weakening, $A\mathcal{R}B$ iff $B_0 \subseteq B$. Let $\eta(A)$ be this B_0.

Conversely, given an $\eta : \mathfrak{P}(\mathfrak{M}) \to \mathfrak{P}(\mathfrak{M})$, we can, by (2), define a relation $\mid\!\sim$ on $\mathfrak{L} \times \mathfrak{L}$, and it is trivial to verify that it satisfies the conditions of Table 2. □

Example 1. Given a partial order, $\preceq$, on $\mathfrak{M}$, we can define $\eta(A) = \{a \in A | \nexists \tilde{a} \in A . \tilde{a} \prec a\}$ and recover the usual definition of a non-monotonic inference relation induced by a model preference relation.

So now we have our argument. Suppose we have an entailment $A \mid\!\sim B$, formulated in one language $\mathfrak{L}$, and suppose also we have an interpretation α of $\mathfrak{L}$ in $\mathfrak{L}'$ which induces a bijection on models; then, by the above results, we can just as well formulate $\mid\!\sim$ in $\mathfrak{L}'$. Nonmonotonic inference relations may well be syntactically *formulated*, but, because of their logical properties, they are independent of the particular logical *language* that they are formulated in.

In fact, more than this invariance under *equivalent* vocabularies is true: it is also possible to establish that *any* interpretation (whether it is a bijection on models or not) induces mappings with good properties, in both directions, between nonmonotonic relations formulated in each language.

5 Comparisons

The reason why predictions are different in our various vocabularies seems to be this. Vocabulary plays a dual role in the situation calculus. On the one hand, it furnishes terms out of which the language of the theory is constructed. On the other hand, it plays a role in the measurement of change: changes are more or less according to whether their respective sets of changed fluents are included in each other, and fluents are items in the vocabulary. (Or, similarly, the vocabulary plays a role in circumscribing predicates: we circumscribe predicates by minimising the intersection of their extension with the set of fluents).

In the semantics of the situation calculus, then, we should be concerned, not just with the relation between the terms of its formal language and reality: that is adequately described by model theory, and we know that model theory behaves well under change of vocabulary. We should also be concerned with the auxiliary notions of the calculus: the closeness relations, or the way that the circumscription relations are constructed.

5.1 David Lewis

Contrast this with David Lewis' account of counterfactuals (Lewis, 1986). He defines the truth-conditions for counterfactual conditionals in terms of an accessibility relation on a set of possible worlds. His counterfactual connective, $\Box\!\!\rightarrow$, has the following semantics:

> $A \Box\!\!\rightarrow C$ is true at i iff some (accessible) $A \wedge C$-world is closer to i than any $A \wedge \neg C$-world, if

$$\frac{\alpha \vdash \beta}{\alpha \mid\!\sim \beta}\ \text{Supraclassicality} \qquad \frac{\alpha \dashv\vdash \beta \quad \alpha \mid\!\sim \gamma}{\beta \mid\!\sim \gamma}\ \text{Left Logical Equivalence}$$

$$\frac{\alpha \mid\!\sim \beta \quad \vdash \beta \rightarrow \gamma}{\alpha \mid\!\sim \gamma}\ \text{Right Weakening} \qquad \frac{\alpha \mid\!\sim \beta \quad \alpha \mid\!\sim \gamma}{\alpha \mid\!\sim \beta \wedge \gamma}\ \text{And}$$

Table 2: Rules for Non-Monotonic Inference Relations

> there are any (accessible) A-worlds. (Lewis, 1986, p. 10)

Closeness, here, is defined in terms of a triadic relation $j \preceq_i k$, which has certain formal properties; it plays a similar role in Lewis' theory to the fluent-based comparisons of the situation calculus.

The vital difference, though, is that it closeness is a relation between *possible worlds*; and worlds, for Lewis, are elements of reality.[1]

Our situation calculus comparisons cannot be interpreted, then, in terms of Lewis' closeness relations. Since descriptions of the same situations in our two vocabularies describe the same states of affairs – and how could they not, since the the vocabularies are inter-translatable without loss – they are descriptions of the same worlds; and any Lewis-style closeness relation would have to give the same results in both cases. The situation calculus relations are thus not relations between propositional worlds, but relations between propositional descriptions of those worlds.

6 Semantics and Identity

But why should this matter? There are two main reasons, theoretical and practical: they are closely intertwined, but somewhat distinct. We will start with the theoretical reasons.

Quine distinguishes between two levels of semantic comprehension. The first is that of merely responding appropriately to stimuli; this involves a child, for example,

> learning how much of what goes on about him counts as the mother, or as red, or as water. ... Hello! more mama, more red, more water. (Quine, 1969, 7)

Later on, however, we learn to individuate the items that we encounter:

> To learn 'apple' it is not sufficient to learn how much of what goes on counts as an apple; we must learn how much counts as *an* apple, and how much as another. ... [the child] can never fully master 'apple' in its individuative use, except as he gets on with the scheme of enduring and recurrent physical objects. (Quine, 1969, 8)

And thus

> Words like 'apple', and not words like 'mama' or 'water' or 'red', are the terms whose ontological involvement runs deep. (Quine, 1969, 8)

What is at issue here is that these two levels of understanding – the pre-identity stage, and the identity stage – are distinct, and that one may comprehend one without comprehending the other. So a piece of theory – the situation calculus, for example – may succeed in correlating its theoretical terms with the phenomena that it is trying to explain, and may seem to do so quite successfully; but, unless it can go further, and give well-motivated conditions of identity for the theoretical entities that it is trying to introduce, one may well feel that there is something lacking.

7 Excuses

We still need some further argument: we need to know that the problem with change of vocabulary is, indeed, a problem for our common-sense grasp of action and change. If it were not, then we might well argue that things were perfectly correct as they stood: that situations (as described in the situation calculus) could be given conditions of identity, but that they were more fine-grained than the conditions of identity for possible worlds. Indeed, Costello and MacCarthy give an account of counterfactual conditionals in which claims very like these are made. (Costello & McCarthy, 1999) So, if we had such a fine-grained account, situations in one vocabulary could maybe distinct from situations in another vocabulary. This is not so: but we need some argument to show it.

The first stage of the argument uses our metatheoretical results: we can achieve, by appropriate choice of vocabulary, *any* partial order on models that we desire. So, if we want to talk about the mathematical

[1] See (Lewis, 1983c, p. 39) "A world is a large possible individual; it has smaller possible individuals as parts".

structure which results from a choice of vocabulary, it is simply this: an arbitrary choice of a partial order on models.

However, one might consider the following response to these results: that we are not concerned with *all* possible vocabularies, but only with those that are, in some sense, natural; those that arise from our everyday use of language. If this response were valid, then it would have to be the case that the vocabulary given by natural language was quite restrictive: that we could not choose vocabularies arbitrarily, but that, for each situation, we naturally used a vocabulary which was tightly enough prescribed to fix our inferences adequately, and that we never, in the course of reasoning about a given situation, changed our vocabulary enough to change the validity of our inferences.

Davidson (following Austin (1956–7)) considers the following pattern of reasoning.

> 'I didn't know that it was loaded' belongs to one standard pattern of excuse. I do not deny that I pointed the gun and pulled the trigger, nor that I shot the victim. My ignorance explains how it happens that I pointed the gun and pulled the trigger intentionally, but did not shoot the victim intentionally. ... What is the relation between my pointing the gun and pulling the trigger, and my shooting the victim? The natural and, I think, correct answer is that the relation is that of identity. The logic of this sort of excuse includes, it seems, at least this much structure: I am accused of doing b, which is deplorable. I admit I did a, which is excusable. My excuse for doing b rests upon my claim that I did not know that $a = b$. (Davidson, 1980, p. 109)

What we are doing here is identifying two actions which are described differently, using different vocabulary: and the grounds on which we are identifying the actions are that they are the actions consist of the same sequence of events in the world. Thus, our everyday language exploits a wide range of redescriptions of the same action using different vocabulary. If all these things, variously redescribed, are the same action, then they must all have the same preconditions, the same effects, and they must all function in the same way in any adequate theory of action. And, finally, it is hard to see how the two actions of 'lifting the ball to position 2' and 'lifting the ball above positions 0 and 1, but not above position 2' could not be the same action. To establish identity, one needs, after all, only the facts of geometry, whereas for the identity between 'shooting' and 'pulling the trigger' one needs certain empirical matters, such as the presence of the bullet. So, if the identity holds in the case of Davidson's shooting example, it should, *a fortiori*, also hold in our example.

If, furthermore, we are to take seriously the appeal to natural language, we ought to take seriously Proposition 3. Natural languages typically abound in synonyms: they have large vocabularies, and most of the items in those vocabularies can be defined, in multiple ways, in terms of other items of those vocabularies. As the proposition shows, when we go about enlarging our vocabulary by definitions, we make our partial order successively more discrete. As the examples show, we do not, typically, need to make our vocabulary particularly large in order to make the resulting order completely discrete, and thus to make circumscription yield the same results as classical logic. And surely, if there is any particular vocabulary that the appeal to natural language ought to recommend, it must be the entire vocabulary of natural language, with its accompanying discrete partial order.

8 Conclusion

In these situation calculus inferences, the vocabulary plays a double role. One is its normal role: to form expressions in the relevant formal language. The other role, however, is to define the similarity relation between sets of fluents: two sets are more or less similar depending on inclusions between sets of fluents that have changed truth values in each of them. This relation is not independent of the vocabulary that it is formulated in; consequently, it is not actually a relation between possible worlds, but rather between *descriptions* of worlds. This means that the situation calculus, despite superficial similarities, is, in fact, markedly different from Lewis' semantics of counterfactual conditionals. It is also very different from the account of change that we get in physics: here we have a theory of physical change which relies on minimising a certain quantity (namely the integral of the action over a path in phase space), but the action is carefully defined so that the values of such integrals are independent of the coordinate system that they are expressed in.

One could argue, at this point, that these are little more than analogies: that the purpose of situation calculus is merely to imitate common-sense reasoning, and the role of change of vocabulary in such reasoning is small. Davidson's arguments show that this is not so: that change of vocabulary plays a very extensive role in our reasoning about actions.

These considerations have two main repercussions: theoretical and practical. Theoretically, it seems difficult to imagine how the situation calculus could be given a semantics in any strong sense. A semantics, however it might be expressed formally, should surely correlate linguistic items either with components of reality or with more abstract, mathematical objects: and, although the components of reality might have to be *described* in linguistic terms, the *correlation* itself should be independent of merely linguistic matters such as vocabulary. In particular, it should not be affected by the substitution of one vocabulary for another, logically equivalent one. And one should observe, in passing, that the rules of left logical equivalence and right weakening mean that any semantics for Kraus, Lehmann and Magidor-style inference relations must have these properties.

Practically, there are the following consequences. We would like to be able to use a theory (the situation calculus, maybe) by starting with whatever vocabulary presented itself and working with that. All we should have to worry about would be whether the vocabulary could describe the phenomena, and the correctness of our description: given both of these, the calculus ought to work. The situation ought to be similar to that in physics, where we can choose a coordinate system at our convenience: given a correct description, and correct equations, the results will be correct. But with the situation calculus as it is, we have more to worry about than merely adequacy and correctness: we also have to worry whether we have chosen a vocabulary which yields the right closeness relation between situations. All of our vocabularies are descriptively adequate, and our descriptions in any of them are correct. But they vary greatly in the predictions they make. And there seems to be no other way of telling whether the vocabulary is the magic correct one, other than by checking the results; and this rather diminishes the utility of the calculus as a predictive theory.

References

Austin, J.L. 1956–7. A Plea for Excuses. *Proceedings of the Aristotelean Society*, **57**. Reprinted in (Austin, 1970, pp. 175–204).

Austin, J.L. 1970. *Philosophical Papers.* Oxford University Press.

Costello, Tom, & McCarthy, John. 1999. Useful Counterfactuals. *Linköping Electronic Articles in Computer and Information Science*, 4(12). Available at `http://www.ep.liu.se/ea/cis/1999/012`.

Davidson, Donald. 1980. The Logical Form of Action Sentences. *Pages 105–148 of: Essays on Actions and Events.* Oxford University Press. Originally published in N. Rescher (ed.), *The Logic of Decision and Action*, University of Pittsburgh Press, 1967.

Gabbay, Dov. 1985. Theoretical Foundations for Nonmonotonic Reasoning in Expert Systems. *In:* Apt, K. (ed), *Logics and Models of Concurrent Systems.* Berlin: Springer-Verlag.

Gabbay, Dov M., Hogger, C.J., & Robinson, J.A. 1994. *Handbook of Logic in Artificial Intelligence and Logic Programming.* Vol. 3. Clarendon. Volume Coordinator D. Nute.

Kraus, Sarit, Lehmann, Daniel, & Magidor, Menachem. 1990. Nonmonotonic Reasoning, Preferential Models and Cumulative Logics. *Artificial Intelligence*, **44**, 167–207.

Lewis, David. 1983a. Counterpart Theory and Quantified Modal Logic. *In:* (Lewis, 1983b). Originally appeared in *Journal of Philosophy* 65 (1968), 113–126.

Lewis, David. 1983b. *Philosophical Papers.* Vol. I. Oxford University Press.

Lewis, David. 1983c. Postscripts to 'Counterpart Theory and Quantified Modal Logic'. *In:* (Lewis, 1983b). Postscripts to (Lewis, 1983a).

Lewis, David. 1986. Counterfactuals and Comparative Possibility. *Pages 3–31 of: Philosophical Papers*, vol. II. Oxford University Press. Originally published in the *Journal of Philosophical Logic* 2 (1973): 418–46.

Lifschitz, Vladimir. 1994. Circumscription. *In:* (Gabbay *et al.*, 1994). Volume Coordinator D. Nute.

Makinson, David. 1989. General Theory of Cumulative Inference. *Pages 1–18 of:* Reinfranck, M. (ed), *Non-Monotonic Reasoning.* Lecture Notes in Artificial Intelligence, no. 346. Springer-Verlag.

Makinson, David. 1994. General Patterns in Nonmonotonic Reasoning. *In:* (Gabbay *et al.*, 1994). Volume Coordinator D. Nute.

Quine, Willard van Ormond. 1969. Speaking of Objects. *Pages 1–25 of: Ontological Relativity and Other Essays.* New York NY: Columbia.

Shanahan, Murray. 1997. *Solving the Frame Problem: A Mathematical Investigation of the Common Sense Law of Inertia.* MIT Press.

Modeling Complex Domains of Actions and Change

Antonis Kakas and Loizos Michael
Dept. of Computer Science
University of Cyprus
P.O.Box 20537, CY-1678 Nicosia, Cyprus
antonis@ucy.ac.cy, cs98ml1@ucy.ac.cy

Abstract

This paper studies the problem of modeling complex domains of actions and change within high-level action description languages. We investigate two main issues of concern: (a) can we represent complex domains that capture together different problems such as ramifications, non-determinism and concurrency of actions, at a high-level, close to the given natural ontology of the problem domain and (b) what features of such a representation can affect, and how, its computational behaviour. The paper describes the main problems faced in this representation task and presents the results of an empirical study, carried out through a series of controlled experiments, to analyze the computational performance of reasoning in these representations. The experiments compare different representations obtained, for example, by changing the basic ontology of the domain or by varying the degree of use of indirect effect laws through domain constraints. This study has helped to expose the main sources of computational difficulty in the reasoning and suggest some methodological guidelines for representing complex domains. Although our work has been carried out within one particular high-level description language, we believe that the results, especially those that relate to the problems of representation, are independent of the specific modeling language.

1 Introduction and Background

Over the last decade several new frameworks for reasoning about actions and change have been proposed. Together with the original frameworks of the situation and event calculi, we have seen a number of different action description languages emerging with recent implemented prototype systems [3, 1, 4, 7]. Many of these languages have been motivated by the desire to have simple representation frameworks, with semantics that relate more directly to the natural intended meaning of their statements. They, therefore, aspire to provide high-level representations of domains that are close to their natural specification.

This paper studies the problem of modeling complex domains of reasoning about actions and change within high-level action description languages. It aims to expose the main problems involved in this representation task and to study the computational behaviour of the ensuing representations. It considers the Zoo domain, proposed at the Logic Modeling Workshop of the ETAI electronic communication forum [2], as one representative complex domain that frameworks of reasoning about actions and change should aim to capture.

A complex domain brings together a number of different problems associated with reasoning about actions and change, such as ramifications, non-determinism and concurrency. Although all these aspects have been extensively studied and several solutions have been proposed these studies are largely done for one problem in isolation and with little attention to the computational behaviour of these solutions. In this work we study how we can use the knowhow generated over the years in these separate problems, in order to represent complex and realistic domains that encompass together all these different features. Despite the fact that we will use a particular representation framework, that of the Language $\mathcal{E}$, we believe that the results apply more generally. This language encompasses (partial) solutions to the main problems of reasoning about action and change that are based on the same principles used by most of the other approaches. Hence the findings of our work, especially those that refer to the representational task would apply to other formalisms as well.

In particular, we investigate two main issues of concern: (a) can we represent complex domains in a natural way

using directly the given natural ontology of the problem domain and (b) what features of such a representation can affect (and how) its computational behaviour. Our method of study uses an empirical investigation through a series of controlled experiments. In some of these experiments we compare different representations by varying, for example, the basic ontology of the domain or the degree of use of indirect effect laws through domain constraints. In other experiments, we investigate the performance of a fixed representation as we vary the type of problems we consider, e.g by varying the degree of incompleteness of information in a scenario or by varying the complexity of the narrative with irrelevant action occurrences.

These experiments help to analyze the computational difficulties of the reasoning by exposing their main sources. In turn, they suggest methodological guidelines on how to develop complete representations of complex domains with improved computational properties.

In the rest of this section we present briefly the specific action description language that we will use in our modeling study. Section 2 studies the main problems faced in the task of representing a complex domain (by analyzing these for the Zoo domain). Section 3 discusses some of the experiments carried out to study the computational behaviour of representations and presents their main conclusions. Section 4 discusses the overall conclusions drawn from our work.

1.1 The Action Description Language $\mathcal{E}$

Action description languages have been developed over the last years to provide high level representation frameworks for theories about actions and change. They generally do not have an explicit representation of the frame axioms and they have been extended to address the ramification and qualification problems. They have been motivated by the need to have simple representation frameworks with natural semantics that relate more directly to the natural interpretations of the domains.

We will consider the problem of modeling complex domains within the particular action description Language $\mathcal{E}$ [7, 8]. The vocabulary of the Language $\mathcal{E}$ consists of a set Φ of *fluent constants*, a set of *action constants*, and a partially ordered set $\langle \Pi, \preceq \rangle$ of *time-points*. This vocabulary depends each time on the domain being modeled. A *fluent literal* is either a fluent constant F or its negation $\neg F$. In the current implementation of the language, the only time structure that is supported is that of the natural numbers, so we restrict our attention here mainly to domains of this type.

Domain descriptions in the Language $\mathcal{E}$ are collections of the following types of statements (where A is an action constant, T is a time-point, F is a fluent constant, L is a fluent literal and C is a set of fluent literals):

- *t-propositions* of the form: L **holds-at** T
- *h-propositions* of the form: A **happens-at** T
- *c-propositions* of the form: A **initiates** F **when** C or, A **terminates** F **when** C
- *r-propositions* of the form: L **whenever** C
- *p-propositions* of the form: A **needs** C.

For the purposes of this paper it is sufficient to describe the semantics of the language in an informal way, giving the intended meaning of each type of sentence and describing the basic tenets of the reasoning supported. T-propositions record observations that particular fluents hold or do not hold at particular time-points. H-propositions state that particular actions occur at particular time-points. C-propositions state general "action laws" – the intended meaning of "A **initiates** F **when** C" is that "C is a minimally sufficient set of conditions for an occurrence of A to initiate F". R-propositions, also called **ramifications or domain constraints**, serve a dual role in that they describe both static constraints between fluents and ways in which fluents may be indirectly affected by action occurrences. As a constraint this must be satisfied as a classical implication at any time point. The intended meaning of "L **whenever** C" is that "at every time-point that C holds, L holds, and hence every action occurrence that brings about C also brings about L". P-propositions state necessary conditions for an action to occur.

The semantics of $\mathcal{E}$ are based on a notion of a *model* of a domain D that assigns a truth value to each fluent at each time point. A model encompasses two basic requirements:

Persistence Direct and indirect action laws, stated by c-propositions and through r-propositions respectively, are the only way to bring about a change (over time) in the truth value of a fluent, when an appropriate action occurs. This new value persists in time from then onwards, until another such point of change.

Consistency The model must satisfy as classical implications the r-propositions at each time point and as facts the t-propositions together with all the preconditions C at times T that follow from each p-proposition A **needs** C and associated h-proposition A **happens-at** T.

Given a consistent domain D and a query or goal G comprising a set of t-propositions, we say that G is a **credulous or possible** conclusion of D iff there is a model of D in

which G is true and that G is a **skeptical or necessary** conclusion of D iff G is true in all models of D.

As an example, consider the following simple "bulb domain" with action constants *SwitchOn*, *SwitchOff* and *Break* and fluents *Light* and *Normal*:

SwitchOn **initiates** *Light* **when** {*Normal*}
SwitchOff **terminates** *Light*
Break **terminates** *Normal*
¬*Light* **whenever** {¬*Normal*}
SwitchOn **needs** {¬*Light*}
SwitchOn **happens-at** 2
Normal **holds-at** 0

This domain has as a skeptical conclusion *Light* `holds-at` 4, but the conclusion is only credulous when we remove the last sentence.

The Language $\mathcal{E}$ has been implemented [9] via an argumentation-based translation of the language. This system, called $\mathcal{E}$-RES, can support directly a variety of modes of common sense reasoning such as: default persistence in credulous or skeptical form, assimilation of observations and their diagnosis possibly under incomplete information, as well as combinations of these. The computational model on which this is based integrates argumentation goal-oriented proofs for default persistence, together with classical theorem proving techniques for the satisfaction of the constraints expressed in a domain. It also employs a (weak) notion of syntactic relevance in order to help focus the computation to the relevant part of the theory.

This argumentation-based reformulation also extends the Language $\mathcal{E}$ allowing its action laws to be treated as default laws.

2 Complex Domain Representations

In this section we study how the Zoo domain, defined at the Logic Modeling Workshop (LMW) [2], can be represented in the Language $\mathcal{E}$. We will follow closely the presentation of the domain and guidelines given at LMW trying to address all the issues raised there. This section is therefore best read in conjunction with the documentation available at LMW. We will present the different aspects of representing such a complex domain, while trying to expose the main problems faced and choices made.

The LMW describes the Zoo domain as "...a world containing the main ingredients of a classical zoo: cages, animals in the cages, gates between two cages, as well as gates between a cage and the exterior. In the Zoo world there are animals of several species, including humans. Actions in the world may include movement within and between cages, opening and closing gates, ..., riding animals, etc."

2.1 Background Landscape and Active Structure

Our first task is to represent the **landscape and active structure** of the domain i.e. describe the background information of the Zoo environment and the objects that populate it. This information is generally taken to be static, not changing over time. For this type of information the Language $\mathcal{E}$ has a simple facility of using **constant** fluents and declaring the static state of affairs for these at some initial time point (time 0). For example, we declare the known animals in the domain by a set of statements of the following form:

$animal(-)$ **is_constant**
$animal(john)$ **holds-at** 0
$animal(jane)$ **holds-at** 0
$animal(elly)$ **holds-at** 0
...

In addition, we can use ramification statements to represent static information that is derived from other such information e.g. that "an animal is large when it is an adult and belongs to a large species":

$animal_is_large(A)$ **whenever** $\{animal_is_adult(A), animal_species(A, S), species_is_large(S)\}$.

In the same way, we represent the zoo terrain by declaring all its positions and the neighbor relation via a constant fluent $neighbor_pos(_,_)$ and ramification statements to express its properties e.g. $neighbor_pos(P_1, P_2)$ **whenever** $\{neighbor_pos(P_2, P_1)\}$ for the symmetry of this relation.

We then reason with the Closed World Assumption on these static fluent predicates at time 0 and project their truth value unchanged to any other time point. If complete information for a background fluent predicate is not known at 0, then this predicate would not be declared as constant and thus would be interpreted under the Open World Assumption with the possibility to assume its truth value.

We note that the inherent propositional nature of the Language $\mathcal{E}$ does pose some representational restrictions. We need for example to define the primary typing constants, which enumerate entities of the landscape and hence only domains with known predefined entities can be represented. In general, the underlying ontology of the language can be extended with a notion of sorted variables over a set of finite sorts. The details of this are not important for this work. In this paper, we will assume that we have in the Zoo domain the basic types of animals, species, gates, locations and positions, defined as constant fluents, and that any variable (i=1,2,3,...) denoted by $A_i, S_i, G_i, L_i, or\ P_i$ has the respective type.

2.2 Action Effect Laws

Till now we have only been concerned with the static aspects of the representation of the domain i.e. representing information at one time slice. The central part of any domain of reasoning about actions and change is that of its **Action laws specifying the effects of actions**. In the Language $\mathcal{E}$ action laws are represented directly via c-propositions. For example, the effect of the $move_to_position(A, P)$ action in the Zoo domain, that the animal "A" takes the position "P" is represented by:

$move_to_position(A, P)$ **initiates** $animal_pos(A, P)$
when $\{reachable(A, P)\}$[1].

Here this effect law depends on the condition that at the time of the action the position "P" is reachable by "A", namely that "P" is an adjacent position to the position of the animal "A" at the time of the action. This means that an instance of the action $move_to_position(A, P)$ could occur without successfully reaching "P". Alternatively, we can remove this condition from the effect law and set it as a pre-condition for the action to occur via the statement:

$move_to_position(A, P)$ **needs** $\{reachable(A, P)\}$.

This choice depends on how we wish to interpret the occurrence of events in a given narrative. If when given an event we mean that this event has successfully occurred, then this alternative is more appropriate and allows for more conclusions to be drawn, given an action occurrence. For example, we would include the statement

$move_to_position(A, P)$ **needs** $\{\neg rides(A, A_1)\}$

to capture the requirement that a rider cannot perform the $move_to_position$ action at the time that s/he is riding. Under this interpretation of event occurrences this way of representing these preconditions allows us to draw more conclusions from knowledge that an action has occurred. If then, we are given that $move_to_position(john, p_1)$ happened at time T then we would conclude that $john$ is not riding any (known) animal at T. Depending on other information in the domain we maybe able to project this backward or forward in time.

The above law of change for the action $move_to_position(A, P)$ expresses a **direct** effect. This action can also have other effects, for example that a rider A_1 of the animal A will also acquire the new position P, when A moves to P. In comparison, this is not a direct effect but nevertheless, as argued in the recent literature (e.g. [11]), actions should be allowed to have such **indirect** effects. This poses a dilemma in the representation. Indirect effects could be represented either by explicit direct effect laws or via the domain constraints (or ramification statements) of the theory. In the above example we could have the additional direct laws:

$move_to_position(A, P)$ **initiates** $animal_pos(A_1, P)$
when $\{rides(A_1, A)\}$,
$move_to_position(A, P)$ **terminates** $animal_pos(A, P_1)$
when $\{animal_pos(A, P_1)\}$

for moving an animal's rider along with the animal and terminating the animal's current position, respectively. On the other hand, we could omit these, since the following domain constraints (stating that an animal's rider always has the same position with the ridden animal and that an animal cannot be at two positions at the same time) would generate the required effects indirectly:

$animal_pos(A_1, P)$ **whenever** $\{animal_pos(A, P), rides(A_1, A)\}$,

$\neg animal_pos(A, P_1)$ **whenever** $\{animal_pos(A, P), P_1 \neq P\}$.

However, such domain constraints are also needed in the representation to suitably restrict the assumptions that we might make while reasoning, when we have incomplete information in the theory.

Hence a dilemma emerges, either (1) to use direct laws to explicitly enumerate all possible effects of an action together with domain constraints which do not generate indirect effects, but simply act as integrity constraints for assumptions relating to incomplete fluents or (2) to use direct laws for the basic effects of an action and let the domain ramification constraints generate the other effects indirectly. We will see below that these two choices can be semantically different and that they differ in their computational performance.

2.3 Domain Constraints

To represent the whole Zoo domain we need to define all direct action laws for each of the type of actions together with their pre-conditions (as we have done above for $move_to_position$) and state the various **domain constraints** that the specification of the problem requires. In addition, to the domain constraints given above, we also have several other statements. For example, we have:

$false$ **whenever** $\{animal_species(A, human), rides(A_1, A)\}$,

$\neg rides(A, A_1)$ **whenever** $\{rides(A, A_2), A_1 \neq A_2\}$

to capture the fact that a human cannot be ridden and to

[1]Note that the current version of the $\mathcal{E}$-RES system, that implements the Language $\mathcal{E}$, requires that the typing of each variable of a fluent or action to be stated explicitly each time the fluent or action is used in a statement. Hence in $\mathcal{E}$-RES this will written as:
$move_to_position(A, P)$ **initiates** $animal_pos(A, P)$
when $\{animal(A), position(P), reachable(A, P)\}$.

represent the fact that an animal cannot ride two animals at the same time. Here the first domain is written in this particular way as a denial to indicate that it is not necessary to produce indirect effects through it.

The domain constraints are also used to define various auxiliary dynamic predicates that we need, such as *reachable* used above:

$reachable(A, P)$ **whenever** $\{animal_pos(A, P_1), neighbor(P_1, P)\}$,

$\neg reachable(A, P)$ **whenever** $\{animal_pos(A, P_1), \neg neighbor(P_1, P)\}$

Note that here we do have the option (or dilemma) to define such dynamic predicates via direct effect laws of the actions, but this would be more complex and non-modular requiring the full enumeration of all cases where the fluent *reachable* could be affected. Its definition through the above domain constraints and their indirect effects that they embody, provides a compact and modular way of representing such a dynamic predicate.

2.4 Issues in the choice of Representation

We return now to study in more detail the above dilemma and in particular how this is related to non-determinism in the domain and its representation. One of the actions of the Zoo domain, that of $throwoff(A_1, A_2)$, is specified to be non-deterministic. Animal A_1 throws off its back A_2 and the latter can land at any position reachable from their common position at the time of the throw. As there could be many such reachable positions and we do not know at which one the animal would fall this is a non-deterministic action. The argumentation semantics of the Language $\mathcal{E}$ allows us to represent such a non-deterministic action law in the same way as any other law:

$throwoff(A_1, A_2)$ **initiates** $animal_pos(A_2, P)$ **when** $\{reachable(A_2, P)\}$.

This effect law does not contain any explicit qualifications to state that the animal would fall in one position only provided that it does not fall on anyone of the other (reachable) positions. This qualification is implicitly produced by the domain constraint that an animal cannot have two different positions at the same time. The combination of these two statements captures the non-determinism of the action. An inconsistency is not produced (although apparently it would be), because for each direct initiation of a particular position P that the action law produces, the domain constraint then produces the indirect effect that all other positions are terminated. In this way each position can be chosen. Note that it is important that the blockage of the other positions is not captured simply by the static application of the constraint, but that at the same time the position of an animal changes to P we also have a change with regards to all the other positions, namely they "begin to be false".

We therefore see a useful need for the production of indirect effects, i.e. of change through the domain constraints. This provides a natural mechanism of implicit qualification of the several direct laws for a non-deterministic action. Alternatively, we would need to qualify explicitly the direct law with the condition that the animal can not be at any other position. This would be complex and difficult to express because the qualifications to the action law need to refer to the state of the world after the action has occurred.

The same behaviour can also be observed in the case of concurrent actions whose effects are contradictory. Consider for example the simultaneous occurrence of a $getoff(A, A_1, P)$ and a $move_to_position(A_1, P_1)$ action, at a time when the animal A is riding another animal A_1. The first action requires that A has the position P as its direct effect, while the second action would give that A has the different position P_1 since A is riding A_1. Again, the above constraint that an animal can have only one position at a time, acts as a qualifier and splits this scenario in a non-deterministic way into two possibilities. In effect, in the first case A manages to get off before A_1 moves, whereas in the second case A_1 moves first and then A gets off. The domain constraint gives an implicit qualification to the two effect laws involved, in a way that would be very difficult (and unnatural) to represent explicitly.

Concurrent action scenaria also show the main difference between direct and indirect effect laws. Let us return to the indirect effect law that a rider acquires the same position of the animal that s/he is riding and compare it with its direct effect law counterpart as given above. A difference between the two laws shows up when we consider, for example, the concurrent $getoff$ and *move_to_position* scenario, above. The direct law would allow the rider to move to the new position with the animal, whereas the indirect law would not generate this effect. This is because the condition of *rides* in the direct law case needs to hold at the exact time of the action, while in the indirect law case this needs to hold at an infinitesimal time (or next time point) after this, so that both conditions in the ramification can hold together to produce its conclusion. Indeed, due to the simultaneous $getoff$ action, rides holds at the time of these actions, but not later as it is terminated by this action unconditionally. Hence, in this concurrent scenario we would not get the second possibility of the indirect generation of the rider moving with the animal.

In this way we see that direct laws can be stronger than a corresponding indirect effect law and in fact could be used to break non-determinism when we do not want this. To see this better and show its significance for the task of modeling

a domain, let us consider a case where the non-determinism could exist as a result of the occurrence of one action alone. Consider the domain constraint that two animals at different positions cannot ride each other:

$\neg rides(A, A_1)$ **whenever** $\{animal_pos(A, P), animal_pos(A_1, P_1), P_1 \neq P\}$.

This would then give indirect effects of the termination of riding when we perform actions that change the position of the rider alone (such as $getoff$ or $throwoff$). But it would also give the indirect effect that riding is terminated when the animal moves and hence it changes its position relative to its rider. In fact, adding the above ramification to the representation and performing an action $move_to_position(A_1, P_1)$, at a time point where the animal A_1 is ridden by another animal, results in two possibilities: either the rider comes off and is at the original position of A_1, or the rider stays on and is at the new position P_1 where the animal has moved to. With regards to the effect of the action $move_to_position(A_1, P_1)$ on the position of the rider the action is non-deterministic.

In this case we do not want this and we have a preference for the rider to move along with the animal that it rides. We can achieve this preference in a natural way by explicitly stating the preferred effect through a direct effect law. Despite the fact that we have a domain constraint that gives, as an indirect effect, that the rider moves along with the animal, we **also** represent this explicitly by a direct initiation law for the action $move_to_position(A, P)$, i.e. we keep both sentences in the dilemma presented above in this section. This then blocks the indirect effect of terminating the riding by the movement of the animal as the required condition for this, namely that the rider is at a different position immediately after the action, cannot now be supported due to the strong argument of changing its position to that of the animal's new position through the new direct law and the constraint that the rider can only have one position at a time. The explicit statement of the direct law has given preference to it.

In summary, we see that the ramifications and their indirect effects provide an implicit form of qualification through the general constraint properties of the domain. This means that the theory treats its effect laws as default laws where one can qualify another through these constraints. The direct effect laws set out preferences amongst conflicting possibilities. The redundancy of repeating an indirect law as a direct law as well, forms a natural way of setting a preference. We give emphasis to effects by articulating them explicitly.

2.5 An example Scenario

Closing this section, we show how a particular scenario can be represented. A scenario is given by stating events, i.e. specific instances of actions that occurred (through h-propositions) and observations, that is values of fluents at specific time points (through t-propositions). For example,

$throwoff(elly, john)$ **happens-at** 1
$move_to_position(john, p1)$ **happens-at** 2
$mount_animal(john, dumpo)$ **happens-at** 3
$reachable(john, p1)$ **holds-at** 2

represents the scenario that $elly$ threw off $john$, who then observed that position $p1$ was reachable, moved to position $p1$, and then mounted $dumpo$. In this scenario we would reason to $rides(john, elly)$ at time 1 or 0 as a necessary (or skeptical) conclusion, but that $animal_pos(john, p2)$ at time 2, where $p2$ is any neighbor of $p1$, is a possible (or credulous) conclusion. At time 4 onwards we can conclude that $rides(john, dumpo)$ is necessarily true. But if we are given also that $move_to_position(dumpo, p3)$ **happens-at** 3, then $rides(john, dumpo)$ at 4 will only be a possible conclusion. Observing that $animal_pos(john, p3)$ **holds-at** 5 i.e that $john$ and $dumpo$ have the same position then $rides(john, dumpo)$ at 4 would be a necessary conclusion.

3 Computational Behaviour of Representations

In this section we study empirically, via suitable controlled experiments, the computational behaviour of a representation and try to ascertain what factors of the representation might or might not affect this behaviour. We have conducted two major types of experiments: (a) given a fixed representation of the Zoo domain, we have examined various scenaria that differ in the amount of given information, its relevancy to the queries and the type of queries performed, and (b) given some fixed scenarios and queries, we have examined various representations that differ on the extend the action effects are represented directly/indirectly and on the richness of the vocabulary used. The basic scale of the domains on which we have carried these experiments is of the order of 3 thousand ground clauses per time point. A scenario typically spans over 3-6 time points.

As the representative results[2] presented in figure 1 suggest, increasing the information in a scenario can affect positively the computational performance, even if the extra

[2] All numbers shown in the tables of this paper are in seconds measured on a Pentium II, 266 MHz, PC. The absolute value of these times are not important as the purpose of the experiments is to compare different cases.

information given explicitly in the scenario is a necessary conclusion of this.

Query / Result	1/Y	2/Y	3/N	4/Y	5/N	6/Y	7/Y	8/Y	9/N	10/N
Complete	9	12	2	13	12	7	8	11	3	2
Incomplete	14	33	19	19	17	9	11	28	4	17

Figure 1: Complete vs Incomplete Scenario Knowledge

These experiments have also shown that it is possible to use simple syntactic relevancy checks to focus the computation to the relevant (w.r.t. the query at hand) part of the domain. In fact, the experiments have helped us develop further a semantically enhanced notion of relevancy that neutralizes to some extent the effect of additional irrelevant information in the scenario. Figure 2 shows that the addition of irrelevant actions in a scenario does not have a significant impact on the computation. The stability under irrelevant information also depends strongly on the type of representation used and the choice of vocabulary, as we will see below in the second class of experiments. Finally, as expected in some cases, particularly in scenarios with a lot of missing information, skeptical queries can be significantly slower than the corresponding credulous queries.

Query / Result	1/Y	2/Y	3/N	4/N	5/Y	6/Y	7/Y	8/Y	9/Y	10/N
Relevant	35	48	17	54	49	31	42	38	261	266
Irrelevant	43	70	24	81	67	48	93	112	290	440

Figure 2: Relevant vs Additional Irrelevant Actions

As mentioned above, the second class of experiments considers changes in the representation. Figure 3 compares the computational behaviour between representations which use direct effect laws to a different degree. In general, using only direct laws gives a significantly better computational performance than a representation that uses to a maximum indirect effect laws. We should note though, as we have seen in section 2, that the first representation cannot always fully capture the domain e.g. the non-determinism of conflicting concurrent actions. In such cases, this becomes inconsistent and is unable to return an answer. It is also noteworthy, that when effect laws are duplicated in the representation both directly and indirectly, the behaviour varies from being as good as a representation that primarily uses direct effects, to being worse than a representation that primarily uses indirect effects. This depends mostly on whether the specific laws that are duplicated as direct laws can generate a large number of additional indirect effects.

Another series of experiments has examined the effects on the computational behaviour of using an enriched, more specific vocabulary of the domain in the representation. Starting from a minimally sufficient vocabulary, we have

Query / Result	1/Y	2/N	3/N	4/Y	5/N	6/N	7/Y	8/Y	9/N	10/Y
Direct (D)	17	22	7	19	8	33	15	22	9	8
Indirect (I)	42	26	3	14	362	18	9	43	328	37

Figure 3: Direct vs Indirect Effect Laws

developed a number of other equivalent representations using more specialized vocabularies that separate explicitly different specific cases for an action, e.g. the single action fluent *move_to_position* is replaced by *move_in_cage*, *enter_cage* and *exit_cage*. Similarly, we can distinguish explicitly in the vocabulary some of the fluent properties e.g. those that refer to the (2 dimensional) position on the floor from the properties that refer to the third dimension off the floor. We have found that a representation that uses, in this way, a carefully selected ontology and proper typing can significantly improve its computational efficiency. This occurs mainly by reducing the number of assumptions (sometimes impossible ones) from being considered in the reasoning.

With regards to the issue of scaling of the computational behaviour of the representations, our experiments suggest that the representation that maximizes the use of direct effect laws scales up relatively well. These are preliminary results, tested only on small sized domains that contain up to a maximum of 15 positions in the terrain, corresponding to around 25,000 ground clauses. Representations with non-minimal use of indirect effect laws scale up poorly.

In general, our empirical study has revealed two main computational bottlenecks: (a) the number of additional consequences drawn from the conclusions of a query that need to be considered to ensure that the query can be consistently satisfied in the whole domain and (b) the number of counter-proofs (counter-arguments) that are generated via persistence on an assumption needed to prove the query. Our experiments have helped us address (at least partly) these difficulties. The enhanced notion of relevancy, which helps us avoid unnecessary re-computation of known consequences and a (non-minimal) choice of vocabulary that distinguishes explicitly different scenaria, can result in significant improvement of computational behaviour. Another potential source of improvement can come from the way we use direct and indirect effect laws. Generally, the more use of direct laws the better the performance, but there is a trade off here with the reduction in modularity and readability of a representation that uses direct laws extensively. In some cases the computational gain is relatively small to justify this and a representation that combines the use of these two types of effect laws is overall better.

We have also begun to examine the same problems using a different computational model in place of the argumentation based model used so far. We have translated the Lan-

guage $\mathcal{E}$ to a SAT theory so that we can then employ a SAT solver for query answering. This translation is not complete and supports only a restricted class of domains. In particular, it does not support domains with non-deterministic actions, or concurrent actions that generate conflicting effects nor domains in which the ramifications form cycles. Initial results indicate that the SAT-based approach has a more stable behaviour. They also indicate that it scales up better as we increase the number of actions in a scenario, but not so as we increase the number of positions in the terrain of the Zoo domain. An interesting difference of the SAT-based approach is the fact that it is difficult to employ relevancy notions in it. Some simple forms of relevancy can be used, but this requires that for each query we need to translate the relevant part of the domain theory afresh and this can have an adverse effect on its performance.

4 Related Work and Conclusions

We have studied how to model complex domains of action and change based on the theoretical foundations that the community at large has developed over the last decade. In particular, we have addressed the challenge set by the Logic Modeling Workshop [2], to apply this theory in modeling complex domains that encompass together many different aspects of the problem. As in the related work of [5, 6, 12] our work shows that the existing frameworks of action description languages are sufficiently expressive to meet this representational challenge.

Moreover, applying this otherwise well understood theory of reasoning about actions and change to capture complex domains has helped to expose more clearly some problems. For example, it has shown a link between non-determinism and an implicit qualification of effect laws via the domain constraints where the explicit representation of an indirect effect law as a direct law can give this higher priority. As mentioned in the introduction, we believe that these considerations are independent of the particular framework that we have used to carry out our study and that they would apply also to other approaches that are based on the same underlying principles for developing a theory for reasoning about actions and change. Differences may arise in the computational aspects of the different frameworks. For this reason we have concentrated more on comparison experiments and have tried to exposed inherent computational problems. Also the propositional nature of the particular framework that we have used could be limiting in the static aspects of the representation, i.e. in the ability to represent knowledge at any single time or situation. But this does not affect the ability to represent the dynamic aspects of a problem domain. At this initial stage of the study we believe that it is methodologically correct to de-couple these two aspects and examine at a later stage how they would affect each other.

Our study of the computational behaviour of representations has suggested some methodological guidelines for modeling complex domains that can help us control their computational performance. These include the balanced use of direct and indirect effect laws and the adoption of an appropriate vocabulary as presented in the previous section. To cross check these comparison results it would be useful to study similar experiments using other languages, e.g. using the Causal Calculator [10] with its SAT based computation. We also need to study how other mechanisms, external to the reasoning, can help focus more the computation to the relevant part of the theory.

More work is needed to understand better what factors can influence the computational performance of a representation and how this would scale up. We also need to develop a stricter methodology on how to choose a vocabulary for a complex domain that would give it good computational properties. In our work so far this choice was mostly motivated by the natural common sense reasoning that we as humans apply to the domain and our use of a specific vocabulary for it. In fact, a distinction between "natural" and "formal" (or compact) representations emerges, where the first type of representation uses a non-minimal enriched vocabulary which helps it encode directly in the domain qualitative solutions to computational problems (e.g. the geometry of a terrain) and thus results to a better computational behaviour. This and its possible link to qualitative reasoning in AI is an interesting problem for future study.

Our work has been mainly motivated with trying to understand a natural computational model of human commonsense reasoning and how this is affected both by issues of representation and methods of computation. In the future we plan to investigate, through a series of cognitive experiments, how natural are these representations and how well they capture the human common sense reasoning in the domain of action and change.

Acknowledgements

This work was partly supported by the European Union KIT project CLSFA (9621109) and IST project SOCS (IST-2001-32530).

References

[1] M. Gelfond C. Baral and A. Provetti, 'Representing actions: Laws, observations and hypotheses', in *JLP 31(1-3), pp. 201-243*, (1997).

[2] ETAI Forum, 'http://www.ida.liu.se/ext/etai/lmw'.

[3] M. Gelfond and V. Lifschitz, 'Action languages', in *ETAI*, volume 3(16), (1998).

[4] E. Giuchiglia and V. Lifschitz, 'An action language based on casual explanation', in *AAAI-98*, pp. 623–630, (1998).

[5] A. Hensechel and M. Thielscher, 'Logic modeling workshop: The traffic world', in *ETAI*, volume 3, nr 7, http://www.ida.liu.se/ea/cis/1999/007, (1999).

[6] V. Lifschitz J. Lee and H. Turner, 'A representation of the zoo world in the language of the causal calculator', in *Proc. Common Sense 2001.*

[7] A.C. Kakas and R.S. Miller, 'A simple declarative language for describing narratives with actions', in *JLP 31(1-3), pp. 157-200*, (1997).

[8] A.C. Kakas and R.S. Miller, 'Reasoning about actions, narratives and ramifications', in *ETAI Vol 1(4), pp. 39-72*, (1998).

[9] A.C. Kakas, R.S. Miller, and F. Toni, 'E-res: Reasoning about actions, events and observations', in *LPNMR'01, pp. 254-266*, (2001).

[10] N. McCain, *Causality in Commonsense Reasoning about Actions*, PhD thesis, University of Texas at Austin, 1997.

[11] M. Thielscher, 'Ramification and causality', in *Artificial Intelligence*, volume 89 (1-2), pp. 317-364, (1997).

[12] J. Lee V. Akman, S. T. Erdogan and V. Lifschitz, 'A representation of the traffic world in the language of the causal calculator', in *Proc. Common Sense 2001*, (2001).

Argument, Dialogue and Decision

Skepticism and Floating Conclusions (invited paper)

John F. Horty
Philosophy Department and
Institute for Advanced Computer Studies
University of Maryland
College Park, MD 20742, USA
horty@umiacs.umd.edu
http://www.umiacs.umd.edu/users/horty

Abstract

The purpose of this paper (published in *Artificial Intelligence* **135**: 55–72) is to question some commonly accepted patterns of reasoning involving nonmonotonic logics that generate multiple extensions. In particular, I argue that the phenomenon of floating conclusions indicates a problem with the view that the skeptical consequences of such theories should be identified with the statements that are supported by each of their various extensions.

On Concise Encodings of Preferred Extensions

Paul E. Dunne
Department of Computer Science
University of Liverpool
Liverpool L69 7ZF
United Kingdom
ped@csc.liv.ac.uk

Abstract

Argument Systems provide a rich abstraction within which divers concepts of reasoning, acceptability and defeasibility of arguments, etc., may be studied using a unified framework. Much work has focused on the so-called *preferred* extensions of such systems, which define the maximal (with respect to $\subseteq$) collectively defensible subsets of arguments within a given system of arguments and attack relationship. In this article we address problems related to the following issue. Identification and enumeration of preferred extensions of an argument system is (under the usual complexity theoretic assumptions) computationally demanding: there may be exponentially many and deciding if a given subset S of $\mathcal{X}$ *does* define a preferred set is CO-NP–complete. For a domain which is questioned 'frequently' it may be acceptable to invest this computational effort once, but having done so it is desirable to encapsulate these data in a form which is compact and allows, for example, questions concerning the acceptability of specific arguments to be dealt with efficiently. In this article we consider two 'plausible' approaches to reducing the complexity of deciding if S is a preferred extension of a system $\mathcal{H}$ both of which assume some initial potentially 'expensive' precomputation, invested to reduce time needed in subsequent queries to the system. The first approach examines 'reasonable encoding' approaches; the second is to determine the subset of all defensible arguments providing these as additional data when attempting to decide if S is a preferred extension. It is shown that if certain properties are required of the encoding scheme, then the former approach is feasible only if NP = CO-NP. In the latter case, we show that, even if provided with information regarding which arguments are credulously accepted, the question of whether a subset of arguments defines a preferred extension remains CO-NP–complete.

Keywords: Argument Systems, Preferred Extension, Computational Complexity

1 Introduction

Since they were introduced by Dung [5], Argument Systems have provided a fruitful mechanism for studying reasoning in defeasible contexts. They have proved useful both to theorists who can use them as an abstract framework for the study and comparison of non-monotonic logics, e.g. [1], and for those who wish to explore more concrete contexts where defeasibility is central. In the study of reasoning in law, for example, they have been used to examine the resolution of conflicting norms, e.g. [11], especially where this is studied through the mechanism of a dispute between two parties, e.g. [9]. The basic definition below is derived from that given in [5].

Definition 1 *An* argument system *is a pair* $\mathcal{H} = \langle \mathcal{X}, \mathcal{A} \rangle$, *in which* $\mathcal{X}$ *is a set of* arguments *and* $\mathcal{A} \subset \mathcal{X} \times \mathcal{X}$ *is the* attack relationship *for* $\mathcal{H}$. *Unless otherwise stated,* $\mathcal{X}$ *is assumed to be* finite, *and* $\mathcal{A}$ *comprises a set of ordered pairs of* distinct *arguments. A pair* $\langle x, y \rangle \in \mathcal{A}$ *is referred to as '*x is attacked by y*' or '*y attacks *(or* is an attacker of*)* x*'.*

For R, S subsets of arguments in the system $\mathcal{H}(\langle \mathcal{X}, \mathcal{A} \rangle)$, *we say that*

a) $s \in S$ *is* attacked *by R if there is some* $r \in R$ *such that* $\langle r, s \rangle \in \mathcal{A}$.

b) $x \in \mathcal{X}$ *is* acceptable with respect to S *if for every* $y \in \mathcal{X}$ *that attacks x there is some* $z \in S$ *that attacks y.*

c) S is conflict-free *if no argument in S is attacked by any other argument in S.*

d) A conflict-free set S is admissible *if every argument in S is acceptable with respect to S.*

e) S is a preferred extension *if it is a maximal (with respect to $\subseteq$) admissible set.*

f) S is a stable extension *if S is conflict free and every argument $y \notin S$ is attacked by S.*

g) $\mathcal{H}$ is coherent *if every preferred extension in $\mathcal{H}$ is also a stable extension.*

An argument x is credulously accepted *is there is* some *preferred extension containing it; x is* sceptically accepted *if it is a member of* every *preferred extension.*

The notation $\mathcal{PE}(\mathcal{H})$ is used to describe the set of all subsets of $\mathcal{X}$ which are preferred extensions of $\mathcal{H}$. Similarly, $\mathcal{SE}(\mathcal{H})$ denotes the set of all stable *extensions of $\mathcal{H}$ and $\mathcal{BE}(\mathcal{H})$ refers to an arbitrary one of these sets. We use n to denote $|\mathcal{X}|$.*

The preferred extensions of an Argument System can be taken as being the consistent positions that can be adopted within the Argument System. Any argument that appears in all preferred extensions will be acceptable in every consistent position, and any argument that appears in no preferred extension cannot be held in any consistent position. This means that the notion can be related to varieties of semantics for non-monotonic reasoning: credulously acceptable arguments will be those that appear in at least one preferred extension, and sceptically acceptable arguments will be those which appear in all preferred extensions. In the context of legal reasoning, the notion allows us to distinguish those arguments which must be accepted, those which can be defended, and those which are indefensible.

To avoid repetition we will subsequently refer to the following decision problems:

PREF-EXT (Preferred Extension)
Instance: An argument system $\mathcal{H}(\mathcal{X}, \mathcal{A})$ and $S \subseteq \mathcal{X}$.
Question: Is $S \in \mathcal{PE}(\mathcal{H})$?
STAB-EXT (Stable Extension)
Instance: An argument system $\mathcal{H}(\mathcal{X}, \mathcal{A})$ and $S \subseteq \mathcal{X}$.
Question: Is S a subset of some $T \in \mathcal{SE}(\mathcal{H})$, i.e. can S be expanded to a stable extension?
PREF-EXT-INF (Preferred Extension given Information)
Instance: An argument system $\mathcal{H}(\mathcal{X}, \mathcal{A})$, $S \subseteq \mathcal{X}$, and $\alpha = \langle a_1, a_2, \ldots, a_n \rangle \in \langle \bot, \top \rangle^n$ an n-tuple of Boolean values such that $a_i = \top$ if and only if the argument x_i is credulously accepted in $\mathcal{H}$.
Question: Is $S \in \mathcal{PE}(\mathcal{H})$?
STAB-EXT-INF (Stable Extension given Information)
Instance: An argument system $\mathcal{H}(\mathcal{X}, \mathcal{A})$, $S \subseteq \mathcal{X}$, and $\alpha = \langle a_1, a_2, \ldots, a_n \rangle \in \langle \bot, \top \rangle^n$ an n-tuple of Boolean values such that $a_i = \top$ if and only if the argument x_i is credulously accepted in $\mathcal{H}$.
Question: Is S a subset of some $T \in \mathcal{SE}(\mathcal{H})$, i.e. can S be expanded to a stable extension?

Before proceeding it may be useful to consider in more detail the concepts of 'preferred' versus 'stable' extensions of an argument system. Both [5] and [1] offer a view of preferred extensions as providing a more general construct than stable extensions: thus any stable extension is also preferred but the converse is not always true. A significant difference between the two models is that whereas some argument systems may have no stable extension, it is always the case that a preferred extension extension exists since the empty set is always admissible. This difference raises a number of general questions that are discussed in some detail in [5]. In particular,

1. Are there 'natural' or 'meaningful' argument systems with no stable extensions?

2. Are there 'natural' systems whose set of stable extensions form a strict (non-empty) subset of the set of preferred extensions?

Through consideration of a particular n-player game, [5, p.336], argues that studies of Lucas and Shubik[13] suggest '*stable semantics do not capture the intuitive semantics of every meaningful argumentation system*'. As a further example, using a variant of the *Stable Marriage Problem*, [5, p.338] exhibits a concrete 'natural' system which has no stable extension. While both examples suggest a positive answer to the first of the two questions raised, neither treats the second question. Instead, [5, Defn. 31, p. 332] introduces the concept of *coherence* to describe systems for which every preferred extensions is also stable, discussing forms of argumentation system whose instantiation guarantees coherence. Recent work of Dunne and Bench-Capon[7], however, indicates that even when restricted to the context of finitely presented argument systems, deciding if a given system $\mathcal{H}(\mathcal{X}, \mathcal{A})$ is coherent is 'likely' to be extremely hard. One consequence of the proof employed in [7], is that it, naturally, gives rise to an infinite class of argument systems having a non-empty set of stable extensions but which are nonetheless incoherent: i.e. there is a supportable case that the answer to the second question raised is also positive.

A major difficulty that is encountered within these formalisms is the computational intractability of several decision problems that arise: results of Dimopoulos and Torres [4] indicate that deciding if p is credulously accepted in $\mathcal{H}$ is NP–complete and that PREF-EXT is CO-NP–complete (even when S is the empty set). Similarly deciding if $\mathcal{H}$ has any stable extension (i.e. the decision problem $\mathcal{SE}(\mathcal{H}) \neq$

$\emptyset$?) is NP-complete (notice that this is the special case of STAB-EXT when $S = \emptyset$). Dunne and Bench-Capon [7] proved that deciding coherence is $\Pi_2^{(p)}$–complete, deducing the same complexity classification for sceptical acceptance as a consequence. Related work, described in [8], has shown that the sound and complete reasoning method for credulous argumentation introduced by Vreeswijk and Prakken [14] in which reasoning proceeds via a dialogue game requires an exponential number of moves to resolve some disputes. Examining differing concepts of 'acceptance' in various non-monotonic Logics, [2, 3], indicate that decision problems predicated on preferred extensions are complete for divers levels of the polynomial–time hierarchy ranging from Σ_1^p (i.e. NP) in the case of credulous reasoning in the Logic Programming formalism (LP) to Π_4^p in the case of sceptical reasoning within Autoepistemic Logic (AEL).

The concern of this paper is to consider some further contrasts between computational properties of preferred and stable extension sets, arising from the following scenario. Suppose one is given a specific argument system $\mathcal{H}(\mathcal{X}, \mathcal{A})$ which describes a 'frequently' used application, for example one incorporating complex legal data in which the justification for various different positions may have to be assessed. Rather than deciding acceptability and support for an argument on each new query to $\mathcal{H}$, we may be prepared to invest some computational effort *once* in the hope that the information elicited as a result may help in reducing the time taken for subsequent queries. For example, one could compute all sets in $\mathcal{BE}(\mathcal{H})$ and then have queries on $\mathcal{H}$ performed with respect to a *representation* (encoding) of this set of subsets. There are, of course, several properties which such a representation should, ideally, satisfy. Suppose $\eta(\mathcal{H})$ is some encoding of $\mathcal{BE}(\mathcal{H})$. We define, informally, two such properties that are the main focus of this paper:

R1. η is *terse*, i.e. $|\eta(\mathcal{H})|$ – the number of *bits* needed – is polynomially bounded in n.

R2. η is *extension tractable*, i.e. given any $S \subseteq \mathcal{X}$, the question $S \in \mathcal{BE}$ can be decided from $\eta(\mathcal{H})$ in time polynomial in $|\eta(\mathcal{H})|$.

One obvious representation scheme is simply to use a table $T_{\mathcal{H}}$ of $|\mathcal{BE}(\mathcal{H})|$ rows, each row being n bits in length, so that if $\langle S_1, \ldots, S_k \rangle$ is an ordering of $\mathcal{BE}(\mathcal{H})$, then $T_{i,j} = 1$ if and only if $x_j \in S_i$. While this representation meets the criterion specified by R2, it will fail to satisfy R1 in those cases where $|\mathcal{BE}(\mathcal{H})|$ is superpolynomial in n. We note that systems may be defined where this number is $\Omega(3^{n/3})$. Alternatively, the system $\mathcal{H}$ itself is a representation of $\mathcal{BE}(\mathcal{H})$: while satisfying R1 it is, however, unlikely to satisfy R2 in the case $\mathcal{BE} = \mathcal{PE}$ (assuming NP $\neq$ CO-NP).

Given such examples, a natural question to raise is whether these extremes are inherent, or are there representation formalisms that are terse *and* extension tractable – a property we subsequently refer to as *concise*. In the next Section we formalise these concepts and, in Section 3, prove some basic results concerning them. In particular it is shown that in the case of $\mathcal{PE}(\mathcal{H})$ 'effective' concise encodings are not possible, in general, unless NP = CO-NP. In contrast, concise encoding schemes for $\mathcal{SE}(\mathcal{H})$ are easy to construct. We note that this provides another example of a property which is considerably 'easier' under stable semantics than under preferred semantics for argument systems, cf. [2, 3]. A further indication of the computational difficulties arising in considering preferred extensions is given in the concluding result of Section 3 where the problem PREF-EXT-INF is shown to be CO-NP–complete: thus, even if the defensibility status for every argument in $\mathcal{X}$ is supplied as part of an instance, the problem of deciding whether a given subset S is a preferred extension does not become any easier, i.e. remains CO-NP–complete. Discussion and conclusions occupy Section 4.

2 Definitions

In the remainder of this paper the following notational conventions are used.

$\mathcal{X}_n$ is a set of n arguments $\{x_1, x_2, \ldots, x_n\}$.

For an n element set X_n, $\wp(X_n)$ is the set of all subsets of X_n.

The following definition formalises our abstract concept of *encoding scheme* for the set $\mathcal{BE}$ in an argument system $\mathcal{H}$.

Definition 2 *A* $\mathcal{BE}$ encoding scheme *is a pair* $\langle \eta, P \rangle$ *where*

$$\eta : \{\mathcal{H}(\mathcal{X}_n, \mathcal{A}) : \mathcal{H} \textit{ is an argument system}\} \rightarrow \{0,1\}^*$$

is a mapping from argument systems to finite binary words, and P *is a* deterministic *Turing machine program, that takes as input a pair* $\langle \eta(\mathcal{H}), S \rangle$ *in which* $S \subseteq \mathcal{X}_n$, *accepting if and only if* $S \in \mathcal{BE}(\mathcal{H})$.

Before proceeding there are several points that should be noted. First we observe that it is not insisted that schemes be *uniform*, i.e. it is *not* required that there is an algorithm which given $\mathcal{H}$ computes $\eta(\mathcal{H})$. Of course, in 'practical' schemes, one would wish to have some mechanism for automating this translation. In order to capture some sense of 'practical' scheme, we introduce the notion of *verifiability*

Definition 3 *A* verifiable $\mathcal{BE}$ encoding scheme *is a triple* $\langle \eta, P, Q \rangle$ *in which* $\langle \eta, P \rangle$ *is a* $\mathcal{BE}$ *encoding scheme and* Q *a (non-deterministic) Turing Machine program, that is given* $w \in \{0,1\}^*$ *and* $\mathcal{H}(\mathcal{X}, \mathcal{A})$ *as input, accepting if* $w = \eta(\mathcal{H})$.

Secondly, the definition provides a foundation for introducing more 'sophisticated' schemes other than binary words.

With this abstract idea of encoding scheme we can formalise the notions of *terse* and *extension tractable* outlined earlier.

Definition 4 *Let $\langle \eta, P, Q \rangle$ be a verifiable $\mathcal{BE}$ encoding scheme. We say $\langle \eta, P \rangle$ is* terse *if there is a constant k, such that for all $\mathcal{H}(\mathcal{X}_n, \mathcal{A})$, $|\eta(\mathcal{H})| \leq n^k$; it is* extension tractable *if there is a constant k such that: given $\eta(\mathcal{H})$ and $S \in \wp(X_n)$ as input, P decides if $S \in \mathcal{BE}(\mathcal{H})$ taking at most $(n + |\eta(\mathcal{H})|)^k$ steps. Finally, $\langle \eta, P, Q \rangle$ is* effective *if there is a constant k for which Q has an accepting computation of $w = \eta(\mathcal{H})$ in non-deterministic time bounded by $(n + |w|)^k$. A $\mathcal{BE}$ encoding scheme $\langle \eta, P \rangle$ is* concise *if it is both terse and extension tractable. A* verifiable *$\mathcal{BE}$ encoding scheme $\langle \eta, P, Q \rangle$ is* usefully concise *if it is terse, extension tractable, and effective.*

In 'practical' terms *usefully concise* verifiable $\mathcal{BE}$ encoding schemes, define the 'ideal' representation form: if $\langle \eta, P, Q \rangle$ is usefully concise then one can describe $\mathcal{BE}(\mathcal{H})$ in its entirety using only a 'small' amount of space – since η is *terse*; one can determine efficiently (in terms of $|\eta(\mathcal{H})|$ and $|S|$) if $S \in \mathcal{BE}(\mathcal{H})$ – since the scheme is *extension tractable*; and, finally, one may test if an arbitrary w does, indeed, describe the encoding $\eta(\mathcal{H})$ for a given $\mathcal{H}$.

We now present some examples of encoding approaches.

2.0.1 Tabular Representation

Given $\mathcal{H}$, $tab(\mathcal{H})$ is the $n|\mathcal{BE}(\mathcal{H})|$-bit word in which bit $t_{(i-1)n+j} = 1$ if and only if $x_j \in S_i$ where $1 \leq j \leq n$ and $\langle S_1, S_2, \ldots, S_r \rangle$ is an ordering of $\mathcal{BE}(\mathcal{H})$. If the algorithm P in $\langle tab, P \rangle$ is chosen to be an appropriate table look up method, then $\langle tab, P \rangle$ is extension tractable. It is not, however, terse.

2.0.2 Representations via Propositional Logic Functions

Given any $\mathcal{H}(\mathcal{X}_n, \mathcal{A})$ there is a unique propositional logic function, $f_{\mathcal{H}}(X_n)$ definable from $\mathcal{BE}(\mathcal{H})$ as follows. For any $S \in \wp(\mathcal{X}_n)$ let the instantiation, α_S of the propositional variables X_n be $x_i = \top$ if $x_i \in S$ and $x_i = \bot$ if $x_i \notin S$. The function $f_{\mathcal{H}}(X_n)$ takes the value $\top$ on exactly those instantiations α_S for which $S \in \mathcal{BE}(\mathcal{H})$.

Given this approach, any representation formalism for arbitrary n-argument propositional logic functions serves as a basis for a $\mathcal{BE}$ encoding scheme, e.g. truth-tables, propositional formulae over a finite complete basis, etc.

3 Properties of Usefully Concise Encoding Schemes

We first observe that construction of usefully concise encoding schemes for stable extensions is trivial: the problem of deciding, given $\mathcal{H}(\mathcal{X}, \mathcal{A})$ and $S \subseteq \mathcal{X}$ whether $S \in \mathcal{SE}(\mathcal{H})$ is polynomial-time solvable; therefore since $\mathcal{H}(\mathcal{X}, \mathcal{A})$ already defines a terse encoding of $\mathcal{SE}(\mathcal{H})$ with an appropriate decision algorithm we have a concise $\mathcal{SE}$ encoding scheme. This can be extended to give a usefully concise scheme, by encoding $\mathcal{H}$ as its n^2 element adjacency matrix, so that $w = \eta(H)$ is decided in $|w|$ steps.

In contrast to the easy construction above, usefully concise encoding schemes for *preferred* extensions are 'unlikely' to exist.

Theorem 1 *If* NP $\neq$ CO-NP *then usefully concise $\mathcal{PE}$ encoding schemes do not exist.*

Proof. Suppose NP $\neq$ CO-NP and that for the sake of contradiction, $\langle \eta, P, Q \rangle$ is a usefully concise $\mathcal{PE}$ encoding scheme. We show that $\langle \eta, P, Q \rangle$ can be used as the basis of an NP decision method for PREF-EXT. Since this problem is CO-NP–complete it follows that the existence of such a decision method would imply NP = CO-NP. Since $\langle \eta, P, Q \rangle$ is terse there is some constant k such that $|\eta(\mathcal{H})| \leq n^k$ for any n-argument system $\mathcal{H}$. Our NP algorithm is as follows: given an instance $\langle \mathcal{H}(\mathcal{X}, \mathcal{A}), S \rangle$ of PREF-EXT non-deterministically choose a sequence $\beta(\mathcal{H})$ of (at most) $|\mathcal{X}|^k$ bits. Then simulate Q on input $\langle \mathcal{H}, \beta(\mathcal{H}) \rangle$. If $\beta(\mathcal{H}) = \eta(\mathcal{H})$ Q will have an accepting computation of polynomial length (since Q is effective). Finally, the program, P, is run with input $\langle \beta(\mathcal{H}), S \rangle$. Since $|\beta(\mathcal{H})| \leq |\mathcal{X}|^k$ and P is a deterministic polynomial time computation, the (non-deterministic) algorithm runs in time polynomial in $\mathcal{X}$. To see that the algorithm accepts instances for which $S \in \mathcal{PE}(\mathcal{H})$ it suffices to observe that *if* $S \in \mathcal{PE}(\mathcal{H})$ then there is *some* choice of $\beta(\mathcal{H})$ that will correspond to $\eta(\mathcal{H})$, be accepted by Q and on which the extension tractable algorithm P will accept $\langle \beta(\mathcal{H}), S \rangle$.

It should be noted that the argument used in the proof requires the assumption that $\langle \eta, P, Q \rangle$ is effective. The reason being that if P were invoked directly on the word $\beta(\mathcal{H})$ then *every* instance would be accepted: given a subset S there is certainly some argument system, $\mathcal{G}$ for which $\mathcal{PE}(\mathcal{G}) = \{S\}$ – the system with $|S|$ isolated arguments – and if $\beta(\mathcal{H}) = \eta(\mathcal{G})$ then the instance is accepted regardless of whether $S \in \mathcal{PE}(\mathcal{H})$.

Theorem 1 indicates that even if one is prepared to invest considerable computational effort in *constructing* an encoding $\eta(\mathcal{H})$, such effort will not aid in testing $S \in \mathcal{PE}(\mathcal{H})$ if the encoding form is terse *and* effective.

The problem, PREF-EXT-INF in allowing knowledge regarding the set of credulously accepted arguments to be given for free, can be seen as defining an alternative *terse* encoding scheme. We note that this scheme is *not* effective (in our usage) (assuming NP $\neq$ CO-NP) since in the encoding $\langle \mathcal{H}, \alpha \rangle$, should any bit of α be $\perp$ indicating the associated argument is *not* credulously accepted, we cannot test $w = \eta(\mathcal{H})$ using an NP computation.

Thus, since determining the set of credulously accepted arguments in a system, may at worst involve similar computational expenditure to that of enumerating preferred extensions, the complexity of the problem PREF-EXT-INF is of some interest: the implied encoding scheme is not one which is within the scope of Theorem 1. Our next result shows that PREF-EXT-INF is no easier than PREF-EXT.

Theorem 2 PREF-EXT-INF *is* CO-NP*–complete.*

Proof. Membership in CO-NP is immediate from the fact that PREF-EXT $\in$ CO-NP. To show that PREF-EXT-INF is CO-NP–hard, we give a reduction from the problem of deciding is a propositional formula in 3–CNF is *unsatisfiable*: 3-UNSAT. Let

$$\Phi(X_n) = \bigwedge_{i=1}^{m} C_i \;=\; \bigwedge_{i=1}^{m} (y_{i,1} \vee y_{i,2} \vee y_{i,3})$$

be an instance of 3-UNSAT, so that each $y_{i,j}$ is a literal from $\{x_1, \ldots, x_n, \neg x_1, \ldots, \neg x_n\}$. Let $\{x_{n+1}, x_{n+2}\}$ be two new propositional variables (i.e. not belonging to X_n) and consider the CNF formula $\Psi(X_n, x_{n+1}, x_{n+2})$ defined as,

$$\bigwedge_{i=1}^{m} (C_i \vee \neg x_{n+1} \vee x_{n+2}) \wedge (C_i \vee x_{n+1} \vee \neg x_{n+2})$$

The following properties of $\Psi(X_n, x_{n+1}, x_{n+2})$ are easily verified.

a) For each variable $y \in \{X_n, x_{n+1}, x_{n+2}\}$ there is *some* satisfying instantiation for Ψ under which $y = \top$.

b) For each variable $y \in \{X_n, x_{n+1}, x_{n+2}\}$ there is *some* satisfying instantiation for Ψ under which $y = \perp$.

c) There is a satisfying instantiation for Ψ under which $x_{n+1} = \top$ and $x_{n+2} = \perp$ if and only if $\Phi(X_n)$ is satisfiable.

We use $\Psi(X_n, x_{n+1}, x_{n+2})$ to build an argument system $\mathcal{H}_\Psi(\mathcal{V}, \mathcal{A})$. The argument set $\mathcal{V}$ contains $2(m + n + 3)$ arguments labelled

$$\begin{aligned} V = \{\Psi, \chi\} \;&\cup\; \{x_i, \bar{x}_i : 1 \leq i \leq n+2\} \\ &\cup\; \{C_j^{(1)}, C_j^{(2)} \;:\; 1 \leq j \leq m\} \end{aligned}$$

The attack relationship, $\mathcal{A}$, comprises

1. $\{\langle x_i, \bar{x}_i \rangle, \langle \bar{x}_i, x_i \rangle \;:\; 1 \leq i \leq n+2\}$
2. $\{\langle \chi, \bar{x}_i \rangle, \langle \chi, x_i \rangle \;:\; 1 \leq i \leq n\}$
3. $\{\langle y_{i,j}, C_i^{(k)} \rangle \;:\; 1 \leq k \leq 2,\ 1 \leq i \leq m,\ y_{i,j} \in C_i\}$
4. $\{\langle C_i^{(1)}, \Psi \rangle, \langle C_i^{(2)}, \Psi \rangle \;:\; 1 \leq i \leq m\}$
5. $\{\langle \bar{x}_{n+1}, C_i^{(1)} \rangle, \langle x_{n+2}, C_i^{(1)} \rangle \;:\; 1 \leq i \leq m\}$
6. $\{\langle x_{n+1}, C_i^{(2)} \rangle, \langle \bar{x}_{n+2}, C_i^{(2)} \rangle \;:\; 1 \leq i \leq m\}$
7. $\{\langle \Psi, \chi \rangle\}$.

In the system $\mathcal{H}_\Psi(\mathcal{V}, \mathcal{A})$ all *except* the arguments

$$\{\chi\} \cup \{C_j^{(1)}, C_j^{(2)} \;:\; 1 \leq j \leq m\}$$

are credulously accepted. To see this first observe that a satisfying instantiation α of $\Psi(X_n, x_{n+1}, x_{n+2})$ induces a preferred extension of $\mathcal{H}_\Psi$ containing Ψ together with the $n+2$ arguments corresponding to literals evaluating to $\top$ under α. From properties (a) and (b) above we deduce that Ψ is credulously accepted and each of the arguments x_i, $\bar{x}_i$ for $1 \leq i \leq n+2$. In order for an argument $C_i^{(k)}$ to be credulously accepted, an admissible set containing it, would have to contain $\bar{y}_{i,j}$ for each $y_{i,j} \in C_i$. These arguments, however, are attacked by χ: Ψ (the only attacker of χ) cannot be included in an admissible set containing $C_i^{(k)}$. Similarly χ cannot be credulously accepted, since the only attackers of its attacker - Ψ - are $C_i^{(k)}$.

The instance of PREF-EXT-INF constructed from $\Phi(X_n)$ is $\langle \mathcal{H}_\Psi, \{x_{n+1}, \bar{x}_{n+2}\}, \alpha_{\text{CA}} \rangle$ where α_{CA} is the $2(m + n + 3)$ tuple indicating the credulously accepted arguments in $\mathcal{H}_\Psi$ as described in the preceding paragraph.

We claim that $\{x_{n+1}, \bar{x}_{n+2}\} \in \mathcal{PE}(\mathcal{H}_\Psi)$ if and only if $\Phi(X_n)$ is unsatisfiable. First suppose that $\{x_{n+1}, \bar{x}_{n+2}\} \in \mathcal{PE}(\mathcal{H}_\Psi)$. Assume, for the sake of contradiction, that $\Phi(X_n)$ is satisfiable and let α be a satisfying instantiation of X_n for Φ. Consider the subset, S_α of $\mathcal{V}$ given by,

$$\{x_{n+1}, \bar{x}_{n+2}\} \cup \{\Psi\} \cup \{x_i \;:\; \alpha_i = \top\} \cup \{\bar{x}_i \;:\; \alpha_i = \perp\}$$

We claim that S_α is an admissible set (in fact $S_\alpha \in \mathcal{PE}(\mathcal{H}_\Psi)$). To see this first observe that S_α is conflict-free and consider any argument $p \in \mathcal{V}$ that attacks some argument of S_α. If $p = \chi$, then $\Psi \in S_\alpha$ attacks p; if $p = y \in \{x_i : 1 \leq i \leq n+2\}$, then $\bar{y} \in S_\alpha$ attacks p; similarly if $p = \bar{y} \in \{\bar{x}_i : 1 \leq i \leq n+2\}$ then $y \in S_\alpha$ attacks p. If $p \in \{C_i^{(2)} : 1 \leq i \leq m\}$ then x_{n+1} attacks p. We are left with the case $p \in \{C_i^{(1)}\}$. Since α satisfies Φ some literal $y_{i,j}$ of C_i must take the value $\top$ under α. Now we find an attack on $C_i^{(1)}$ with the corresponding x_i or $\bar{x}_i$ argument in S_α. We deduce that $\Phi(X_n)$ satisfiable would contradict the assumption $\{x_{n+1}, \bar{x}_{n+2}\} \in \mathcal{PE}(\mathcal{H}_\Psi)$.

On the other hand, suppose that $\Phi(X_n)$ is unsatisfiable. We show that $\{x_{n+1}, \bar{x}_{n+2}\} \in \mathcal{PE}(\mathcal{H}_\Psi)$.

Certainly $\{x_{n+1}, \bar{x}_{n+2}\}$ is admissible. Consider any $S \subset \mathcal{V}$ for which $\{x_{n+1}, \bar{x}_{n+2}\} \subseteq S$ and with S admissible. Let $p \in S/\{x_{n+1}, \bar{x}_{n+2}\}$. If $p = y \in \{x_i, \bar{x}_i : 1 \leq i \leq n\}$, then in order to counter-attack the attack by χ on y, the argument Ψ must be in S. If $\Psi \in S$, then for each argument $C_i^{(1)}$, S must contain some argument corresponding to a literal $y_{i,j}$ of C_i. Since S is assumed admissible, it follows that the set of literals identified do not conflict. Choosing an instantiation of X_n which makes each of these literals take the value $\top$ will satisfy Φ. This, however, contradicts the assumption that Φ was unsatisfiable. We deduce that $S/\{x_{n+1}, \bar{x}_{n+2}\} = \emptyset$ and therefore $\{x_{n+1}, \bar{x}_{n+2}\} \in \mathcal{PE}(\mathcal{H}_\Psi)$ as claimed.

The following Corollary is easily obtained,

Corollary 1 STAB-EXT-INF *is* NP*–complete.*

Proof. Using the construction of the Theorem, $\{x_{n+1}, \bar{x}_{n+2}\}$ can be developed to a *stable* extension, if and only if $\Phi(X_n)$ is satisfiable.

It is stressed that Theorem 2 and its corollary are addressing *different* decision problems from their counterparts PREF-EXT and STAB-EXT: for the latter problems an instance comprises an argument system $\mathcal{H}(\mathcal{X}, \mathcal{A})$ and a subset S of $\mathcal{X}$; in the problems PREF-EXT-INF and STAB-EXT-INF an instance additionally provides $|\mathcal{X}|$ bits of information, $\alpha_\mathcal{X}$, delineating which arguments of $\mathcal{X}$ are credulously accepted. One indication of the different nature of these problems can be seen by considering the case when $S = \emptyset$: PREF-EXT is CO-NP–complete for this case, however PREF-EXT-INF has an easy polynomial time algorithm by simply checking if any bit of $\alpha_\mathcal{X}$ is $\top$. It is of interest to note, however, that the complexity of STAB-EXT-INF when $S = \emptyset$ is less clear: while we can deduce the absence of any stable extension in $\mathcal{H}$ from $\alpha_\mathcal{X} = \langle \bot \rangle^{|\mathcal{X}|}$, we cannot deduce that one *does* exist if $\alpha_\mathcal{X}$ is not of this form. We conjecture that, in fact, STAB-EXT-INF is NP–complete even for instances $\langle \mathcal{H}, \emptyset, \alpha_\mathcal{X} \rangle$. We note that if correct, this provides a rare, albeit arguably 'unnatural', example of a problem where a decision concerning preferred extensions is 'easier' than the corresponding decision regarding *stable* extensions.

4 Conclusions

The principal focus of this article has been in deriving negative results concerning various mechanisms for reducing the complexity of deciding $S \in \mathcal{PE}(\mathcal{H})$ though 'expensive' precomputation. Thus for a rather general notion of 'useful encoding scheme' it has been shown that such approaches are unlikely to succeed. There remains, of course, the possibility that specific sub-classes of argument system are amenable to concise encoding approaches.

Another direction for further work arises from the fact that the requirement for $\langle \eta, P, Q \rangle$ to be *effective* is rather strong and, as we have seen in Theorem 2 does not apply to what might be regarded as otherwise 'reasonable' approaches. One possible encoding approach concerning which Theorem 1 will not in general apply is the following.

Recall that $f_\mathcal{H}(X_n)$ is the propositional logic *function* for which $f_\mathcal{H}(\alpha_S) = \top$ if and only the subset S of $\mathcal{X}$ indicated by α_S is in $\mathcal{PE}(\mathcal{H})$. Given $f_\mathcal{H}(X_n)$ one might represent this using a suitable *propositional formula.* Of course, there are (infinitely) many equivalent formulae in this regard. Suppose for a propositional formula $\Phi(X_n)$ over the (binary operation) basis of $\{\wedge, \vee, \neg\}$ we define the *length* of $\Phi(X_n)$ as its total number of occurrences of literals., denoting this $|\Phi(X_n)|$. Now consider the following measures.

$$L(\mathcal{H}) =_{def} \min\{|\Phi(X_n)| : \Phi(X_n) \text{ represents } f_\mathcal{H}(X_n)\}$$

$$L(n) = \max_{\mathcal{H}(\mathcal{X}_n, \mathcal{A})} \{L(\mathcal{H})\}$$

Informally, $L(n)$ is given by: for each different n-argument $\mathcal{H}$ identify the shortest formula representing $f_\mathcal{H}$; $L(n)$ is then the maximum of these values. The function $L(n)$ is well-defined (and computable, albeit by highly infeasible mechanisms). It is certainly the case that encoding $\mathcal{PE}(\mathcal{H})$ by a propositional formula $\Phi(X_n)$ is an *extension tractable* approach: to test $S \in \mathcal{PE}(\mathcal{H})$ simply evaluate $\Phi(\alpha_S)$. This is unlikely to be *effective*: given $w \in \{0,1\}^*$, even though (assuming some standard encoding of propositional formulae, e.g. [6, p.273]) one could determine whether w encodes *some* Φ, it is unlikely that one can test within NP if this represents $f_\mathcal{H}$. In summary we have *open* the possibility that propositional formulae offer *concise* encodings of $\mathcal{PE}(\mathcal{H})$ since such are not ruled out by Theorem 1.

Problem 1 *Do propositional formulae admit concise $\mathcal{PE}$ encodings, i.e. is there any $k \in \mathbb{N}$ such that $L(n) = O(n^k)$?*

There are two points worth considering concerning Problem 1. First, the classic information-theoretic argument of Riordan and Shannon[12], (cf [6, pp.273–274]) does not help in proving superpolynomial lower bounds: even if it is assumed that each distinct n argument system encodes a different preferred extension set, the lower bound on $L(n)$ implied by this is only $n^2/\log n$. A second point concerns the class of propositional functions being addressed: work of Lupanov[10] indicates how formulae for propositional functions satisfying certain 'inheritance' properties can be constructed. This approach – the so-called "Principle of Local Coding" – allows 'small' formulae to be built for suitable classes of functions provided that specific small formulae used in the approach can also be built.

An overview of the mechanism is given in [6, Chapter 3, pp. 136–8]. If one considers the class $\mathcal{G} = \cup_{n=1}^{\infty}\{\mathcal{F}_n\}$ in which $\mathcal{F}_n$ is the set of n-variable propositional functions $f_{\mathcal{H}}$ for systems $\mathcal{H}$ of n arguments, then it may be possible to show that $\mathcal{G}$ has the required 'inheritance' property and that this, given suitable subsidiary formula constructions *might* lead to concise encoding schemes. It should be noted, however, even if this route is possible, it is likely to be the case that generating and verifying the correctness of resulting formulae may well be computationally demanding (although such a process need only be performed *once* with respect to any given $\mathcal{H}$).

As a final open question we mention the following decision problem. As our starting point for building an encoding of $\mathcal{PE}(\mathcal{H})$ we have assumed that the argument system $\mathcal{H}$ is provided as the instance. One might 'relax' this and assume that an arbitrary *subset* of $\wp(X_n)$ is given and we wish to encode *only* those subsets that correspond to $\mathcal{PE}(\mathcal{H})$ for *some* $\mathcal{H}(\mathcal{X}, \mathcal{A})$. Thus, we have the following decision problem:
REALISABLE
Instance: $\mathcal{S} = \{S_1, S_2, \ldots, S_k\} \subseteq \wp(\mathcal{X}_n)$.
Question: Does there exist an argument system $\mathcal{H}(\mathcal{X}_n, \mathcal{A})$ for which $\mathcal{PE}(\mathcal{H}) = \mathcal{S}$?

Problem 2 *Determine the complexity classification of* REALISABLE.

We conjecture that REALISABLE $\in$ P, which would follow by proving that the following condition (which is easily shown to be necessary) is also sufficient for $\mathcal{S}$ to be realisable.

$$\forall T \subseteq \mathcal{X}_n \forall \{x, y\} \subseteq T$$
$$(\exists S_i \in \mathcal{S}\ S_i \subset T) \Rightarrow (\forall S_i \in \mathcal{S}\ \{x, y\} \not\subseteq S_i)$$

This condition can be tested in polynomial-time simply by restricting T to range over those supersets of $S_i \in \mathcal{S}$ formed by adding a single new argument.

References

[1] A. Bondarenko, P. M. Dung, R. A. Kowalski, and F. Toni. An abstract, argumentation-theoretic approach to default reasoning. *Artificial Intelligence*, 93(1–2):63–101, 1997.

[2] Y. Dimopoulos, B. Nebel, and F. Toni. Preferred arguments are harder to compute than stable extensions. In D. Thomas, editor, *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99-Vol1)*, pages 36–43, San Francisco, 1999. Morgan Kaufmann Publishers.

[3] Y. Dimopoulos, B. Nebel, and F. Toni. Finding admissible and preferred arguments can be very hard. In A. G. Cohn, F. Giunchiglia, and B. Selman, editors, *KR2000: Principles of Knowledge Representation and Reasoning*, pages 53–61, San Francisco, 2000. Morgan Kaufmann.

[4] Y. Dimopoulos and A. Torres. Graph theoretical structures in logic programs and default theories. *Theoretical Computer Science*, 170:209–244, 1996.

[5] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reason, logic programming, and N-person games. *Artificial Intelligence*, 77:321–357, 1995.

[6] P. E. Dunne. *The complexity of Boolean networks*. Academic Press, 1988.

[7] P.E. Dunne and T.J.M. Bench-Capon. Coherence in finite argument systems. Technical Report ULCS-01-006, Dept. of Comp. Sci., Univ. of Liverpool, (submitted), December 2001.

[8] P.E. Dunne and T.J.M. Bench-Capon. Two party immediate response disputes: Properties and efficiency. Technical Report ULCS-01-005, Dept. of Comp. Sci., Univ. of Liverpool, (submitted), October 2001.

[9] H. Jakobovits and D. Vermeir. Dialectic semantics for argumentation frameworks. In *Proceedings of the Seventh International Conference on Artificial Intelligence and Law (ICAIL-99)*, ACM SIGART, pages 53–62, N.Y., June 1999. ACM Press.

[10] O.B. Lupanov. Complexity of formula realization of functions of logical algebra. *Prob. Kibernetki (in Russian)*, 3:782–811, 1962.

[11] H. Prakken. *Logical Tools for Modelling Legal Argument*. Kluwer Academic Publishers, 1997.

[12] J. Riordan and C.E. Shannon. The number of two–terminal series–parallel networks. *Jnl. Math. and Phys.*, 21:83–93, 1942.

[13] M. Shubik. *Game Theory in the Social Sciences*. MIT Press, Cambridge MA, 1985.

[14] G. Vreeswijk and H. Prakken. Credulous and sceptical argument games for preferred semantics. In *Proceedings of JELIA'2000, The 7th European Workshop on Logic for Artificial Intelligence.*, pages 224–238, Berlin, 2000. Springer LNAI 1919, Springer Verlag.

Strategical considerations for argumentative agents (preliminary report)

Leila Amgoud
IRIT, 118 route de Narbonne, 31062
Toulouse Cedex, France
amgoud@irit.fr

Nicolas Maudet
IRIT, 118 route de Narbonne, 31062
Toulouse Cedex, France
maudet@irit.fr

Abstract

This paper aims at exploring some aspects of the strategical problem of move selection in the context of persuasive dialogues. The proposed heuristics are based on some human strategies issued from natural dialogues. A three-layered approach of strategy is defended, and a preliminary strategical deliberation process (illustrating the interplay of these different levels) is proposed.

Keywords: dialogue, strategies, argumentation.

1 INTRODUCTION

An argumentative agent is an agent whose reasoning capability is based on some argumentative system, *i.e.* an agent who is able to support his beliefs/intentions with some justifying reasons called *arguments*.
Let us consider the case of two argumentative agents involved in some dialogic interaction. Arguably, the normative models of dialogue introduced in the seventies by the pioneering work of Hamblin [9] and best known as *dialectical systems* or *dialogue games* are well-suited to capture such interactions[1]. A dialectical system mainly consists of:

- a set of moves (*e.g.* challenge, assertion, question);
- one commitment store ($\mathcal{CS}$) for each conversant;
- a set of commitment rules defining the effect of the moves on the commitment stores;
- a set of dialogue rules regulating the moves (*i.e.* a protocol);
- some termination rules defining when the dialogue ends.

While the model was primarily introduced to capture persuasive dialogue, it has recently been argued [18] and practicaly shown [3] that the model is adapted to other kinds of interactions (negociation, deliberation, etc.).

A protocol gives the set of moves possibly expected after a previous move. Choosing among these moves is the strategy problem. While protocol is a public notion, strategy is crucially an individualistic matter. There is of course an interesting debate about whether to put things on the normative or on the strategical side (Carlson [7] says: "*Is it a rule or just a good move to answer a question if one can?*"). This is an interesting question, but a design question if any. The protocol issue really depends on the kind of interaction studied, and all what we can say, following Loui [10], is that a fair protocol might leave room for strategy. Interaction between autonomous agents is not proof-theory: the outcome of a dialogue is expectedly non-deterministic, and agents might "lose" dialogue when they are not wise [14].

To make the problem more concrete consider the following simple piece of dialogue:

(1) A: You should ratify the Kyoto Protocol.
B: Why?
A: Because global warming is dangerous, and the Kyoto Protocol is the best means to avoid global warming.

Now imagine that B has (at least) a counter-argument against the fact that the Kyoto Protocol is indeed the

[1] Although the underlying logic of these systems is generally monotonic, see [16].

best means to avoid global warming. First of all, it is not surprising that most systems prevent the agent to rationally accept the reasons in this case (see *e.g.* [3]). But within these rational bounds (and within the normative bounds of the protocol) what should agent B prefer: continue to challenge the opponent or rather propose its counter-argument? And then: which element of the support is it best to challenge? Which counter-argument should be presented? And again would the choice be the same if the time of the debate is limited?

Strategy depends on a collection of factors uneasily captured: protocol, goals of the players, resource allowed, etc. A comprehensive and exhaustive account of this notion is of course out of the scope of the paper. However, this paper aims at exploring some aspects of this issue. In the next section, we present the kind of interaction studied here, and we exhibit evidence of strategy and heuristics inspired from human dialogues. Section 3 introduces our formal framework, and adapts the notion of acceptability of arguments which forms the backbone of our approach. This allows in turn to propose some strategies adapted to artificial agents. The paper ends with a discussion.

2 STRATEGY IN PERSUASIVE DIALOGUE

This section introduces the kind of interaction studied in this paper (thus defining the roles and goals of the agents), and presents some evidence of strategies inspired from naturally occuring dialogues.

2.1 CRITICAL DISCUSSION

In the case of persuasive dialogue [14], an agent (the *proponent*) has the role to defend its point of view while the *opponent* tries to attack it. This does not involve however that the opponent is reduced to only challenge the proponent: the opponent can propose counter-arguments, and this *may* (see [15]) put the burden to defend some facts on its side. According to [18], critical discussion can be considered as a special case of persuasive dialogue in that each protagonist has in the initial situation the burden to prove its own point of view on the (conflicting) issue.

2.2 EVIDENCE FOR STRATEGIES IN DIALECTICAL INTERACTIONS

Moore [13] has studied how natural critical discussions are constrained by the rule of the DC dialectical system [11]. Studying the resulting dialogues, he concludes that strategy is best analysed a three levels:

1. maintaining focus of the dispute;
2. building its point of view or destroying the opponent's one;
3. select method to fulfill the objective set at level 1 and 2.

This 3-level account appropriatly captures the classical distinction made between strategy and tactic. Strategy involves deliberation (*e.g.* planning the line of argumentation). Tactic is basically the mean to reach the aims fixed at the strategical level. In Moore's approach, level 1 and level 2 refer to strategy, while level 3 refers to tactic. Moore also notes that the dialogue participant only considers levels 1 and 2 if he has the initiative in the conversation (a participant has the initiative if its move is not strongly constrained by the partner's previous move: *e.g* at the beginning of the dialogue or after a statement). He also gives examples of heuristics triggered at level 3 (see Tab.1)

While this helps by giving example of concrete *natural* heuristics, further work needs to be done to computationally capture such notions. How to model "unwelcome" commitments, "good" reasons, or "carefully choosen" support? The aim of the next section is to give formal definitions of some of these notions in order to propose an account of the strategical reasoning of artificial argumentative agents.

3 STRATEGIES FOR ARGUMENTATIVE AGENTS

To start with, we make more precise the notions of protocol and strategy informally introduced in Section 1. The set of argumentative agents is denoted by $\mathcal{A}$. $\mathcal{M}$ is the language used by our agents to communicate. It specifies the kinds of acts the agents can make and the content language used to convey information. Although it may be extended to capture other types of dialogue such as negociation [5], this set at least contains claims (asserting conclusions and reasons), challenges (asking for reasons), and (yes/no) questions [14]. A dialogue protocol is the set of rules governing the high-level behavior of interacting agents. It first defines the allowed replies to each act (given the current state of the commitment stores). Since the update of the commitment stores ($\mathcal{CS}$) is related to the kind of act played by an agent, we can imagine that it is also part of a dialogue protocol. These two components of the

Table 1: Moore's heuristics for DC-like games

Previous move	Prefered following moves
question	answer in such a way to avoid unwelcome commitments give answers for which good reasons can be found answer truthfully
challenge	answer with some carefully choosen support
resolution	retract the least harmful conjunct
statement	if there is contradiction in partner's $\mathcal{CS}$, ask for resolution if there is evidence *vs* any partner's commitment, assert it seek objection in partner's $\mathcal{CS}$ challenge partner's arguments

dialogue protocol can be seen as two different functions denoted respectively by Π and $\mathcal{U}$. The function Π returns, from the last act made (and the current state of the commitment store) the set of legal next acts. Although we will use in this paper only simple protocols (*e.g.* where no backtracking replies are allowed), it is worth pointing out that a particular interest of dialogue-game protocols is to allow flexibility in dialogue structures by relying more on the commitment stores than on dialogue history. The function $\mathcal{U}$ describes how the commitment store of an agent is updated after a given act. We can now formally define a dialogue protocol as follows:

Definition 1 *A dialogue protocol is defined as a tuple* $\mathcal{P} = \langle \mathcal{M}, \Pi, \mathcal{U} \rangle$ *such that:*

- $\mathcal{M}$ *is a set of allowed acts;*
- $\Pi : \mathcal{M} \times \mathcal{CS} \mapsto 2^{\mathcal{M}}$;
- $\mathcal{U} : \mathcal{A} \times \mathcal{M} \times \mathcal{CS} \mapsto \mathcal{CS}$.

The set of moves that an agent might consider at his turn is constrained by some rationality rules. The function $\mathcal{R}$ specifies the preconditions ($\mathcal{C}$) that must be checked by the agent before providing a given act: it returns a boolean saying if the act can rationally be played or not by a given agent.

Definition 2 *The rationality preconditions is defined as a function* $\mathcal{R}$ *s.t.*

- $\mathcal{R} : \mathcal{M} \times \mathcal{A} \times \mathcal{C} \mapsto \{0, 1\}$

Thus, the dialogue protocol and the rationality preconditions specify the different possible replies after a given move. From that, the exact reply that an agent proposes is the result of the strategy. Hence, the strategy depends on the chosen protocol. Formally, we can therefore think of a given strategy for an agent as being a function from the set of moves identified by the protocol and the rationality preconditions to a single move which is then uttered by an agent:

Definition 3 *A* dialogue strategy *is defined as a function* $\mathcal{S}$*:*

$$\mathcal{S} : 2^{\mathcal{M}} \mapsto \mathcal{M}.$$

3.1 LAYERED STRATEGIES

As simple as it is however, the function hides some real difficulties: in particular, the agent must select the kind of act, but also the content to expose, and this depends on the beliefs of the agent and on the current state of the dialogue, among others. We argue in consequence that this notion is best captured at different levels, as suggested by Moore. In the context of our argumentative agents however, we leave aside the level 1 as defined by this author (the matter of maintaining the focus of the dispute) since this would involve more complex protocols than those discussed here —possibly to take into account some notions of relevance [16]. We replace this level with the different profiles of the agents involved in the interaction (a profile will influence the choice made at the other levels). The account that we propose is then layered as follows:

1. define some agent profile;
2. choose to build or destroy;
3. choose some appropriate argumentative content.

This does not indicate however, that the levels are independant. On the contrary, there is an interplay between these levels, and we propose at the end of the section a process that take into account this dependance, and selects the best move to play *w.r.t* to the

strategy defined below. But let us first discuss the different levels separatly.

3.2 THE AGENT PROFILE

Amgoud and Parsons have recently [4] made some proposals towards the definition of broad classes of agent profiles:

- Agreeable: *accept* whenever possible.
- Disagreeable: only *accept* when no reason not to.
- Open-minded: only *challenge* when necessary.
- Argumentative: *challenge* whenever possible.
- Elephant Child: *question* whenever possible.

They recognize however that "although [they] have termed these "strategies" each is only a partial definition of $\mathcal{S}$—a full definition would have to take into account the nature of the previous move and hence the overall protocol for the dialogue." In other words, the agent profile is just the first level of strategy, it needs to be completed with some other levels (and this is what is tentatively done through the other levels of strategies). In this paper, the only agent profile that we consider is some fixed level p of *prudence*. Intuitively, prudent agents will be more reluctant to expose their arguments than non prudent agents.

3.3 BUILD OR DESTROY

At the next level, we will consider that the agent can adopt either a *build* or a *destroy* strategy (b-strategy vs. d-strategy). A difference with Moore's proposal is that the strategy is not necessarily selected towards the main claim of the dialogue (the controversial issue), but possibly towards any statement from the participants' commitment stores.

b-strategy consists of defending some facts in its commitment store;

d-strategy consists of attacking some commitments in the opponent's commitment store.

The question is of course: how does the agent choose to switch from one strategy to another? No satisfying answer can be made in isolation from the other levels.

3.4 THE ARGUMENTATIVE CONTENT

The last level concerns the problem of the (argumentative) content of the act. As suggested in [3], the argumention system might help to improve this level of strategy. Undoubtly, the argumention system has things to say about the notions used by Moore in his heuristics: "good" reasons, selection of the support, and so on. In the following, we will modify the notion of acceptability proposed in [1] in order to make the system really helpful for this level of strategy. We start with a brief description of the framework proposed in [1], which is based on the abstract argumentation frameworks proposed by Dung [8].

3.4.1 Basic Definitions

Definition 4 *An* argument *is a pair (H, h) where h is a formula of a propositional language $\mathcal{L}$ and H a subset of a knowledge base Σ such that i) H is consistent, ii) $H \vdash h$ and iii) H is minimal, so no subset of H satisfying both i) and ii) exists. H is called the* support *of the argument and h is its* conclusion.

In general, since Σ is inconsistent, arguments in $\mathcal{A}(\Sigma)$, the set of all arguments which can be made from Σ, will conflict, and we make this idea precise with the notion of undercutting:

Definition 5 *Let (H_1, h_1) and (H_2, h_2) be two arguments of $\mathcal{A}(\Sigma)$. (H_1, h_1)* undercuts *(H_2, h_2) iff $\exists h \in H_2$ such that $h \equiv \neg h_1$. In other words, an argument is undercut iff there exists an argument for the negation of an element of its support.*

To capture the fact that some facts are more strongly believed (or desired, or intended, depending on the nature of the facts) we assume that any set of facts has a preference order over it which derives from the stratification of the knowledge base Σ into non-overlapping sets $\Sigma_1, \ldots, \Sigma_n$ such that facts in Σ_i are all equally preferred and are more preferred than those in Σ_j where $j > i$. The preference level of a nonempty subset H of Σ, $level(H)$, is the number of the highest numbered layer which has a member in H.

Definition 6 *Let (H_1, h_1) and (H_2, h_2) be two arguments in $\mathcal{A}(\Sigma)$. (H_1, h_1) is* preferred *to (H_2, h_2) according to Pref iff $level(H_1) \leq level(H_2)$.*

The preference order makes it possible to distinguish different types of relation between arguments:

Definition 7 *Let A, B be two arguments of $\mathcal{A}(\Sigma)$.*

- *B* strongly undercuts *A iff B undercuts A and it is not the case that $A \gg^{Pref} B$.*
- *If B undercuts A then A* defends itself *against B iff $A \gg^{Pref} B$.*

- *A set of arguments $\mathcal{S}$* defends *A if for every argument B which undercuts A and A cannot defend itself against, we can find an argument C in $\mathcal{S}$ which strongly undercuts B.*

Henceforth, $C_{undercut,Pref}$ will gather all non-undercut arguments and arguments defending themselves against all their undercutting arguments. In [2], it has been shown that the set $\underline{\mathcal{S}}$ of acceptable arguments of the argumentation system $\langle \mathcal{A}(\Sigma), Undercut, Pref \rangle$ is the least fixpoint of a function $\mathcal{F}$ defined as follows:

$\mathcal{S} \subseteq \mathcal{A}(\Sigma), \mathcal{F}(S) = \{(H,h) \in \mathcal{A}(\Sigma) | (H,h)$ is defended by $\mathcal{S}\}$.

Definition 8 *The set of* acceptable *arguments for an argumentation system $\langle \mathcal{A}(\Sigma)$, Undercut, Pref$\rangle$ is:*

$$\underline{\mathcal{S}} = \bigcup \mathcal{F}_{i\geq 0}(\emptyset) = C_{Undercut,Pref} \cup \left[\bigcup \mathcal{F}_{i\geq 1}(C_{Undercut,Pref})\right]$$

An argument is acceptable if it is a member of the acceptable set.

(2) Let $\Sigma = \Sigma_1 \cup \Sigma_2 \cup \Sigma_3$ with $\Sigma_1 = \{\neg a\}, \Sigma_2 = \{a, a \rightarrow b\}$ and $\Sigma_3 = \{\neg b\}$.

(3) (follows 2)
$(\{\neg a\}, \neg a)$ and $(\{a, a \rightarrow b\}, b)$ are two arguments of $\mathcal{A}(\Sigma)$. The argument $(\{\neg a\}, \neg a)$ undercuts $(\{a, a \rightarrow b\}, b)$. The preference level of $(\{a, a \rightarrow b\}, b)$ is 2 whereas the preference level of $(\{\neg a\}, \neg a)$ is 1, and so $(\{\neg a\}, \neg a) \gg^{Pref} (\{a, a \rightarrow b\}, b)$. The argument $(\{\neg a\}, \neg a)$ is in $C_{Undercut,Pref}$ because it is preferred (according to *Pref*) to the unique undercutting argument $(\{a\}, a)$. Consequently, $(\{\neg a\}, \neg a)$ is in $\underline{\mathcal{S}}$. The argument $(\{\neg b\}, \neg b)$ is undercut by $(\{a, a \rightarrow b\}, b)$ and does not defend itself. On the contrary, $(\{\neg a\}, \neg a)$ undercuts $(\{a, a \rightarrow b\}, b)$ and $(\{\neg a\}, \neg a) \gg^{Pref} (\{a, a \rightarrow b\}, b)$. Therefore, $C_{Undercut,Pref}$ defends $(\{\neg b\}, \neg b)$ and consequently $(\{\neg b\}, \neg b) \in \underline{\mathcal{S}}$.

3.4.2 Levels Of Acceptability

According to the definition of [2], all arguments in $\underline{\mathcal{S}}$ are acceptable. However, the intuition is that they are not *equally* acceptable. The idea is simply that the more defense you need, the more you are exposed to contradiction. In this respect, unattacked arguments are considered stronger than self-defended arguments which are in turn considered stronger than jointly defended arguments. In the class of these defended arguments then, it is considered stronger to be defended with the shortest chain of arguments.

Definition 9 (leveled set of acceptable arguments) *The set of* acceptable *arguments at level l for an argumentation system $\langle \mathcal{A}(\Sigma)$, Undercut, Pref$\rangle$ is:*

$$\underline{\mathcal{S}^l} = \bigcup_{i=0..l} \mathcal{F}_{i\geq 0}(\emptyset)$$

Definition 10 (level of acceptable arguments) *The level l at which the argument is acceptable in the context of some argumentation system $\langle \mathcal{A}(\Sigma)$, Undercut, Pref$\rangle$ is given by the lacc function defined as follows:*

- *lacc: $\mathcal{A}(\Sigma) \mapsto N$ such that:*
  1. *if $A \in C_{Undercut}$ then $lacc(A) = 0$;*
  2. *if $A \in C_{Undercut,Pref}$ then $lacc(A) = 1$;*
  3. *$A \in S^l$ and $A \notin S^{l-1}$ then $lacc(A) = l + 1$.*

Definition 11 (strengh of an argument) *Let A and B be two arguments of $\underline{\mathcal{S}}$. A is considered stronger than B iff $lacc(A) < lacc(B)$.*

(4) (follows 2)
The argument $(\{\neg a\}, \neg a)$ is in $C_{Undercut,Pref}$ because it is preferred (according to *Pref*) to the unique undercutting argument $(\{a\}, a)$. Consequently, $(\{\neg a\}, \neg a)$ is in $\underline{\mathcal{S}}$ and then the level of acceptability of this argument is $lacc(\{\neg a\}, \neg a) = 1$.
The argument $(\{\neg b\}, \neg b)$ is undercut by $(\{a, a \rightarrow b\}, b)$ and does not defend itself. However, $(\{\neg a\}, \neg a)$ undercuts in turn $(\{a, a \rightarrow b\}, b)$ and $(\{\neg a\}, \neg a) \gg^{Pref} (\{a, a \rightarrow b\}, b)$. Therefore, $C_{Undercut,Pref}$ defends $(\{\neg b\}, \neg b)$ and consequently $(\{\neg b\}, \neg b) \in \underline{\mathcal{S}}$. The level of acceptability of this argument $(\{\neg b\}, \neg b)$ is 2. As a consequence, $(\{\neg a\}, \neg a)$ is considered stronger than $(\{\neg b\}, \neg b)$.

The stronger is an argument, the more grounded (the "good" reasons of Moore's strategies) is the conclusion supported by this argument.

3.5 PUTTING THE LEVELS TOGETHER: THE STRATEGY DELIBERATIVE PROCESS

As correctly pointed out by Moore himself, the picture of an agent selecting some move to play w.r.t. to the strategy and then filling the content of the move with a carefully choosen argument is far too simple. We expect that the strategy deliberation is a complex process involving all three levels detailed below. As an example of this interplay, we cite the intuition behind the decision of whether choosing a b-strategy or d-strategy when the agent has the initiative: it is rational to select a b-strategy if there is some *sufficiently* acceptable arguments available to support some fact in its commitment store. The sufficiently acceptable level is defined at level 1 by the agent profile (depending on its attitude, *i.e* its level of prudence, the agent will easily or not commit to a b-strategy), and the level of acceptability of the facts is computed at level 3.

strategy deliberative process As an example of the strategical reasoning of argumentative agents involving these different levels, we propose a simple *strategy deliberative process*. Consider an argumentative agent A with a level of prudence p_a involved in a dialectical interaction with an agent B, through the commitment stores denoted respectively $\mathcal{CS}_A$ and $\mathcal{CS}_B$. We suppose that the agent A has adopted a current strategy **c-strategy** $\in$ {**b-strategy, d-strategy**}. When he has the initiative, the agent can trigger the following algorithm:

1. if **c-strategy= b-strategy** then consider first 2. then 3 else consider 3. then 2.
2. if the agent has some acceptable argument A such that:
   (a) $A = (H, p) \in \underline{\mathcal{S}}$ and $p \in \mathcal{CS}_A$;
   (b) $lacc(A) = l'$ and $l' < p_a$.
   then the agent adopts a **b-strategy**, and plays $assert(H)$[2].
3. if the agent has some acceptable argument A such that:
   (a) $A = (H, p) \in \underline{\mathcal{S}}$ and $\neg p \in \mathcal{CS}_B$;
   (b) $lacc(A) = l'$ and $l' < p_a$.
   then the agent adopts an **d-strategy**, and plays $assert(\neg p)$[3].
4. if the agent has no argument for some fact p in the opponent's commitment store, the adopts a **d-strategy** and plays $challenge(p)$.
5. else the agent selects p in the opponent's commitment store for which he has some acceptable argument A such that:
   (a) $A = (H, p) \in \underline{\mathcal{S}}$ and $p \in \mathcal{CS}_B$;
   (b) $\nexists p' : p' \in \mathcal{CS}_B \wedge$
   $A' = (H', p') \in \underline{\mathcal{S}} \wedge lacc(A) < lacc(A')$
   then the agent adopts a **d-strategy** and plays $challenge(p)$.

The deliberation algorithm proposed above simply captures the intuition that an agent will explore the position of the partner (typically by challenging its position) in order to find a weak point in its line of argumentation. Point 1. simply stabilizes the strategical behaviour of the agent by giving preference to the current strategy (considered first).

3.6 AN EXAMPLE OF DIALOGUE

We conclude with an example dialogue that illustrates the notion presented below. We consider that agent A is a very prudent agent (*e.g.* the level p is 1) whose beliefs and preferences are described by the content of the layered knowledge base given in Tab. 2. We consider that the dialogue starts with the statement by the participants of the controversial issue. We do not give here the detail of the protocol and rationality preconditions used, but provide the useful information through the example. Note that the example is discussed from A's point of view.

Turns 1-2:
A_1: *I think newspapers can't publish this information.*
B_2: *Of course they can.*

The commitment stores are updated in consequence (the statements are added in the $\mathcal{CS}$).

$\mathcal{CS}_A = \{\neg publish\}$
$\mathcal{CS}_B = \{publish\}$

Now A has the initiative. The set of legal moves given by the protocol is $\{assert(H), challenge(publish)\}$. Note that $(assert(\neg publish)$ is not permitted since $\neg publish$ is already in the $\mathcal{CS}$. So the agent A uses the strategy deliberation process to select its move. He first evaluates the level of acceptability of its arguments in favor of $\neg publish$. The argument

[2] In case of different arguments with the same level of acceptability, the agent may prefer to choose the smallest argument in order to restrict exposure to defeaters.

[3] Note that this prepares some forthcoming b-strategies for p.

Table 2: The knowledge base for agent A

Σ	A
1	$\neg agree$ $minister, minister \rightarrow \neg private$
2	$health, private$ $private \wedge \neg agree \rightarrow \neg publish, health \rightarrow \neg(minister \rightarrow \neg private)$

$\mathcal{A}_1 = (\{private, \neg agree, private \wedge \neg agree \rightarrow \neg publish\}, \neg publish)$

has a level of acceptability of 2, since it is attacked by

$\mathcal{A}_2 = (\{minister, minister \rightarrow \neg private\}, \neg private)$

and is reinstated by

$\mathcal{A}_3 = (\{health, health \rightarrow \neg(minister \rightarrow \neg private)\}, \neg(minister \rightarrow \neg private))$

The level of acceptability of the argument in favor of $\neg publish$ (and of its arguments against $publish$) is higher than the level of prudence p. Agent A then adopts a d-strategy and choose to challenge B in order to find some others targets. The dialogue continues as follows:

Turns 3-4:
A_3: Why can they publish this information?
B_4: Because the information is not private, and any public information can be published.

The commitment stores are updated in consequence:

$\mathcal{CS}_A = \{\neg publish\}$
$\mathcal{CS}_B = \{publish, \neg private, \neg private \rightarrow publish\}$

Since the current strategy of agent A is a d-strategy, the agent compute the levels of acceptability of its arguments against $\neg private$ and $\neg private \rightarrow publish\}$. Agent A has no acceptable argument against $\neg private \rightarrow publish$ and an acceptable argument for $private$:

$\mathcal{A}_4 = (\{private\}, private)$

whose level of acceptability is 2, since attacked by $\mathcal{A}_2$ and reinstated by $\mathcal{A}_3$. Hence agent A stays in a d-strategy and challenges $\neg private$.

Turns 5-6:
A_5: Why is the information public?
B_6: Because the information concerns a minister, and informations concerning a minister are public.

The new state of the commitment stores is:

$\mathcal{CS}_A = \{\neg publish\}$
$\mathcal{CS}_B = \{publish, \neg private, \neg private \rightarrow publish, minister, minister \rightarrow \neg private\}$

Now, using again the strategy deliberation process, the agent will find an acceptable argument against one of the facts of the partner's commitment store $(minister \rightarrow \neg private)$ with a level of acceptability 0, since supported by the argument

$\mathcal{A}_5 = (\{health, health \rightarrow \neg(minister \rightarrow \neg private)\}, \neg(minister \rightarrow \neg private)$

which is not attacked. This score is lower than the prudence level p of the agent, so this allows the agent to prepare a b-strategy by directly contradicting the opponent's these.

Turns 7-8:
A_7: Informations concerning a minister are not necessarily public.
B_8: Why?

$\mathcal{CS}_A = \{\neg publish, \neg(minister \rightarrow \neg private)\}$
$\mathcal{CS}_B = \{publish, \neg private, \neg private \rightarrow publish, minister, minister \rightarrow \neg private\}$

Agent A has not the initiative, so the process is not used. Agent A answers by giving the support of the argument $\mathcal{A}_5$, and we can suppose for convenience that the partner will in turn accept this argument.

Turns 9-10:
A_9: Informations about the health of minister remain private.
B_{10}: I accept your argument.

Note that a less prudent agent (*e.g.* p=5) would have choosen a b-strategy earlier in the dialogue.

4 DISCUSSION

The paper has presented an exploration of strategies for move selection in persuasive dialogue conducted by argumentative agents. We believe that this area of research will soon become very important, considering the recent developments of commitment-based or dialogue game based protocols for inter-agent communication —see [12] for a survey. The issue is also important when trying to make a computational argumentative agent interact with a human, for instance in educational debate. But it is worth noting that the issue slightly differs in this case: a computational agent too strategically wise might lead to difficulties (as noticed by Moore in p.c.).

At the best of our knowledge however, very few works (besides those reported in this paper) tackle the problem of strategy. A notable exception is a paper of Bench-Capon [6]. Having defined a *Toulmin Dialogue Game Machine*, the author proposes some heuristics for move selection (in a kind of interaction more cooperative than critical discussion). Note first that although the *TDG* machine has an interesting flexibility at the level of dialogue structure[4], the computational player never deliberates at this level (Bench-Capon argues that this is a "matter of personal taste" and "not essential" for strategy). So it leaves aside the level 1 as defined by Moore, as we do in this paper. But the main difference with our work lies in the underlying argumentation framework. Whereas Bench-Capon assumes that the knowledge in the base has initially the form of some defeasible Toulmin-like rules, we use a more general (extended) Dung-like framework which computes the acceptability of the arguments. And this argumentation framework is used to guide the strategical behaviour of the agent. We also defend a three-layered approach of strategy, where this level is used in combination with the agent profile and long-term strategies (build or destroy). An example of strategical deliberation —that we admit very simple as it is— illustrates the interplay of these different levels. Beside the fact that the proposed heuristics are still preliminary, we believe that some points deserve further discussion here.

First of all, the approach described in this paper is based on classical propositional logic. It was justified by our previous works in argument-based inter-agent communication [3]. As pointed out by a reviewer, this is however a serious loss of generality since it excludes many types of arguments discussed in the litterature. In further works, we plan to explore means of abstracting from this logic.

Also, the paper has only considered the case of ideal agents. Ideal agents compute (in particular, determine the acceptability of their arguments) for free. But computation takes resources, and for instance spending too much time trying to determine the acceptability of an argument may be a poor strategy. What if the agent cannot conclude within the bound of the resources? Loui suggests for instance that "players can be exhorted to advance their best arguments when time is limited" [10] The role of strategy is even more crucial when taking into account the resource-bounded nature of the agents. For instance, knowing that the partner is close to resource termination will certainly incite to try risky moves, with the hope he will not have enough resource to contradict it. Hence the level of prudence of the agent could vary through the dialogue —to avoid procrastination of attack of the agents.

Another important question that we leave aside in this paper concerns the prefered content order of the selected arguments. We consider here that the agent expose the argument as a whole. Several works has shown the importance of the content ordering of argument [17], and certainly more elaborated strategies should take into ccount this aspect of the problem.

At last, as it is now, the level of acceptability only concerns arguments and not directly the supported facts. This prevents to take into account the intuition that a fact defended by several different acceptable arguments should be prefered for exposure to a fact defended only by an acceptable argument (without considering the levels of acceptability). The question of how this might interfere with the notion of level of acceptability for the arguments defined here is an open issue for future research.

Acknowledgements

We would like to thank D. Moore and the three anonymous reviewers for the quality of their comments about this paper.

References

[1] L. Amgoud. *Contribution a l'intégration des préferences dans le raisonnement argumentatif.*

[4] Basically, a *swith focus* move is introduced, which permits to modify the stack structure of the claims under discussion.

PhD thesis, Université Paul Sabatier, Toulouse, 1999.

[2] L. Amgoud and C. Cayrol. On the acceptability of arguments in preference-based argumentation framework. In *Actes de 14th Conference on Uncertainty in AI*, pages 1–7, 1998.

[3] L. Amgoud, N. Maudet, and S. Parsons. Modelling dialogues using argumentation. In *Proceedings of the 4th International Conference on Multi Agent Systems (ICMAS00)*, Boston, 2000.

[4] L. Amgoud and S. Parsons. Agent dialogue with conflicting preferences. In *Proceedings of the International Workshop on Agent Theories, Architectures and Languages (ATAL01)*, pages 1–17, 2001.

[5] L. Amgoud, S. Parsons, and N. Maudet. Arguments, dialogue, and negociation. In *Proceedings of the 4th European Conference on Artificial Intelligence (ECAI00)*, Berlin, 2000.

[6] T. Bench-Capon. Specification and implementation of toulmin dialogue game. In *Proceedings of the JURIX eleventh Conference on Legal Knowledge-Based Systems*, pages 5–19, 1998.

[7] L. Carlson. *Dialogue games: an approach to discourse analysis.* D. Reidel Publishing company, 1983.

[8] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reason, logic programming, and n-person games. *Artificial intelligence*, 77:321–357, 1995.

[9] C. L. Hamblin. *Fallacies.* Methuen, 1970.

[10] R. P. Loui. Process and policy: Resource-bounded non-demonstrative reasoning. *Computational Intelligence*, 14:1–38, 1998.

[11] J. MacKenzie. Question-begging in non-cumulative systems. *Journal of philosophical logic*, 8:117–133, 1979.

[12] N. Maudet and B. Chaib-draa. Commitment-based and dialogue-game based protocols: new trends in agent communication languages. *Knowledge Engineering Review*, 2002. Submitted.

[13] D. Moore. *Dialogue game theory for intelligent tutoring systems.* PhD thesis, Leeds Metropolitan University, England, 1993.

[14] H. Prakken. On dialogue systems with speech acts, arguments, and counterarguments. In *Proceedings of the 7th European Workshop on Logic for Artificial Intelligence (JELIA)*, Lecture Notes in AI 1919, pages 239–253, 2000.

[15] H. Prakken. Modelling defeasibility in law: logic or procedure? *Fundamentae Informaticae*, 2001.

[16] H. Prakken. Relating protocols for dynamic dispute with logics for defeasible argumentation. *Synthese*, 127:187–219, 2001.

[17] C. Reed and D. Long. Content ordering in the generation of persuasive discourse. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI'97)*, pages 1022–1028, 1997.

[18] D.N. Walton and E.C.W. Krabbe. *Commitment in dialogue: basic concepts of interpersonal reasoning.* State University of New York Press, Albany, NY, 1995.

"Minimal defence": a refinement of the preferred semantics for argumentation frameworks

C. Cayrol
IRIT - UPS
118, rte de Narbonne
31062 Toulouse, France
{ccayrol@irit.fr}

S. Doutre
IRIT - UPS
118, rte de Narbonne
31062 Toulouse, France
{doutre@irit.fr}

M.-C. Lagasquie-Schiex
IRIT - UPS
118, rte de Narbonne
31062 Toulouse, France
{lagasq@irit.fr}

J. Mengin
IRIT - UPS
118, rte de Narbonne
31062 Toulouse, France
{mengin@irit.fr}

Abstract

Dung's abstract framework for argumentation enables a study of the interactions between arguments based solely on an "attack" binary relation on the set of arguments. Various ways to solve conflicts between contradictory pieces of information have been proposed in the context of argumentation, nonmonotonic reasoning or logic programming, and can be captured by appropriate semantics within Dung's framework. A common feature of these semantics is that one can always maximize in some sense the set of acceptable arguments. We propose in this paper to extend Dung's framework in order to allow for the representation of what we call "restricted" arguments: these arguments should only be used if absolutely necessary, that is, in order to support other arguments that would otherwise be defeated. We modify Dung's preferred semantics accordingly: a set of arguments becomes acceptable only if it contains a minimum of restricted arguments, for a maximum of unrestricted arguments.

1 INTRODUCTION

Argumentation is a form of reasoning that has recently captured the interest of many researchers in the Artificial Intelligence community, among others. An abstract framework for studying it has been proposed by Dung [Dung, 1995]. It is based on the assumption that the process of argumentation is built from two types of jobs: the formation of arguments, associated with the interactions that exist between them; and the evaluation of the validity of the arguments. A form of interaction between arguments that is often difficult to ignore is contradiction, since a usual basic measure of the validity of a piece of reasoning is the level of self-contradiction it contains. In Dung's approach, an argumentation framework consists of two elements: a set of arguments, and a binary relation between arguments, so that one can represent an "attack", or contradiction, relation between arguments. Thus, the set of arguments and the attack relation form a graph.

If one accepts that the graph contains all needed information, it is possible to define which arguments are valid, or acceptable, by considering solely this graph. Our intuition tells us that an argument which is not attacked at all should be acceptable. More generally, one can define that an acceptable argument is one which is defended, that is, whose attackers are all attacked by arguments which are themselves acceptable.

When the graph contains cycles, this definition is circular, and one needs a more global approach. A solution is to define acceptable sets of arguments. Arguments in such a set do usually not attack each other, and defend each other: every attacker of an argument in the set is itself attacked by an argument in the set. This definition leads to Dung's "admissible" sets of arguments. Maximal sets of arguments that are acceptable are then called "preferred extensions" of the framework.

These definitions of acceptability make no distinction between arguments, in the sense that defenders are all equally desirable: if an argument can be independently defended by two different arguments, then one may choose any of the two, or even the two arguments together if they do not attack each other, to defend the first one. However, there are problems where a distinction between several levels of desirability of defenders is needed. Suppose for example that our arguments "belong" to two agents: one, the proponent, is trying to make a point, while the other one, the opponent, is trying to refute it. The proponent may have

arguments that should only be disclosed if absolutely necessary (like very personal information).

We propose below to refine Dung's argumentation framework with a distinction between two types of arguments: we consider *unrestricted* arguments, that one can maximize, and *restricted* arguments, the use of which will be kept to a minimum. Restricted arguments should only be used if absolutely necessary, that is, in order to support other arguments that would otherwise be defeated.

Notice that the distinction between restricted and unrestricted arguments is at a defensive level. Other frameworks have been designed in order to authorize varying strengths for the arguments. In the frameworks described in [Amgoud and Cayrol, 1998] and [Prakken and Sartor, 1997], a preference relation among arguments is taken into account when the interactions between arguments are studied: the idea is that in order to be a serious menace, an argument must be preferred to the argument it attacks. In this case, there is no need to minimize the number of arguments of a given strength, except maybe for a purpose of conciseness or of efficiency.

We also propose to allow for a distinction between the arguments of the proponent and those of the opponent: this enables us to ensure, when this is needed, that the arguments of the opponent are never used to defend an argument of the proponent. Note that this distinction is independent from the previous one, between restricted and unrestricted arguments.

The paper is built as follows: our refined argumentation framework is presented in the next section. In particular, we detail the partition of the set of arguments, and provide a new definition for admissibility; we then define a new type of extensions for our refined argumentation framework. We illustrate our definitions with three particular argumentation frameworks in section 3. Finally, we conclude and sketch future works in section 4.

2 FORMALIZATION

We recall the definition of [Dung, 1995]'s argumentation framework.

Definition 1 *[Dung, 1995] An* argumentation framework *is a pair $\mathcal{AF} = (A, R)$ where A is a set of arguments and R is an attack relation between arguments ($R \subseteq A \times A$).*

An argumentation framework is well-founded *if and only if there is no infinite sequence*

$$a_0, a_1, \ldots, a_n, \ldots$$

such that $\forall i, a_i \in A$ and $a_{i+1} R a_i$[1].

An argumentation framework can easily be represented as a directed graph, where vertices are the arguments and edges correspond to the elements of the relation R.

Example 1 *Let $\mathcal{AF}_1 = (A, R)$ be an argumentation framework such that:*

$A = \{u_1, u_2, u_3, u_4, u_5, r_1, r_2, r_3, r_4, o_1, o_2, o_3, o_4, o_5\}$ *and*

$R = \{(o_1, u_1), (o_2, u_2), (u_3, o_2), (o_3, u_4), (u_5, o_4), (r_1, o_2), (r_2, o_3), (o_4, r_3), (o_5, r_4)\}$.

The graph representation of $\mathcal{AF}_1$ is depicted on figure 1.

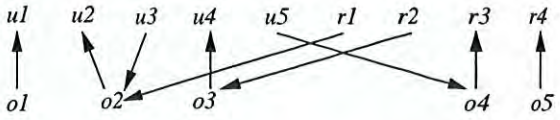


Figure 1: Graph representation of the argumentation framework $\mathcal{AF}_1$

Partitions of the set of arguments of an argumentation framework Let $\mathcal{AF} = (A, R)$ be an argumentation framework. In order to refine some existing semantics for argumentation and considering that A represents the arguments of different agents, the meaning of a partition of A may be the following:

- A simple case is to take into account one agent against the others. So we have a simple partition of A, called `type-1-partition`: $A = F \cup (A \setminus F)$ for a given set of arguments F (corresponding to the arguments of the selected agent);

  the argumentation framework $\mathcal{AF}$ `type-1-` partitioned, given a set of arguments F, is denoted by $\mathcal{AF}_F$;

- the other type of a partition corresponds to the distinction between restricted arguments (which should not be used if possible) and unrestricted arguments. So, we have a special partition of A, called `type-2-partition`, which is a refinement of a `type-1-partition`: given a set F such that $F = F_u \cup F_r$ with $F_u \cap F_r = \varnothing$, $A = F_u \cup F_r \cup (A \setminus (F_u \cup F_r))$ (F_u, F_r are respectively the set

[1] If R is finite, an argumentation framework is well-founded if and only if it does not contain any cycle.

of the unrestricted arguments and the set of the restricted arguments of the selected agent);

the argumentation framework $\mathcal{AF}$ type-2- partitioned, is denoted by $\mathcal{AF}_{F_{ru}}$.

Note that, although every type-2-partition has an underlying type-1-partition, the latter can be a vacuous one: it is possible to set F to be the entire set A.

2.1 ADMISSIBILITY

Let $\mathcal{AF} = (A, R)$ be an argumentation framework, we have the following definitions.

Definition 2 *[Dung, 1995] A set $S \subseteq A$ is* conflict-free *if and only if $\nexists a, b \in S$ such that aRb.*

[Dung, 1995] defines the notion of defence. Defence can be of two kinds: collective or individual. Collective defence is achieved by a set of arguments, individual defence by one argument.

Definition 3 *[Dung, 1995] Let $S \subseteq A$, $a \in A$. S* defends (collectively) *a if and only if $\forall b \in A$, if $bRa, \exists c \in S$ such that cRb. S* defends all its elements *if and only if $\forall a \in S$, if $\exists b \in A$ such that bRa then $\exists c \in S$ such that cRb.*

The following definition is inspired by [Dung, 1995]:

Definition 4 *Let $a, c \in A$. c* is an individual defender of *a if and only if there is a finite sequence $x_0, \ldots, x_{2n}$ such that:*

1. *$a = x_0$ and $c = x_{2n}$, and*
2. *$\forall i, 0 \leq i \leq (2n-1), x_{i+1} R x_i$.*

If $n = 1$, c is a direct individual defender *of a.*

Then, [Dung, 1995] proposes the admissible semantics:

Definition 5 *[Dung, 1995] A set $S \subseteq A$ is* admissible *if and only if S is conflict-free and S defends all its elements.*

Every argumentation framework has at least one admissible set because the empty set is admissible.

Example 2 *The sets $\emptyset$, $\{o_1\}$, $\{o_1, u_2, u_3\}$, $\{o_1, u_2, u_3, u_4, u_5, r_1, r_2, r_3, o_5\}$ are admissible sets of the argumentation framework $\mathcal{AF}_1$ represented on figure 1.*

We refine this admissible semantics in the context of a type-2-partition of A in order to accept restricted arguments only if their presence is justified, that is when they defend at least one unrestricted argument. This refinement is called the *restricted admissibility.*

Let $\mathcal{AF}_{F_{ru}}$ be a type-2-partitioned argumentation framework, we have the following notations and definition:

Notations Given $F \subseteq A$ such that $F = F_u \cup F_r$ and $F_u \cap F_r = \varnothing$ and a set $S \subseteq A$, S_u denotes $S \cap F_u$ and S_r denotes $S \cap F_r$. We say that S_u is the *unrestricted* part of S, and S_r is the *restricted* part of S.

Definition 6 *Let $S \subseteq F$. S is* restrictedly admissible *if and only if:*

1. *S is admissible and*
2. *$\forall x \in S_r$, $\exists y \in S_u$ such that x is an individual defender of y.*

Example 3 *The set of arguments of the argumentation framework $\mathcal{AF}_1$ depicted on figure 1, can be partitioned as follows: let $F = F_u \cup F_r$ with $F_u = \{u_1, u_2, u_3, u_4, u_5\}$, $F_r = \{r_1, r_2, r_3, r_4\}$ and $A \setminus F = \{o_1, o_2, o_3, o_4, o_5\}$. The resulting type-1- (resp. type-2-) partitioned argumentation framework is called $\mathcal{AF}_2$ (resp. $\mathcal{AF}_3$). $\mathcal{AF}_2$ and $\mathcal{AF}_3$ are depicted on figure 2.*

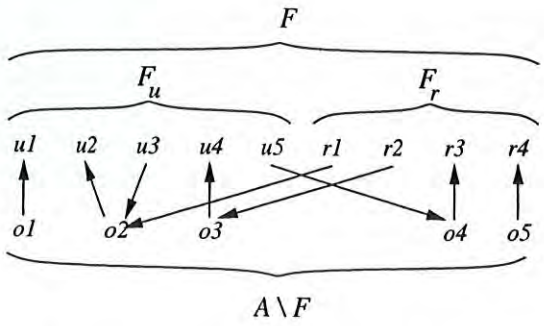


Figure 2: Argumentation frameworks $\mathcal{AF}_2$ and $\mathcal{AF}_3$

$\mathcal{AF}_3$ has several restrictedly admissible sets. For example:
$\emptyset$, $\{u_2, u_3\}$, $\{u_2, u_3, u_4, u_5, r_2\}$, $\{u_2, u_3, u_4, u_5, r_1, r_2\}$.
The set $S = \{u_5, r_3\}$ is not restrictedly admissible since r_3 does not defend any argument of the unrestricted part of S (that is $\{u_5\}$).

2.2 EXTENSIONS

Let $\mathcal{AF} = (A, R)$ be an argumentation framework. [Dung, 1995] has also defined the preferred semantics in which an acceptable set of arguments is called a *preferred extension.*

Definition 7 *[Dung, 1995] A set $S \subseteq A$ is a* preferred extension *of $\mathcal{AF}$ if and only if S is maximal for set-inclusion among the admissible sets of $\mathcal{AF}$.*

We recall:

Property 1 *[Dung, 1995]*

1. *Every admissible set of $\mathcal{AF}$ is included in a preferred extension of $\mathcal{AF}$.*
2. *Every argumentation framework has at least one preferred extension.*
3. *Every well-founded argumentation framework has one and only one preferred extension.*

Example 4 *$\mathcal{AF}_1$, $\mathcal{AF}_2$ and $\mathcal{AF}_3$ have only one preferred extension:*

$$\{o_1, u_2, u_3, u_4, u_5, r_1, r_2, r_3, o_5\}.$$

In order to privilege some arguments of A, and to facilitate the computation of the extensions under the new semantics, we define a *preferred extension on* a given set X. Note that it implies the use of a `type-1-partition` of A: $A = X \cup (A \setminus X)$.

Definition 8 *Let $\mathcal{AF} = (A, R)$ be an argumentation framework and let $X \subseteq A$. A* preferred extension on *X of $\mathcal{AF}$ is an admissible set $S \subseteq X$ such that S is $\subseteq$-maximal on X, i.e. $\forall Y \subseteq (X \setminus S)$, $Y \cup S$ is not admissible.*

Note that :

- The preferred extensions on A in the sense of definition 8 are the preferred extensions in the sense of definition 7.
- A preferred extension on X is not simply the intersection of X and of a preferred extension of $\mathcal{AF}$. For example, let $\mathcal{AF} = (\{a, b, c\}, \{(c, b), (b, a)\})$ be an argumentation framework and $X = \{a\}$, the preferred extension of $\mathcal{AF}$ is $\{c, a\}$, its intersection with X is $\{a\}$ and the only preferred extension on X is $\varnothing$!

The following results hold:

Property 2 *Let $\mathcal{AF} = (A, R)$ be an argumentation framework and let $X \subseteq A$.*

1. *Every admissible set included in X is included in a preferred extension on X.*
2. *Every preferred extension on X is included in a preferred extension of $\mathcal{AF}$.*
3. *Let (X, R') be the restriction of $\mathcal{AF}$ to X[2]. If (X, R') is a well-founded argumentation framework then $\mathcal{AF}$ has one and only one preferred extension on X.*

The proof of this property (and the detailed proofs of all the properties of this paper, except the property 1 which is given in [Dung, 1995]) is given in [Cayrol et al., 2002].

Example 5 *$\mathcal{AF}_2$ has one preferred extension on F:* $\{u_2, u_3, u_4, u_5, r_1, r_2, r_3\}$.

Preferred extensions are interesting in several respects:

- they provide a good "summary" of the admissible sets, since every subset of a preferred extension can be completed to an admissible set, and every admissible set is subset of at least one preferred extension;
- the intersection of the preferred extensions can be interpreted as being the set of arguments that cannot be defeated.

Similar remarks can be made for other types of extensions of argumentation frameworks, like stable extensions for example. Similarly, we want to define a family of "best" restrictedly admissible sets. In the context of a `type-2`-partitioned argumentation framework, the idea is to select the restrictedly admissible sets which contain a minimum of restricted arguments for a maximum of unrestricted arguments. In order to do so, we define the following relation:

Definition 9 *Let $\mathcal{AF}_{F_{do}}$ be a* `type-2`*-partitioned argumentation framework. Let S_1 and S_2 be two subsets of F. S_2 is $\prec$-*better than *S_1 (denoted by $S_1 \prec S_2$) if and only if $S_{1u} \subset S_{2u}$, or $S_{1u} = S_{2u}$ and $S_{2r} \subseteq S_{1r}$.*

The relation $\prec$ is a partial order on the set of the subsets of F, which is clearly different from set-inclusion. The meaning of this relation is the following: $S_1 \prec S_2$ if and only if S_2 contains more unrestricted arguments

[2] $R' = \{(a, b) | aRb \text{ and } a \in X, b \in X\}$.

than S_1, or S_1 and S_2 have the same unrestricted arguments but S_2 contains less restricted arguments than S_1.

The idea of the new semantics is to have as few restricted arguments as possible in a restrictedly admissible set. In other words, the set of restricted defenders must be minimal. That is why we call the new semantics the *minimal defence semantics*, or, for short, the *min-def* semantics. The acceptable sets under this semantics are called *min-def extensions*.

Definition 10 *S is a* min-def extension *of $\mathcal{AF}_{F_{ru}}$ if and only if S is $\prec$-maximal among the restrictedly admissible sets of $\mathcal{AF}_{F_{ru}}$.*

Note that the restricted arguments in an acceptable set under the new semantics are such that, not only they defend an unrestricted argument x, but they are essential defenders since there is no unrestricted argument which defends x. Remark also that not every preferred extension is a min-def extension.

Example 6 *$\mathcal{AF}_3$ has only one min-def extension: $S = \{u_2, u_3, u_4, u_5, r_2\}$ (which is different of its preferred extension). S does not contain the restricted argument r_1 whereas r_1 defends u_2 against o_2. The reason is that the unrestricted argument u_3 defends u_2, therefore r_1 is not useful.*

We have proved a number of properties. The first one establishes the link between the admissibility and the min-def semantics (note that the admissible sets defined in $\mathcal{AF}$ are the sames in $\mathcal{AF}_{F_{ru}}$):

Property 3 *Let $S \subseteq F$. S is a min-def extension of $\mathcal{AF}_{F_{ru}}$ if and only if S is $\prec$-maximal among the admissible sets of $\mathcal{AF}$.*

Proof (Sketch) *It is a consequence of the fact that every admissible set which is $\prec$-maximal is restrictedly admissible.* $\diamondsuit$

The following property shows that every finite argumentation framework has at least one min-def extension:

Property 4 *Suppose that A is finite. Then for every admissible set G included in F, there is a min-def extension S of $\mathcal{AF}_{F_{ru}}$ such that $G \prec S$.*

Proof (Sketch) *Given Adm_F the set of the admissible sets of $\mathcal{AF}$ included in F and the relation $\prec$, we prove that every chain in Adm_F has an upper bound in Adm_F.* $\diamondsuit$

The next two properties explain the link between preferred extensions on a given set and min-def extensions. This link is very interesting in a computational perspective (see section 4).

Property 5 *If S is a min-def extension of $\mathcal{AF}_{F_{ru}}$, then there is a preferred extension E on F such that $E_u = S_u$ and $E_r \supseteq S_r$, so that $E \prec S$.*

Proof (Sketch) *The proof relies upon property 3 and the following result:*

If G is a $\prec$-maximal admissible set included in F and if H is an admissible set which contains G, then $H_u = G_u$.

Since every admissible set included in F is included in a preferred extension on F (see property 2), there exists a preferred extension E on F such that $G \subseteq E$ and then $G_u = E_u$. $\diamondsuit$

Property 6 *Let E be a preferred extension on F. There is a min-def extension S of $\mathcal{AF}_{F_{ru}}$ such that $E_u \subseteq S_u$.*

Proof (Sketch) *We show that if G is an admissible set included in F, then there exists an admissible set H included in F maximal for the relation $\prec$ such that $G_u \subseteq H_u$. The property is a consequence of this result and of property 3.* $\diamondsuit$

It can be noticed that the preferred extension on F of example 5 is such that its unrestricted part $\{u_2, u_3, u_4, u_5\}$ is equal to the unrestricted part of the min-def extension of $\mathcal{AF}_3$, and its restricted part $\{r_1, r_2, r_3\}$ contains the restricted part $\{r_2\}$ of the min-def extension of $\mathcal{AF}_3$.

3 ILLUSTRATIONS

Let us give three illustrations of the formal framework presented in section 2. The first two are examples in a dialogical context, and the last one shows that the framework can be applied to other contexts.

3.1 A DISCUSSION BETWEEN FRIENDS

Denis and Theo have a discussion. Denis has a point of view and gives arguments to support it. Denis has a friend, Olivia, who agrees with him. Theo does not agree with Denis and Olivia's point of view and gives arguments attacking their arguments. In this discussion, Denis can be viewed as the proponent and Theo as the opponent. Denis has several solutions to defend his arguments against Theo's attacks:

1. either Denis uses his own arguments only;

2. or Denis also accepts Olivia's arguments and considers that Olivia's arguments have the same importance as his: he uses indiscriminately his arguments or hers to defend his arguments;

3. or Denis accepts Olivia's arguments but he wants to privilege his own arguments: he takes into account Olivia's arguments only when they defend some of his arguments which he cannot defend with his own arguments.

The different ways Denis can accept Olivia's arguments are justified by the fact that Olivia's argument could contain, for example, very personal information on Denis. Therefore, one can imagine that if Denis does not really want these informations to be revealed, he accepts Olivia's arguments only if absolutely necessary (case 3) or he does not accept them at all (case 1). On the contrary, if Denis agrees to reveal these informations, he makes no distinction between Olivia's arguments and his (case 2).

The arguments and the attacks of the discussion constitute an argumentation framework where Denis' arguments are the unrestricted arguments, Olivia's arguments are the restricted arguments and Theo's arguments are the other arguments.

We assume that the acceptable sets of arguments supporting Denis' point of view are conflict-free sets which defend all their arguments against Theo's attacks and contain a maximum of Denis' arguments. Thus, in case 1, these sets are preferred extensions on Denis' arguments; in case 2, they are restrictedly admissible sets; in case 3, they are min-def extensions.

In a multi-agent framework, it is usually the case that the agents need to communicate to fulfil tasks or to share resources, so that they have to engage in dialogues. The works of [Parsons et al., 1998], [Reed, 1998] and [Amgoud et al., 2000] outline that argumentation can be used as a basis for dialogues. [Amgoud et al., 2000] studies different types of dialogue in argumentation theory, and mentions the strategies that an agent uses to choose an argument during a dialogue. This point is more precisely investigated in [Amgoud and Maudet, 2002]. The example above suggests that our work could be used as a support to implement the strategies. The unrestricted arguments of an agent would be the ones which satisfy the strategy, the restricted arguments would be the other arguments of the agent.

3.2 MAFIA ON TRIAL

Consider a criminal court case with the mafia on trial, and assume that some piece of witness testimony is very convincing evidence against the suspects, but that the prosecution still hesitates to use it, since doing so would inevitably disclose the witnesses' identity, and the mafia gang is known to be very violent against people who testify against them. In such a case, the prosecutor will try to avoid using this piece of evidence as long as possible, and try to win the case in other ways. Only if these other ways fail, the prosecutor may decide to use the witness testimony.

The parallel between this criminal court case and a `type-2`-argumentation framework is easy to do: the pieces of evidence which can be disclosed are the unrestricted arguments, and the witness testimony is a restricted argument. [3]

The set of arguments advanced by the prosecution is a conflict-free set, which defends all its elements and which contains the restricted argument only if it defends a piece of evidence which could not be defended otherwise. One can imagine that the set of pieces of evidence which can be disclosed is maximal, since the more pieces of evidence are justified, the better the suspects' guilt is ensured. Therefore, the set of arguments advanced by the prosecution is a min-def extension.

3.3 CONFIGURATION OF A WARPLANE

The following illustration shows that our work can be applied to non dialogical contexts, to a configuration problem for instance.

Assume that the army wants to configure its warplanes in order for them to defend against any attack of the enemy during the next mission. Among the functions which can be put on a plane, some are desirable and even essential, like flight, propulsion and communication functions; others are optional, like the jammers of radar. A maximum of desirable functions must be put on a plane, whereas a minimum of optional functions must be installed because of cost, speed and weight constraints. The enemy has materials which threaten some functions; for instance, their jammer of flight threatens the flight function.

A parallel can easily be established between this problem of configuration and a `type-2`-partitioned argumentation framework: we set A to be the set of functions and threats; F is then the set of functions, and

[3] It is this example, suggested by the reviewers, which prompted to us the expression "restricted argument", in the sense of "confidential argument".

F_u is the set of desirable functions whereas F_r is the set of optional functions; the attack relation R corresponds to the conflicts between functions and threats.

A good configuration of a plane is defined as a set of functions C such that:

1. C is conflict-free;
2. if a function in C is threatened, then C contains a function which attacks the threat;
3. C contains a maximum of desirable functions and a minimum of optional functions.

The parallel established with a type-2-partitioned argumentation framework shows that a good configuration is a set included in F which is conflict-free (1) and which defends all its arguments (2); therefore, it is an admissible set included in F. This set contains a maximum of unrestricted arguments (the desirable functions), and a minimum of restricted arguments (the optional functions) (3); in other words, this admissible set is $\prec$-maximal. Consequently, according to property 3, a good configuration is a min-def extension.

4 CONCLUSION AND FUTURE WORKS

In this paper, using a partition of the set of arguments, we propose some refinements of the argumentation framework defined by [Dung, 1995]. The aim of this partition is to privilege one subset of arguments and, in addition, to make a distinction between restricted and unrestricted arguments. In this context, we have defined different semantics and the associated acceptable sets:

- the *restrictedly admissible sets* are admissible sets of arguments in which the presence of each restricted argument is justified (a restricted argument belongs to a restrictedly admissible set only if it defends an unrestricted argument);
- the *min-def extensions* are restrictedly admissible sets which respect the following condition: they maximize the unrestricted arguments and minimize the restricted arguments.

Different properties are presented which show the relations between the different semantics.

Some of these results (reported in section 2) suggest that the min-def extensions can be computed from the set of the preferred extensions on F.

Actually, from property 5, it is clear that each min-def extension can be obtained from one of the preferred extensions on F, by minimizing its intersection with F_r (its restricted part). It follows from property 6 that it is sufficient to consider the preferred extensions on F such that their intersection with F_u is maximal (for set-inclusion).

[Doutre and Mengin, 2001] proposes an algorithm based on the technique of set-enumeration for computing all the preferred extensions of a given argumentation framework. The basic idea is to reduce the number of generated sets of arguments using properties of the preferred extensions. That algorithm can be adapted using the following additional properties: we are interested in subsets of F, which are preferred extensions on F and whose unrestricted part (intersection with F_u) is maximal for set-inclusion.

So, we propose a two-steps method for computing min-def extensions:

1. Compute X_1, X_2, ..., X_n the preferred extensions on F with a maximal unrestricted part.
2. From each X_i, compute the admissible sets by removing as many elements of X_{ir} (*i.e.* restricted elements) as possible.

The above method is correct and complete (see [Cayrol et al., 2002]).

In the particular case where there is no cycle in F, the first step in the above algorithm produces only one preferred extension on F.

We consider an extension of this model to the case where the set of restricted arguments would be stratified. This is motivated by several examples. For instance, among the optional functions which can be put on a plane, some can be more costly than others; one wants to choose the least expensive in priority. Moreover, we have seen that the restricted arguments of an agent can be those which contain personal information. These informations can be more or less personal, and then one prefers to disclose the arguments which contain the least personal information in priority. Consequently, we want to extend a type-2-partitioned argumentation framework with a priority relation over the restricted arguments. We will integrate this priority in the definition of the $\prec$-relation and of the min-def semantics in order to choose the restricted arguments which have the greatest priority (*i.e.* those which contain the least expensive functions or the least personal information) for the defence of the unrestricted arguments.

Acknowledgements

We would like to thank Hélène Fargier for fruitful preliminary discussions on this topic. We would also like to thank the referees for helpful comments; in particular, they suggested the "mafia example".

References

[Amgoud and Cayrol, 1998] Amgoud, L. and Cayrol, C. (1998). On the acceptability of arguments in preference-based argumentation. In *14th Conference on Uncertainty in Artificial Intelligence (UAI'98), Madison, Wisconsin*, pages 1–7, San Francisco, California. Morgan Kaufmann.

[Amgoud and Maudet, 2002] Amgoud, L. and Maudet, N. (2002). Strategical considerations for argumentative agents (preliminary report). In *NMR'2002 special session on Argument, Dialogue and Decision.*

[Amgoud et al., 2000] Amgoud, L., Maudet, N., and Parsons, S. (2000). Modelling dialogues using argumentation. In *Fourth International Conference on MultiAgent Systems (ICMAS'2000), Boston, MA, USA*, pages 31–38.

[Cayrol et al., 2002] Cayrol, C., Doutre, S., Lagasquie-Schiex, M., and Mengin, J. (2002). Défense minimale : un raffinement de la sémantique préférée pour les systèmes d'argumentation. Rapport de recherche 2002-03-R, Institut de Recherche en Informatique de Toulouse (I.R.I.T.), France.

[Doutre and Mengin, 2001] Doutre, S. and Mengin, J. (2001). Preferred Extensions of Argumentation Frameworks: Computation and Query Answering. In R. Goré, A. L. and Nipkow, T., editors, *IJCAR 2001*, volume 2083 of *LNAI*, pages 272–288. Springer-Verlag.

[Dung, 1995] Dung, P. (1995). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77:321–357.

[Parsons et al., 1998] Parsons, S., Sierra, C., and Jennings, N. R. (1998). Agents that reason and negotiate by arguing. *Journal of Logic and Computation*, 8(3):261–292.

[Prakken and Sartor, 1997] Prakken, H. and Sartor, G. (1997). Argument-based logic programming with defeasible priorities. *Journal of Applied Non-Classical Logics*, 7:25–75.

[Reed, 1998] Reed, C. (1998). Dialogue frames in agent communication. In *Proceedings of the 3rd International Conference on Multi Agent Systems*, pages 246–253.

On the existence and multiplicity of extensions in dialectical argumentation

Bart Verheij
Department of Metajuridica
Universiteit Maastricht[1]

Abstract

In the present paper, the existence and multiplicity problems of extensions are addressed. The focus is on extension of the stable type. The main result of the paper is an elegant characterization of the existence and multiplicity of extensions in terms of the notion of dialectical justification, a close cousin of the notion of admissibility. The characterization is given in the context of the particular logic for dialectical argumentation DEFLOG. The results are of direct relevance for several well-established models of defeasible reasoning (like default logic, logic programming and argumentation frameworks), since elsewhere dialectical argumentation has been shown to have close formal connections with these models.

1 INTRODUCTION

When a theory is interpreted in the context of logics for defeasible reasoning, e.g., in terms of the theory's extensions, it occurs for many such logics that there exist theories that cannot be interpreted at all or that have more than one interpretation.

For instance, in Reiter's (1980) well-known logic for default reasoning, a theory's extensions can be thought of as its interpretations. The simplest theory without extension consists of only the default *true* : ¬p / p. A basic example of a theory with more than one extension consists of the facts p and q and the defaults p: r / r and q : ¬r / ¬r.

For a good understanding of defeasible reasoning, it is natural to investigate under what circumstances theories are (so to speak) defeasibly interpretable, in the sense of having an extension, and under what circumstances their defeasible interpretation is ambiguous, in the sense of having more than one extension.

Several properties of theories have been discussed that guarantee the existence of extensions. For instance, in the context of his default logic, Reiter (1980) defined normal theories, that could be shown to have at least one extension. Etherington (1987) defined ordered theories and showed that ordered, semi-normal theories always have an extension (cf. also Papadimitriou and Sideri 1994). More results and references are for instance given by Gabbay *et al.* (1994) and Brewka *et al.*(1997).

In the present paper, the notorious extension existence and extension multiplicity problems are addressed in the context of a specific form of defeasible reasoning, viz. dialectical argumentation. In dialectical argumentation, statements are not only adduced as reasons for other statements, but also as reasons against. As a result of this possibility of simultaneously supporting and attacking statements, there can be statements that are justified (e.g. since there is a justifying reason for the statement) and statements that are defeated (e.g. since there is a defeating reason against the statement). Chesñevar *et al.* (2000) and Prakken and Vreeswijk (to appear) give overviews of models of dialectical argumentation. Formal models of dialectical argumentation are closely related to other nonmonotonic logics, as has especially been shown by Dung (1995) and Bondarenko *et al.* (1997). As a result, it comes as no surprise that the existence and multiplicity problems also arise in the context of dialectical argumentation. Dung (1995) and Bondarenko *et al.* (1997) give several relevant results in this respect (for different possible kinds of extensions).

In the present paper, the focus is on extensions of the stable type (cf. Gelfond and Lifschitz's (1988) stable models of logic programming, Dung's (1995) stable extensions of argumentation frameworks, and Bondarenko *et al.*'s (1997) stable extensions of assumption-based frameworks). The notion of dialectical justification is introduced and is shown to play a central role in the solution of the extension existence and multiplicity problems. The main results of the paper are characterizations of the existence of an extension and of the number of extensions in terms of dialectical justification. By a meta-analysis , it is shown that the notion of dialectical justification is vital in the

[1] bart.verheij@metajur.unimaas.nl,
http://www.metajur.unimaas.nl/~bart/

characterization of the existence and the multiplicity of extensions.

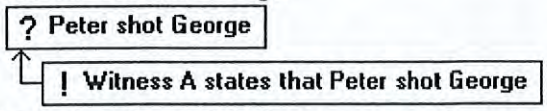
The results are proven for a particular formalization of dialectical argumentation, viz. DEFLOG (Verheij 2000a, b). The system is related to my work on automated argument assistance (e.g., Verheij 1999). DEFLOG is formally closely related to Dung's (1995) and Bondarenko *et al.*'s (1997) formalizations. For Dung's (1995) argumentation frameworks, the connection is formally established in section 4 below. With respect to the logical language, DEFLOG differs from Dung's (1995) and Bondarenko *et al.*'s (1997) formalizations in that it allows the explicit expression in the logical object language of support and attack and of the defeat of a statement. As a result, support and attack become subject to dispute. The defeat of a statement is as a variant of negation, called *dialectical negation*. In contrast, Dung (1995) (who does not discuss support) uses a fixed, undisputable attack relation. Bondarenko *et al.* (1997) use a fixed, undisputable set of rules of inference to express support. They use a domain-dependent mapping of sentences to their contraries that already hints at the explicit expression of defeat in the logical object language.

By the already mentioned close formal relations between well-established models of defeasible reasoning (like default logic, logic programming and argumentation frameworks) and models of dialectical argumentation, the results proven for DEFLOG are of direct relevance for these models.

2 DIALECTICAL ARGUMENTATION

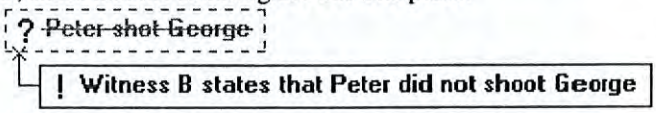
The context of the present work is dialectical argumentation. Here follows a brief introduction along the lines of my recent work on that topic (Verheij 1999, 2000a, b).

In dialectical argumentation, statements can not only support other statements, but also attack them. For instance, as a reason to support that Peter shot George, the statement can be made that some witness, say A, states that Peter shot George:



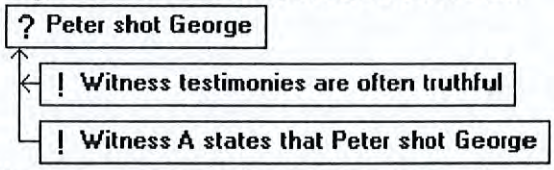
The exclamation mark indicates an assumed statement, the question mark a statement that is at issue. (The graphical presentation of arguments is based on previous work by the present author.) Here the issue that Peter shot George is settled (the statement is justified, as is indicated by the dark, bold font) since there is a justifying reason for it, namely A's testimony.

As a reason against the issue that Peter shot George, the statement can be made that some other witness, say B, states that the shooting did not take place:



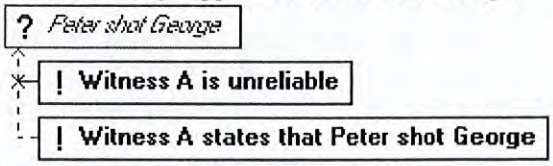
Assuming only B's testimony (not A's), the issue that Peter shot George is again settled, but this time the statement is defeated, as is indicated by the struck-through font. (Note that here B's testimony is used to argue *against* the statement that Peter shot George, and not to argue *for* the statement that Peter did *not* shoot George. As will be seen in the formal discussion below, arguing against a statement is treated as arguing for the statement's so-called dialectical negation, which is not necessarily equivalent to arguing for its ordinary negation.)

That some statement supports or attacks another statement can itself be at issue. For instance, it can be argued that A's testimony supports that Peter shot George since witness testimonies are often truthful:



Likewise, a reason can be given to support that some statement *attacks* another statement.

A's unreliability can be adduced in order to attack that A's testimony supports that Peter shot George:



Here the issue that Peter shot George is unsettled, as is indicated by the light italic font, since it is not justified (e.g., by a justifying reason for it) nor defeated (e.g., by a defeating reason against it). Similarly, a reason can be given to attack that some statement attacks another statement.

3 DEFLOG - A LOGIC OF DIALECTICAL INTERPRETATION

3.1 THE DIALECTICAL INTERPRETATION OF THEORIES

The ideas on dialectical argumentation discussed in section 2 can be made formally precise in terms of the logical system DEFLOG (Verheij 2000a). Its starting point is a simple logical language with two connectives $\times$ and $\rightarrow$. The first is a unary connective that is used to

express the defeat of a statement, the latter is a binary connective that is used to express that one statement supports another. When φ and ψ are sentences, then $\times\varphi$ (φ's so-called *dialectical negation*) expresses that the statement that φ is defeated, and $(\varphi \rightarrow \psi)$ that the statement that φ supports the statement that ψ. Attack, denoted as $\rightthreetimes$, is defined in terms of these two connectives: $\varphi \rightthreetimes \psi$ is defined as $\varphi \rightarrow \times\psi$, and expresses that the statement that φ attacks the statement that ψ, or equivalently that the statement that φ supports that the statement that ψ is defeated. When p, q, r and s are elementary sentences, then $p \rightarrow (q \rightarrow r)$, $p \rightarrow \times(q \rightarrow \times r)$ and $(p \rightarrow q) \rightarrow (p \rightarrow \times(r \rightarrow s))$ are some examples of sentences. (For convenience, outer brackets are omitted.)

The central definition of DEFLOG is its notion of the *dialectical interpretation* of a theory. Formally, DEFLOG's dialectical interpretations of theories are a variant of Reiter's (1980) extensions of default theories, Gelfond and Lifschitz's (1988) stable models of logic programming, Dung's (1995) stable extensions of argumentation frameworks, and Bondarenko *et al.*'s (1997) stable extensions of assumption-based frameworks.[1]

A theory is any set of sentences, and when it is dialectically interpreted, all sentences in the theory are evaluated, either as justified or as defeated. (This is in contrast with the interpretation of theories in standard logic, where all sentences in an interpreted theory are assigned the same positive value, namely true, e.g., by giving a model of the theory.)

An assignment of the values justified or defeated to the sentences in a theory gives rise to a dialectical interpretation of the theory, when two properties obtain. First, the justified part of the theory must be conflict-free. Second, the justified part of the theory must attack all sentences in the defeated part. Formally the definitions are as follows.

(i) Let T be a set of sentences and φ a sentence. Then T *supports* φ when φ is in T or follows from T by the repeated application of $\rightarrow$-Modus ponens (i.e., from $\varphi \rightarrow \psi$ and φ, conclude ψ). T *attacks* φ when T supports $\times\varphi$.

(ii) Let T be a set of sentences. Then T is *conflict-free* when there is no sentence φ that is both supported and attacked by T.

(iii) Let Δ be a set of sentences, and let J and D be subsets of Δ that have no elements in common and that have Δ as their union. Then (J, D) *dialectically interprets* the theory Δ when J is conflict-free and attacks all sentences in D. The sentences in J are the *justified statements* of the theory Δ, the sentences in D the *defeated statements*.

(iv) Let Δ be a set of sentences and let (J, D) dialectically interpret the theory Δ. Then (Supp(J), Att(J)) is a *dialectical interpretation* or *extension* of the theory Δ. Here Supp(J) denotes the set of sentences supported by J, and Att(J) the set of sentences attacked by J. The sentences in Supp(J) are the *justified statements* of the dialectical interpretation, the sentences in Att(J) the *defeated statements*.

Note that when (J, D) dialectically interprets Δ and (Supp(J), Att(J)) is the corresponding dialectical interpretation, J is equal to Supp(J) $\cap$ Δ, and D to Att(J) $\cap$ Δ. It is convenient to say that a dialectical interpretation (Supp(J), Att(J)) of a theory Δ *is specified by* J.

The examples discussed in section 2 can be used to illustrate these definitions. Let s express Peter's shooting of George, a A's testimony, b B's testimony, t the truthfulness of testimonies, and u A's unreliability. Then the first example corresponds to the two-sentence theory $\{a, a \rightarrow s\}$. The arrow in the figure corresponds to the sentence $a \rightarrow s$. The theory has a unique extension in which all statements of the theory are justified. In the extension, one other statement is justified, viz. s. The second example corresponds to the theory $\{b, b \rightarrow \times s\}$. The arrow ending in a cross in the figure corresponds to the sentence $b \rightarrow \times s$. The theory has a unique extension in which again all sentences of the theory are justified. In the extension, there are two other interpreted statements, viz. $\times s$, which is justified, and s, which is defeated. (The reader may wish to check that the theory $\{b, b \rightarrow \times s, s\}$, which is not conflict-free, has the same unique extension, but that one of the statements in the theory is defeated.) The third example corresponds to

[1] In section 4, a formal connection with Dung's (1995) work is discussed. More relations between the formalisms mentioned are e.g. discussed by Dung (1995) and in the extended manuscript on which the present is based (Verheij 2000a). To guide intuition, the following may be useful. A default p : q / r (as in Reiter's 1980) would in DEFLOG be translated to two conditionals, viz. $p \rightarrow r$ and $\neg q \rightarrow \times(p \rightarrow r)$. The second says that the former is defeated in case of $\neg q$. This corresponds to the intuition underlying the default that r follows from p as long as q can consistently be assumed. (Note however that the properties of ordinary negation $\neg$ are not part of DEFLOG proper.) A rule in logic programming $p \leftarrow q, \sim r$ corresponds in DEFLOG to two conditionals, viz. $q \rightarrow p$ and $r \rightarrow \times(q \rightarrow p)$. The second says that $q \rightarrow p$ is defeated in case of r. This corresponds to the intuition underlying the program rule that p follows from q when r is not provable, but not when r is provable.

the theory {a, t, t ⇝ (a ⇝ s)}. In its unique extension, all statements of the theory are justified, and in addition a ⇝ s and s. The fourth example corresponds to the theory {a, u, u ⇝ ×(a ⇝ s)}. In its unique extension, a ⇝ s is defeated and s is not interpreted (i.e., neither justified nor defeated). (The theory {a, u, u ⇝ ×(a ⇝ s), a ⇝ s} is not conflict-free, but has the same unique extension.)

DEFLOG's connectives ⇝ and × are obviously reminiscent of propositional logic's connectives → and ¬. Also some of DEFLOG's definitions remind of propositional logic. These likenesses have been incorporated on purpose. In fact, DEFLOG has been carefully designed to be as close as possible to propositional logic (as the paradigmatic example of deductive logic), while retaining the essence of defeasible logic.[2]

DEFLOG's connectives are not equivalent to propositional logic's connectives. For instance, the set {p, ×p} is not inconsistent, in the dialectical sense: the theory {p, ×p} has a unique dialectical interpretation in which p is defeated and ×p justified. Of course {p, ¬p} is classically inconsistent. The theory {p, ×p} shows the essence of dialectical negation: the dialectical negation of a sentence in a sense 'prevails' over the sentence.[3] By this prevalence of dialectical negation, assumptions are only *prima facie* justified: a *prima facie* assumption is not actually justified when the dialectical negation of the assumption is (actually) justified.

The theory {p, ×p} also shows that dialectical interpretation is not simply maximal consistency: whereas the maximal consistent subset {×p} corresponds to a (the) dialectical interpretation, but {p} does not.

Verheij (2000a) gives much more information on DEFLOG, for instance, on different ways to adapt DEFLOG to incorporate the classical logical connectives. (DEFLOG's connectives ⇝ and × are not meant to *replace* the classical connectives; they express different concepts.)

It is not hard to see that DEFLOG is non-monotonic, for instance in the following sense: when a sentence is justified in some dialectical interpretation of a theory, it need not be in a dialectical interpretation of a larger theory. The simplest example is provided by the theories {p} and {p, ×p}. Both have only one dialectical interpretation. In the dialectical interpretation of {p}, p is justified, but in that of {p, ×p}, p is defeated (and ×p justified).

Notwithstanding the simple structure of DEFLOG's logical language (with only two connectives, viz. ⇝ and ×), many central notions of dialectical argumentation can be analyzed in terms of it. For instance, it is possible to define an inconclusive conditional (i.e., a conditional of which the consequent does not always follow when its antecedent obtains) in terms of DEFLOG's defeasible conditional (that is defeasible in the same way as any other statement). DEFLOG's expressiveness also allows an integrated analysis of Toulmin's (1958) warrants and backings and Pollock's (1987) undercutting and rebutting defeaters. A warrant and an undercutter can be seen as the support and attack, respectively, of the relation between a reason and its conclusion. Undercutting and rebutting defeaters become different instances of the general phenomenon of defeat. Cf. Verheij (2000a, 2001).

3.2 THEORIES WITHOUT EXTENSIONS AND THEORIES WITH SEVERAL EXTENSIONS

The examples of theories discussed above all had a unique extension. Several were examples of the following general property: a conflict-free theory always has a unique extension, namely the extension specified by the theory itself. The simplest theory that is not conflict-free with a unique extension is {p, ×p}. In its extension, p is defeated and ×p justified. Other important examples of theories that are not conflict-free, but do have a unique extension are {p, q, q ⇝ ×p} and {p, q, r, q ⇝ ×p, r ⇝ ×q}. In the former theory, the statement that p, is attacked by the statement that q. In its unique extension, q and ×p are justified and p is defeated. In the latter theory, a superset of the former, in addition to q's attack of p, r attacks q. In its unique extension, p, ×q and r are justified, and q is defeated. The theories together provide an example of *reinstatement*: a statement is first defeated, since it is attacked by a counterargument, but becomes justified by the addition of a counterattack, i.e., an attack against the counterargument. Here p is reinstated: it is first successfully attacked by q, but the attack is then countered by r attacking q.

There are however also theories with no or with several extensions:

(i) The three theories {p, p ⇝ ×p}, {p, p ⇝ q, ×q} and {p_i | i is a natural number} ∪ {p_j ⇝ ×p_i | i and j are natural numbers, such that i < j} lack extensions.

---

[2] One result of this exercise is that I have come to believe that the concept of dialectical negation (along with the corresponding concept of dialectical interpretation) is the essential difference between a deductive and a defeasible logic.

[3] Note that the prevalence relation between a sentence p and its weak negation ~p in logic programming is exactly opposite to that between a sentence p and its dialectical negation ×p: in logic programming ~p can be assumed as long as p is not provable, while in dialectical argumentation p can be assumed as long as ×p is not justified.

For the latter theory, this can be seen as follows. Assume that there is an extension E in which for some natural number n p_n is justified. Then all p_m with $m > n$ must be defeated in E, for if such a p_m were justified, p_n could not be justified. But that is impossible, for the defeat of a p_m with $m > n$ can only be the result of an attack by a justified $p_{m'}$ with $m' > m$. As a result, no p_i can be justified in E. But then all p_i must be defeated in E, which is impossible since the defeat of a p_i can only be the result of an attack by a justified p_j with $j > i$. (Note that any *finite* subset of the latter theory has an extension, while the whole theory does not. This shows a 'non-compactness' property[4] of extensions.)

(ii) The three theories $\{p, q, p \rightsquigarrow \times q, q \rightsquigarrow \times p\}$, $\{p_i, p_{i+1} \rightsquigarrow \times p_i \mid i \text{ is a natural number}\}$ and $\{\times^i p \mid i \text{ is a natural number}\}$ have two extensions. Here $\times^i p$ denotes, for any natural number i, the sentence composed of a length i sequence of the connective $\times$, followed by the constant p. (Note that each finite subset of the latter theory has a unique extension, showing another non-compactness property.)

Traditionally, the main example of multiple extensions is the Nixon diamond. In Reiter's (1980) default logic, it looks thus:

$$q, r$$
$$q : p / p$$
$$r : \neg p / \neg p$$

These express that Nixon is a quaker and a republican, and that quakers are pacifists, while republicans are non-pacifists. Reiter's definitions give rise to two extensions. In one, p follows by the application of the first default, in the other, $\neg p$ follows by the application of the first default.

It may be thought that its DEFLOG representation has the form $\{q, r, q \rightsquigarrow p, r \rightsquigarrow \times p\}$. However, this set does not have a dialectical interpretation. A good representation of the Nixon diamond consists of the following assumptions:

q, r
$q \rightsquigarrow p$
$r \rightsquigarrow \neg p$
$(q \,\&\, (q \rightsquigarrow p)) \rightsquigarrow \times(r \rightsquigarrow \neg p)$
$(r \,\&\, (r \rightsquigarrow \neg p)) \rightsquigarrow \times(q \rightsquigarrow p)$

Here $(\varphi \,\&\, \psi) \rightsquigarrow \chi$ abbreviates $\varphi \rightsquigarrow (\psi \rightsquigarrow \chi)$. The connective $\neg$ is intended to express ordinary negation. The latter two conditionals express that when q justifies p (expressed by the conjunction of q and $q \rightsquigarrow p$), it is defeated that r implies $\neg p$, and that when r is actually justified, it is defeated that q implies p. In each of the two dialectical interpretations of these assumptions, one of $q \rightsquigarrow p$ and $r \rightsquigarrow \neg p$ is defeated, the other justified. In the DEFLOG formalization, the conditionals $q \rightsquigarrow p$ and $r \rightsquigarrow \neg p$ stand for the 'application' of the corresponding Reiter defaults, while the conditionals $(q \,\&\, q \rightsquigarrow p) \rightsquigarrow \times(r \rightsquigarrow \neg p)$ and $(r \,\&\, r \rightsquigarrow \neg p) \rightsquigarrow \times(q \rightsquigarrow p)$ express when that application is blocked. The difference between the Reiter and the DEFLOG formalization has to do with the fact that Reiter's defaults are inconclusive (their consequent does not always follow when their antecedent obtains), while DEFLOG uses conditionals that are prima facie justified, just like other assumptions. Cf. also Verheij (2000a). Note that the opposition between p and $\neg p$ is not represented in terms of dialectical negation (i.e., as p and $\times p$): ordinary negation and dialectical negation are different notions.

3.3 DIALECTICALLY JUSTIFYING ARGUMENTS

Before we proceed to the notion of dialectical justification, some terminology needs to be introduced.

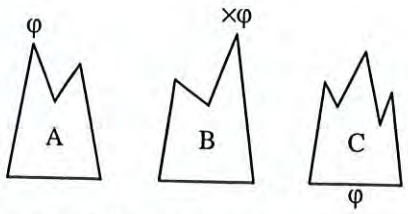
(i) A set of sentences is an *argument* when it is conflict-free. If Δ is a set of sentences, a Δ-*argument* is an argument that is a subset of Δ.

(ii) Let φ be a sentence. An argument C is an *argument for* φ if C supports φ. An argument C is an *argument against* φ if C attacks φ. The sentences in an argument C are also called its *premises*, the sentences φ such that C supports φ, its *conclusions*.

(iii) An argument C *attacks* an argument C' if C attacks a sentence in C'.

(iv) Arguments C and C' are *compatible* when $C \cup C'$ is an argument, and otherwise *incompatible*. The arguments in a collection $\{C_i\}_{i \in I}$ are *compatible* if their union $\cup_{i \in I} C_i$ is an argument, otherwise *incompatible*.

In the following figure, three arguments are graphically suggested.



The bottoms of the alpine shapes consist of the premises of the argument; the tops are the conclusions. Argument A has conclusion φ, argument B conclusion $\times\varphi$ and argument C has premise φ. B attacks C, but not

[4] A property P of sets is called compact if a set S has property P whenever all its finite subsets have the property. Cf. the compactness of satisfiability in first-order predicate logic.

necessarily A (since φ might not be a premise of A). A and B are incompatible, and B and C are too.

When a theory has a dialectical interpretation, the set of sentences of the theory that are justified in the interpretation, clearly form an argument. It has a special property:

Proposition

Let E be an extension of a theory Δ. Then J(E) ∩ Δ is a Δ-argument that attacks any Δ-argument C that is incompatible with J(E) ∩ Δ. Here J(E) denotes the set of justified statements of the extension E.

Proof: Since E is an extension, J(E) ∩ Δ is conflict-free. Hence a Δ-argument C that is incompatible with J(E) ∩ Δ cannot be a subset of J(E) ∩ Δ since J(E) ∩ Δ is not incompatible with any of its subsets. Therefore there is a sentence φ in C that is not in J(E) ∩ Δ. Since E is an extension, it is in D(E), the set of defeated sentences of the extension E. But for any sentence φ in D(E) it holds by the definition of extensions that J(E) ∩ Δ attacks φ, and therefore attacks C. QED

Arguments with the property that J(E) ∩ Δ has in the proposition above, are said to be dialectically justifying:

(v) A Δ-argument C is *dialectically justifying* with respect to Δ if and only if C attacks every Δ-argument C' that is incompatible with C.

(vi) A sentence φ is *dialectically justifiable* with respect to a set of sentences Δ if and only if there is a Δ-argument C for φ that is dialectically justifying with respect to Δ. Such an argument C is then called a *dialectical justification of* φ, and C *dialectically justifies* φ with respect to Δ. A sentence φ is *dialectically defeasible* with respect to Δ if and only if ×φ is dialectically justifiable with respect to Δ. If C is a dialectical justification of φ, then the argument C *dialectically defeats* φ with respect to Δ.

(vii) A sentence φ is *dialectically interpretable* with respect to a set of sentences Δ if and only if it is dialectically justifiable or dialectically defeasible with respect to Δ. A sentence φ is *dialectically ambiguous* with respect to a set of sentences Δ if and only if it is both dialectically justifiable and dialectically defeasible with respect to Δ.

The argument {p, r, r → ×q} dialectically justifies p with respect to the theory {p, q, r, q → ×p, r → ×q}. The argument {p} does not dialectically justify p since the incompatible argument {q, q → ×p} is not attacked. The argument {r, r → ×q} dialectically defeats q with respect to the theory.

The sentences p and q are dialectically ambiguous with respect to the theory {p, q, p → ×q, q → ×p} since the argument {p, p → ×q} dialectically justifies p and dialectically defeats q, and likewise for q.

The sentence p is not dialectically interpretable with respect to the theory {p, p → ×p}.

Note that when an argument is dialectically justifying with respect to a theory, it dialectically justifies all the sentences it supports.

3.4 THE EXISTENCE AND MULTIPLICITY OF EXTENSIONS

When a theory has a dialectical interpretation, all sentences in the theory are dialectically interpretable. In other words, dialectical justification is a kind of 'local' dialectical interpretation. This is an immediate corollary of the proposition proven in section 3.3:

Corollary

Let E be an extension of the theory Δ. Then all sentences in the theory are dialectically justifiable or dialectically defeasible with respect to Δ.

Proof: By the proposition of section 3.3, J(E) ∩ Δ dialectically justifies or defeats all sentences in Δ. QED

Note that the corollary gives a necessary condition for the existence of an extension: when there is a sentence in a theory that is not dialectically interpretable, there cannot be an extension. The corollary can explain all examples of theories without extensions that have been encountered above: in all, there is a sentence that is not dialectically interpretable. Nevertheless the condition in the corollary is *not* sufficient for the existence of an extension, as the theory Δ = {p, q, p → ×q, q → ×p, r, r → ×r, s, s → ×s, p → ×r, q → ×s} shows. It has no extension. Nevertheless all sentences in the theory are dialectically justifiable or defeasible with respect to Δ. The Δ-argument {p, p → ×q, p → ×r} dialectically justifies p and dialectically defeats q and r, while {q, q → ×p, q → ×s} dialectically justifies q and dialectically defeats p and r.

The notion of dialectical justification plays the central role in the main theorem of the present paper, that shows exactly under which circumstances a theory has an extension. One additional definition is needed.

(viii) Let C be an argument. A sentence φ is *dialectically justifiable in the context* C with respect to a theory Δ if it is supported by a dialectically justifying argument of the theory that contains C, and *dialectically defeasible in the context* C if ×φ is supported by a dialectically justifying argument that contains C.

Using this terminology, the main theorem can be formulated:

Theorem

A theory Δ has an extension if and only if there is an argument C in the context of which all sentences in Δ are either dialectically justifiable or dialectically defeasible with respect to the theory, but not both.

(The proof follows below.) In other words, a theory has an extension if and only if there is a context in which all sentences of the theory are dialectically interpretable, while none is dialectically ambiguous. The theorem is closely related to the corollary above that says that the dialectical interpretability of all sentences of a theory is necessary for the existence of an extension. The theorem says that the dialectical interpretability of all sentences *in a context with no dialectical ambiguities* is both necessary and sufficient for the existence of an extension. After fixing all choices allowed by a dialectically ambiguous sentence in the theory, it suffices for the existence of an extension that all sentences in the theory are either dialectically justifiable or dialectically defeasible. The example that showed why the dialectical interpretability of all sentences of a theory is not sufficient for the existence of an extension, shows what can go wrong: the dialectical justification of one sentence (or its dialectical negation) need not be compatible with that of another when there is a dialectical ambiguity. In other words, the dialectical justification of sentences can depend on the particular choice allowed by a dialectical ambiguity. Dialectical justifications that require different choices cannot be 'glued' to form an extension.

Three properties of dialectical justification are essential in the proof of the theorem:

Proposition

(i) *Localization:* Let E be an extension of a theory Δ. Then there is a collection $\{C_i\}_{i \in I}$ of arguments that covers $J(E) \cap \Delta$ (i.e., $J(E) \cap \Delta$ is equal to $\cup_{i \in I} C_i$), that are dialectically justifying with respect to the theory.

(ii) *Union:* If C and C' are compatible arguments, that are dialectically justifying with respect to a theory Δ, then also $C \cup C'$ is dialectically justifying with respect to the theory. (Similarly, for collections of dialectically justifying arguments: the union of a compatible collection of dialectically justifying arguments is again dialectically justifying.)

(iii) *Separation at the base:*[5] If C and C' are incompatible arguments, that are dialectically justifying with respect to a theory Δ, then there is a sentence in Δ that is both dialectically justifiable and defeasible with respect to Δ. (Similarly, for collections of dialectically justifying arguments: given an incompatible collection of dialectically justifying arguments, there is a sentence in the theory that is both dialectically justifiable and defeasible.)

Proof: Localization follows from the proposition at the beginning of this section: it shows that $J(E) \cap \Delta$ is itself dialectically justifying with respect to Δ. The union property (for pairs of arguments) is seen as follows. Let C and C' be compatible dialectically justifying arguments, and let the argument C'' be incompatible with $C \cup C'$. Assume first that C'' is incompatible with C. Then clearly C attacks C''. Assume second that C'' is compatible with C. Then C' is incompatible with the argument $C \cup C''$, and therefore attacks it. Since C and C' are compatible, it then follows that C' attacks C''. The proof of the general case of the union property requires some extra care, but is similar. The property of separation at the base follows directly from the definition of dialectical justification: when C and C' are dialectically justifying and incompatible, they attack each other. Then there is a sentence in each (and therefore in the theory itself) that is attacked by the other. The general case of the separation property can be reduced to the case of pairs of arguments. QED

Proof of the main theorem: First let E be an extension of Δ. Then by the localization property $J(E) \cap \Delta$ can be covered by arguments that are dialectically justifying with respect to Δ. By the union property, it then follows that $J(E) \cap \Delta$ is also dialectically justifying. (In fact, the proof of the corollary at the beginning of the section directly shows that $J(E) \cap \Delta$ is dialectically justifying.) As a result, $J(E) \cap \Delta$ is a context as in the theorem since by the fact that $J(E) \cap \Delta$ is dialectically justifying and by the definition of extensions all sentences in Δ are dialectically interpretable in the context of $J(E) \cap \Delta$, and since by the fact that $J(E) \cap \Delta$ is conflict-free there is no dialectically ambiguous sentence in that context. Second let C be a context as in the theorem, and let, for all sentences φ, C_φ be a Δ-argument dialectically justifying or defeating φ in the context C. The collection of the C_φ is compatible since by the property of separation at the base there would otherwise be a sentence in the theory that is dialectically ambiguous in the context C. By the union property, the union of the C_φ is dialectically justifying. It specifies an extension of Δ. QED

The proof shows that extensions can be built by 'gluing' dialectically justifying arguments. This suggests that a (set-theoretically minimal) argument that dialectically justifies a sentence, is a kind of 'dialectical proof' of the sentence. Similarly, such a dialectical proof of the dialectical negation of a

[5] The property is called separation *at the base* since the dialectically ambiguous sentence can be found in the theory itself.

sentence is a kind of 'dialectical refutation' of the sentence.

The following theorem provides a general answer to the extension existence and multiplicity problems in the context of dialectical argumentation. It is a corollary of the main theorem above:

Theorem

Let n be a natural (or cardinal) number (possibly 0). A theory Δ has exactly n extensions if and only if n is equal to the maximal number of mutually incompatible arguments C in the context of which all sentences in Δ are either dialectically justifiable or dialectically defeasible with respect to the theory, but not both.

4 DUNG'S ARGUMENTATION FRAMEWORKS AND ADMISSIBILITY

Dung's (1995) argumentation frameworks are a fruitful abstraction of ideas from nonmonotonic reasoning and logic programming. Here it is shown how Dung's argumentation frameworks can be mimicked in DEFLOG. Since Dung has shown that his argumentation frameworks have close formal connections with well-established models of defeasible reasoning, such as Reiter's (1980) default logic and logic programming, the results on DefLog presented here become of direct relevance for these models. Moreover it is shown why Dung's notion of admissibility cannot in general replace that of dialectical justification in the characterizations of the existence and of the number of extensions of a theory proven above.

Formally, an argumentation framework consists of a set, its elements called *arguments*, and a binary relation on that set, the *attack* relation. When (A, B) is in the attack relation, the argument A is said to attack B.

In Dung's work, the notion of admissibility is central. It is closely related to DEFLOG's dialectical justification. Using DEFLOG's terminology, an argument C is *admissible* with respect to a theory Δ if C attacks any Δ-argument attacking it. This definition of admissibility depends of course on DEFLOG's particular notions of argument and attack. There is however a straightforward way of mimicking Dung's argumentation frameworks in DEFLOG for which this definition of admissibility is indeed an extrapolation of Dung's admissibility, as follows.

Let each argument of an argumentation framework be an elementary sentence in DEFLOG's language. Then an argumentation framework can be translated to a theory in DEFLOG by taking the union of the set of arguments in the framework and the set of sentences of the form $A \rightsquigarrow \times B$, for any element (A, B) of the attack relation of the framework. In addition, it is easy to restrict DEFLOG's language in such a way that any theory in this restricted language corresponds to an argumentation framework in Dung's sense: simply allow only elementary sentences and sentences of the form $\varphi \rightsquigarrow \times\psi$, where φ and ψ are elementary. Let's call sentences in this restricted sense *Dung sentences* and theories consisting of Dung sentences *Dung theories*.

It is now straightforward to check that several of Dung's notions coincide with DEFLOG's under this translation. Some care is needed however since certain terms have different meanings in Dung's work and in DEFLOG. For instance, the use of the term 'argument' is different. Conflict-free sets of arguments (in Dung's sense) correspond however with conflict-free sets of Dung sentences (in DEFLOG's sense), Dung's admissible sets of arguments correspond to the admissible arguments of Dung theories (in DEFLOG's sense), and Dung's stable extensions of argumentation frameworks correspond with DEFLOG's extensions of Dung theories. In the extended manuscript on which the present paper is based (Verheij 2000a), these results are formally established. The proofs are straightforward.

For theories using DEFLOG's full language, dialectical justification and admissibility are easily seen to be different notions, but on the restricted language of Dung's frameworks, the notions coincide:

Proposition

Let Δ be a Dung theory. Then a Δ-argument is dialectically justifying with respect to Δ if and only if it is admissible with respect to Δ.

Proof: Dialectically justifying arguments are always admissible. (This does not depend on Δ being a Dung theory.) Let now C be an admissible argument, and let C' be an argument incompatible with C. Since C and C' consist of Dung sentences, the incompatibility of C and C' implies that C attacks C' or that C' attacks C. In case C' attacks C, also C attacks C' since C is admissible. This shows that C is dialectically justifying. QED

Note that by this result the theorems on the extension existence and multiplicity problems can *for Dung theories* be rephrased in terms of admissibility instead of dialectical justification. This is not the case for theories in general. Then the notion of dialectical justification is essential. The key point is that admissibility does not have all of the properties used in the proof of the main theorem of this paper. These properties are localization, union and separation at the base.

The analogues of these properties for admissibility can be found by replacing 'dialectically justifying' by 'admissible' in the formulation of the properties. For instance, the union property (for pairs of arguments) for admissibility reads thus: if C and C' are compatible arguments, that are admissible with respect to a theory Δ, then also $C \cup C'$ is admissible with respect to the

theory. Separation at the base becomes (again for pairs of arguments): if C and C' are incompatible arguments, that are admissible with respect to a theory Δ, then there are opposites φ and ψ in the theory, such that C supports φ and C' supports ψ.

It is not hard to see that admissibility has the localization and union properties, but lacks the property of separation at the base.

For instance, that for admissibility, the property of separation at the base does not obtain, can be seen by inspection of the theory $\{p_1, p_1 \rightarrow q, p_2, p_2 \rightarrow (q \rightarrow \times q)\}$. With respect to the theory, there are four admissible arguments with a maximal number of elements, viz. each three-element subset of the theory. (Note that each argument of the theory is admissible since there are no attacking arguments.) Any pair of these arguments is incompatible, yet there is no sentence that is defeated by an argument, let alone by an admissible argument, as is required by the property of separation at the base.

That the localization property obtains for admissibility is straightforward: since, when E is an extension of a theory Δ, $J(E) \cap \Delta$ is dialectically justifying with respect to Δ, $J(E) \cap \Delta$ is certainly admissible.

The proof of the union property for admissibility is almost trivial since any attack of the union of a collection of arguments is also an attack of one of the arguments in the collection.

Inspection of the proof of the main theorem of the paper shows that the property of separation at the base is only used in the 'if'-part. The 'only if'-part indeed has an analogue for admissibility since it only uses localization and union. The theory $\{p_1, p_1 \rightarrow q, p_2, p_2 \rightarrow (q \rightarrow \times q)\}$ (the counterexample against the property of separation at the base) shows that the analogue of the 'if'-part is in fact not true. All sentences in the theory are 'admissibly justifiable', i.e., supported by an admissible argument, since any argument of the theory is admissible. No sentence in the theory is 'admissibly defeasible', i.e., attacked by an admissible argument, since there is no attacking argument at all. Still, the theory has no extension.

Verheij (2000a) expands this meta-analysis for other results (e.g., concerning so-called dialectically preferred and admissibly preferred arguments, i.e., those dialectically justifying or admissible arguments that are maximal with respect to set inclusion) and for other notions that are similar to dialectical justification.

Bondarenko *et al.* (1997) have used admissibility in their discussion of an abstract, argumentation-theoretic approach to default reasoning. Their setting is just as Dung's (1995) related to DEFLOG's, yet they focus on deductive systems. Interestingly, whereas in DEFLOG dialectical negation $\times$ is treated as an ordinary connective, Bondarenko *et al.* consider the question which sentences are the contraries of others as part of the domain theory (as the mapping from sentences to their contraries is explicitly represented in their assumption-based frameworks). It seems that the notion of dialectical justification can be directly transplanted to their system. For the reasons, discussed here and in section 3.4, it can be expected that dialectical justification has better properties for analyzing assumption-based frameworks than admissibility.

5 CONCLUSION

A characterization of the existence of extensions (in the context of dialectical argumentation) has been established that shows that the dialectical interpretability of all sentences in a theory suffices for the existence of an extension provided that all dialectical ambiguities in the theory are fixed. A characterization of the multiplicity of extensions immediately follows.

Whether these characterizations should be regarded as solutions to the extension existence and multiplicity problems depends on one's taste. The notion of dialectical interpretation (a typical specimen of the genus of stable extensions) has been connected to the notion of dialectical justification, itself not a very simple notion. The complexity of the problems does however not suggest a really simple solution.

In an important sense, the connection can however be regarded as a reduction: dialectical interpretation is a *global* notion (of interpreting a theory as a whole), whereas dialectical justification is a *local* notion (of interpreting a sentence with respect to a theory). The characterization shows how the global and local notions are connected. The characterization makes it formally precise that local interpretability is a requirement of global interpretability, and that global ambiguity has its roots in local ambiguity.

REFERENCES

Bondarenko, A., Dung, P.M., Kowalski, R.A., and Toni, F. (1997). An abstract, argumentation-theoretic approach to default reasoning. *Artificial Intelligence*, Vol. 93, pp. 63-101.

Brewka, G. Dix, J., and Konolige, K. (1997). *Nonmonotonic reasoning: an overview.* CSLI Publications, Stanford.

Chesñevar, C.I., A.G. Maguitman and R.P. Loui (2000). Logical models of argument. *ACM Computing Surveys*, Vol. 32, No. 4, pp. 337-383.

Dung, P.M. (1995). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and *n*-person games. *Artificial Intelligence*, Vol. 77, pp. 321-357.

Etherington, D. (1987). Formalizing nonmonotonic reasoning. *Artificial Intelligence*, Vol. 31, pp. 41-85.

Gabbay, D.M., Hogger, C.J., and Robinson, J.A. (eds.) (1994). *Handbook of Logic in Artificial Intelligence and Logic Programming. Volume 3. Nonmonotonic Reasoning and Uncertain Reasoning*. Clarendon Press, Oxford.

Gelfond, M. and Lifschitz, V. (1988). The stable model semantics for logic programming. *Logic Programming. Proceedings of the Fifth International Conference and Symposium* (eds. Robert A. Kowalski and Kenneth A. Bowen), pp. 1070-1080. The MIT Press, Cambridge (Massachusetts).

Papadimitriou, C.H. and M. Sideri (1994). Default theories that always have extensions. *Artificial Intelligence*, Vol. 69, p. 347-357.

Pollock, J.L. (1987). Defeasible reasoning. *Cognitive Science*, Vol. 11, pp. 481-518.

Prakken, H., and Vreeswijk, G. (*to appear*). Logics for Defeasible Argumentation. *Handbook of Philosophical Logic* (ed. D. Gabbay). Kluwer Academic Publishers, Dordrecht.

Reiter, R. (1980). A Logic for Default Reasoning. *Artificial Intelligence*, Vol. 13, pp. 81-132.

Toulmin, S.E. (1958). *The uses of argument.* University Press, Cambridge.

Verheij, B. (1999). Automated Argument Assistance for Lawyers. *Proceedings of the Seventh International Conference on Artificial Intelligence and Law*, pp. 43-52. ACM, New York.

Verheij, B. (2000a). DefLog - a logic of dialectical justification and defeat. http://www.metajur.-unimaas.nl/~bart/

Verheij, B. (2000b). Dialectical Argumentation as a Heuristic for Courtroom Decision-Making. *Rationality, Information and Progress in Law and Psychology. Liber Amicorum Hans F. Crombag* (eds. P.J. van Koppen and N.H.M. Roos), pp. 203-226. Metajuridica Publications, Maastricht.

Verheij, Bart (2001). Evaluating arguments based on Toulmin's scheme. *OSSA 2001: Argumentation and its applications (The Ontario Society for the Study of Argumentation).*

Well-Founded Argumentation Semantics for Extended Logic Programming

Ralf Schweimeier and Michael Schroeder
Department of Computing
City University, London, UK
{ralf,msch}@soi.city.ac.uk

Abstract

This paper defines an argumentation semantics for extended logic programming and shows its equivalence to the well-founded semantics with explicit negation. We set up a general framework in which we extensively compare this semantics to other argumentation semantics, including those of Dung, and Prakken and Sartor. We present a general dialectical proof theory for these argumentation semantics.

1 Introduction

Argumentation has attracted much interest in the area of AI. On the one hand, argumentation is an important way of human interaction and reasoning, and is therefore of interest for research into intelligent systems. Application areas include automated negotiation via argumentation [15, 14, 18] and legal reasoning [17]. On the other hand, argumentation provides a formal model for various assumption based (or non-monotonic, or default) reasoning formalisms [4, 5]. In particular, various argumentation based semantics have been proposed for logic programming with default negation [4, 7].

Argumentation semantics are elegant since they can be captured in an abstract framework [7, 4, 20, 12], for which an elegant theory of attack, defence, acceptability, and other notions can be developed, without recourse to the concrete instance of the reasoning formalism at hand. This framework can then be instantiated to various assumption based reasoning formalisms. Similarly, a dialectical proof theory, based on dialogue trees, can be defined for an abstract argumentation framework, and then applied to any instance of such a framework [7, 11].

In general, an argument A is a proof which may use a set of defeasible assumptions. Another argument B may have a conclusion which contradicts the assumptions or the conclusions of A, and thereby B *attacks* A. There are two fundamental notions of such attacks: undercut and rebut [17] or equivalently *ground-attack* and *reductio-ad-absurdum attack* [6]. We will use the terminology of undercuts and rebuts. Both attacks differ in that an undercut attacks a premise of an argument, while a rebut attacks a conclusion.

Given a logic program we can define an argumentation semantics by iteratively collecting those arguments which are acceptable to a proponent, i.e. they can be defended against all opponent attacks. In fact, such a notion of acceptability can be defined in a number of ways depending on which attacks we allow the proponent and opponent to use.

Normal logic programs do not have negative conclusions, which means that we cannot use rebuts. Thus both opponents can only launch undercuts on each other's assumptions. Various argumentation semantics have been defined for normal logic programs [4, 7, 13], some of which are equivalent to existing semantics such as the stable model semantics [9] or the well-founded semantics [8].

Extended logic programs [10, 2, 21], on the other hand, introduce explicit negation, which states that a literal is explicitly false. As a result, both undercuts and rebuts are possible forms of attack; there are further variations depending on whether any kind of counter-attack is admitted. A variety of argumentation semantics arise if one allows one notion of attack as defence for the proponent, and another as attack for the opponent. Various argumentation semantics have been proposed for extended logic programs [6, 17]. Dung has shown that a certain argumentation semantics is equivalent to the answer set semantics [10], a generalisation of the stable model semantics [9]. To our knowl-

edge, no argumentation semantics has yet been found equivalent to the well-founded semantics for extended logic programs, WFSX [16, 2].

This paper makes the following contributions: we define a least fixpoint argumentation semantics for extended logic programs, and show its equivalence to the well-founded semantics with explicit negation [16, 2, 1]. In order to relate this semantics to other argumentation semantics, we set up a general framework to classify notions of justified arguments, and use it to compare our argumentation semantics to those of Dung [6] and Prakken and Sartor [17] among others. We develop a general dialectical proof theory for the notions of justified arguments we introduce.

The paper is organised as follows: First we define arguments and notions of attack and acceptability. Then we set up a framework for classifying different least fixpoint argumentation semantics, based on different notions of attack. In Section 4, we recall the definition of WFSX, and in Section 5, we prove the equivalence of an argumentation semantics and WFSX. A general dialectical proof theory for arguments is presented in Section 6, and its soundness and completeness is proven.

2 Extended Logic Programming and Argumentation

We summarise the definitions of arguments for extended logic programs, and define various notions of attack between arguments.

2.1 Arguments

Definition 1 *An* objective literal *is an atom A or its explicit negation $\neg A$. We define $\neg\neg L = L$. A* default literal *is of the form not L where L is an objective literal. A* literal *is either an objective or a default literal. An* extended logic program *is a (possibly infinite) set of rules of the form*

$$L_0 \leftarrow L_1, \ldots, L_m, not\ L_{m+1}, \ldots, not\ L_{m+n}$$

where $m, n \geq 0$, and each L_i is an objective literal $(0 \leq i \leq m+n)$.
For such a rule r, we call L_0 the head *of the rule, $head(r)$, and $L_1, \ldots, not\ L_{m+n}$ the* body *of the rule, $body(r)$.*

Our definition of an argument for an extended logic program is based on [17]. Essentially, an argument is a partial proof, resting on a number of *assumptions*, i.e. a set of default literals.[1]

Note that we do not consider priorities of arguments, as used e.g. in [17, 20].

Definition 2 *Let P be an extended logic program. An* argument *for P is a finite sequence $A = [r_1, \ldots r_n]$ of rules $r_i \in P$ such that for every $1 \leq i \leq n$, for every objective literal L_j in the body of r_i there is a $k > i$ such that $head(r_k) = L_j$.*
A subargument *of A is a subsequence of A which is an argument. The head of a rule in A is called a* conclusion *of A, and a default literal not L in the body of a rule of A is called an* assumption *of A. We write $assm(A)$ for the set of assumptions and $conc(A)$ for the set of conclusions of an argument A.*
An argument A with a conclusion L is a minimal argument for L *if there is no subargument of A with conclusion L. An argument is* minimal *if it minimal for some literal L. Given an extended logic program P, we denote the set of minimal arguments for P by $Args_P$.*

The restriction to minimal arguments is not essential, but convenient, since it rules out arguments constructed from several unrelated arguments. Generally, one is interested in the conclusions of an argument, and wants to avoid having rules in an argument which do not contribute to the desired conclusion.

2.2 Notions of Attack

There are two fundamental notions of attack: *undercut*, which invalidates an assumption of an argument, and *rebut*, which contradicts a conclusion of an argument [6, 17]. From these, we may define further notions of attack, by allowing either of the two fundamental kinds of attack, and considering whether any kind of counter-attack is allowed or not. We will now formally define these notions of attacks.

Definition 3 *Let A_1 and A_2 be arguments.*

1. *A_1* undercuts *A_2 if there is an objective literal L such that L is a conclusion of A_1 and not L is an assumption of A_2.*
2. *A_1* rebuts *A_2 if there is an objective literal L such that L is a conclusion of A_1 and $\neg L$ is a conclusion of A_2.*
3. *A_1* attacks *A_2 if A_1 undercuts or rebuts A_2.*
4. *A_1* defeats *A_2 if A_1 undercuts A_2, or (A_1 rebuts A_2 and A_2 does not undercut A_1).*

[1] In [4, 6], an argument *is* a set of assumptions; the two approaches are equivalent in that there is an argument with a conclusion L iff there is a set of assumptions from which L can be inferred. See the discussion in [17].

5. A_1 strongly attacks A_2 *if* A_1 *attacks* A_2 *and* A_2 *does not undercut* A_1.

6. A_1 strongly undercuts A_2 *if* A_1 *undercuts* A_2 *and* A_2 *does not undercut* A_1.

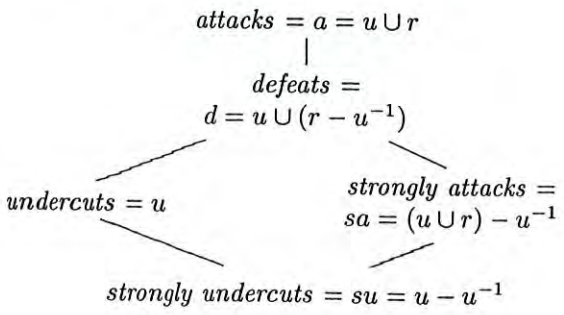
The notions of *undercut* and *rebut*, and hence *attack* are fundamental for extended logic programs [6, 17]. The notion of *defeat* is used in [17], along with a notion of *strict defeat*, i.e. a defeat that is not counter-defeated. For arguments without priorities, rebuts are symmetrical, and therefore strict defeat coincides with strict undercut, i.e. an undercut that is not counter-undercut. Similarly, strict attack coincides with strict undercut. For this reason, we use the term *strong undercut* instead of *strict undercut*, and similarly define *strong attack* to be an attack which is not counter-undercut. We will use the following abbreviations for these notions of attack. r for rebuts, u for undercuts, a for attacks, d for defeats, sa for strongly attacks, and su for strongly undercuts.

These notions of attack define for any extended logic program a binary relation on the set of arguments of that program.

Definition 4 *A* notion of attack *is a function x which assigns to each extended logic program P a binary relation x_P on the set of arguments of P, i.e. $x_P \subseteq Args_P^2$. Notions of attack can be partially ordered by defining $x \subseteq y$ iff $\forall P : x_P \subseteq y_P$*

Definition 5 *Let x be a notion of attack. Then the* inverse *of x, denoted by x^{-1}, is defined as $x_P^{-1} = \{(B,A) \mid (A,B) \in x_P\}$.*

In this relational notation, Definition 3 can be rewritten as $a = u \cup r$, $d = u \cup (r - u^{-1})$, $sa = (u \cup r) - u^{-1}$, and $su = u - u^{-1}$. Using the set-theoretic laws $A - B \subseteq A \subseteq A \cup C$ and $(A \cup B) - C = (A - C) \cup (B - C)$ (for all sets A, B, and C), it is easy to see that the notions of attack of Definition 3 are partially ordered according to the following Hasse diagram.



This diagram contains the notions of attack used in [6, 17], plus *strongly attacks* which seemed a natural intermediate notion between *strongly undercuts* and *defeats*. We have not included *rebuts*, because in the absence of priorities, *rebuts* is somewhat weaker than *undercuts*, because it is symmetric: a rebut is always counter-rebutted, while the same does not hold for *undercuts*.

2.3 Acceptability and Justified Arguments

Given the above notions of attack, we define acceptability of an argument. Basically, an argument is acceptable if it can be defended against any attack. Depending on which particular notion of attack we use as defence and which for the opponent's attacks, we obtain a host of acceptability notions.

Acceptability forms the basis for our argumentation semantics, which is defined as the least fixpoint of a function, which collects all acceptable arguments. The *least* fixpoint is of particular interest [17, 6], because it provides a canonical fixpoint semantics and it can be constructed inductively.

Definition 6 *Let x and y be notions of attack. Let A be an argument, and S a set of arguments. Then A is* x/y-acceptable wrt. *S if for every argument B such that $(B,A) \in x$ there exists an argument $C \in S$ such that $(C,B) \in y$.*

Based on the notion of acceptability, we can then define a fixpoint semantics for arguments.

Definition 7 *Let x and y be notions of attack, and P an extended logic program. The operator $F_{P,x/y} : \mathcal{P}(Args_P) \to \mathcal{P}(Args_P)$ is defined as*

$$F_{P,x/y}(S) = \{A \mid A \text{ is } x/y\text{-acceptable wrt. } S\}$$

We denote the least fixpoint of $F_{P,x/y}$ by $J_{P,x/y}$. If the program P is clear from the context, we omit the subscript P. An argument A is called x/y-justified if $A \in J_{x/y}$; an argument is called x/y-overruled if it is attacked by an x/y-justified argument; and an argument is called x/y-defensible if it is neither x/y-justified nor x/y-overruled.

For any program P, the least fixpoint exists by the Knaster-Tarski fixpoint theorem [19, 3], because $F_{P,x/y}$ is monotone. It can be constructed by transfinite induction as follows:

$$\begin{array}{lll} J_{x/y}^0 & = \emptyset & \\ J_{x/y}^{\alpha+1} & = F_{P,x/y}(J_{x/y}^\alpha) & \text{for } \alpha+1 \text{ a successor ordinal} \\ J_{x/y}^\lambda & = \bigcup_{\alpha<\lambda} J_{x/y}^\alpha & \text{for } \lambda \text{ a limit ordinal} \end{array}$$

Then there exists a least ordinal λ_0 such that $F_{x/y}(J^{\lambda_0}_{x/y}) = J^{\lambda_0}_{x/y} = J_{x/y}$.

3 Relationships of Notions of Justifiability

This section is devoted to an analysis of the relationship between the different notions of justifiability, leading to a hierarchy of notions of justifiability illustrated in Figure 2.

First of all, it is easy to see that the least fixpoint increases if we weaken the attacks, or strengthen the defence.

Proposition 1 *Let $x' \subseteq x$, $y \subseteq y'$ be notions of attack, then $J_{x/y} \subseteq J_{x'/y'}$.*

Theorem 2 states that it does not make a difference if we allow only the strong version of the defence. This is because an argument need not defend itself on its own, but it may rely on other arguments to defend it.

We only give a formal proof for the first theorem; the proofs for the other theorems are similar, and we provide an intuitive informal explanation instead.

Theorem 2 *Let x and y be notions of attack such that $x \supseteq$ undercuts, and let $sy = y - undercuts^{-1}$. Then $J_{x/y} = J_{x/sy}$.*

Proof. Informally, every x-attack B to an x/y-justified argument A is y-defended by some x/sy-justified argument C (by induction). Now if C was *not* a sy-attack, then it is undercut by B, and because $x \supseteq$ *undercuts* and C is justified, there exists a *strong* defence for C against B, which is also a defence of the original argument A against C.

The formal proof is by transfinite induction. By Proposition 1, we have $J_{x/sy} \subseteq J_{x/y}$. We prove the inverse inclusion by showing that for all ordinals α: $J^{\alpha}_{x/y} \subseteq J^{\alpha}_{x/sy}$, by transfinite induction on α.

Base case $\alpha = 0$: $J_{x/y} = \emptyset = J_{x/sy}$.

Successor ordinal $\alpha \rightsquigarrow \alpha + 1$: Let $A \in J^{\alpha+1}_{x/y}$, and $(B, A) \in x$. By definition, there exists $C \in J^{\alpha}_{x/y}$ such that $(C, B) \in y$. By induction hypothesis, $C \in J^{\alpha}_{x/sy}$.

If B does not undercut C, then we are done. If, however, B undercuts C, then because $C \in J^{\alpha}_{x/sy}$, and *undercuts* $\subseteq x$, there exists $D \in J^{\alpha_0}_{x/sy}(\emptyset)(\alpha_0 < \alpha)$ such that $(D, B) \in sy$. It follows that $A \in J^{\alpha+1}_{x/sy}$.

Limit ordinal λ: Assume $J^{\alpha}_{x/y} \subseteq J^{\alpha}_{x/sy}$ for all $\alpha < \lambda$. Then $J^{\lambda}_{x/y} = \bigcup_{\alpha<\lambda} J^{\alpha}_{x/y} \subseteq \bigcup_{\alpha<\lambda} J^{\alpha}_{x/sy} = J^{\lambda}_{x/sy}$ □

In particular, the previous Theorem states that undercut and strong undercut are equivalent as a defence, as are attack and strong attack. This may be useful in an implementation, where we may use the stronger notion of defence without changing the semantics, thereby decreasing the number of arguments to be checked. The following Corollary shows that because defeat lies between attack and strong attack, it is equivalent to both as a defence.

Corollary 3 *Let x be a notion of attack such that $x \supseteq$ undercuts. Then $J_{x/a} = J_{x/d} = J_{x/sa}$.*

Proof. With Proposition 1 and Theorem 2, we have $J_{x/a} \subseteq J_{x/d} \subseteq J_{x/sa} = J_{x/a}$. □

Theorem 4 *Let x be a notion of attack such that $x \supseteq$ strongly attacks. Then $J_{x/u} = J_{x/d} = J_{x/a}$.*

Proof. Every x-attack B to a x/a-justified argument a is attacked by some x/u-justified argument C (by induction). If C is a rebut, but not an undercut, then because B strongly attacks C, and because $x \supseteq$ *strongly attacks*, there must have been an argument defending C by undercutting B, thereby also defending A against B.

The statement for *defeats* follows in a similar way to Corollary 3.

□

Theorem 5 $J_{sa/su} = J_{sa/sa}$

The proof is similar to Theorem 4.

Theorem 6 $J_{su/a} = J_{su/d}$

Proof. Every strong undercut B to a su/a-justified argument A is attacked by some su/d-justified argument C (by induction). If C does not defeat A, then there is some argument D defending C by defeating B, thereby also defending A against B. □

We will now present some example programs which distinguish various notions of justifiability.

Example 1 *Consider P_1 in Figure 1. For any notion of attack x, we have $J_{su/x} = J_{sa/x} = \{[p \leftarrow not\ q], [q \leftarrow not\ p]\}$, because there is no strong undercut or strong attack to any of the arguments. However, $J_{a/x} = J_{d/x} = J_{u/x} = \emptyset$, because every argument is undercut (and therefore defeated and attacked).*

Example 2 *Consider P_2 in Figure 1. Let x be a notion of attack. Then $J_{d/x} = J_{a/x} = \emptyset$, because every argument is defeated (hence attacked).*

$P_1 =$	$P_2 =$	$P_3 =$
$p \leftarrow not\ q$	$p \leftarrow not\ q$	$p \leftarrow not\ q$
$q \leftarrow not\ p$	$q \leftarrow not\ p$	$q \leftarrow not\ r$
	$\neg p$	$r \leftarrow not\ s$
		$s \leftarrow not\ p$
		$\neg p$
$P_4 =$	$P_5 =$	$P_6 =$
$p \leftarrow not\ q$	$p \leftarrow not\ \neg p$	$\neg p \leftarrow not\ q$
$q \leftarrow not\ p$	$\neg p$	$\neg q \leftarrow not\ p$
$r \leftarrow not\ p$		p
		q

Figure 1: Examples

$J_{sa/su} = J_{sa/sa} = \{[q \leftarrow not\ p]\}$, *because* $[q \leftarrow not\ p]$ *is the only argument which is not strongly attacked, but it does not strongly attack any other argument.* $J_{u/su} = J_{u/u} = \{[\neg p]\}$, *because there is no undercut to* $[\neg p]$, *but* $[\neg p]$ *does not undercut any other argument.* $J_{u/a} = \{[\neg p], [q \leftarrow not\ p]\}$, *because there is no undercut to* $[\neg p]$, *and the undercut* $[p \leftarrow not\ p]$ *to* $[q \leftarrow not\ p]$ *is attacked by* $[\neg p]$. *We also have* $J_{sa/u} = \{[\neg p], [q \leftarrow not\ p]\}$, *because* $[q \leftarrow not\ p]$ *is not strongly attacked, and the strong attack* $[p \leftarrow not\ q]$ *on* $[\neg p]$ *is undercut by* $[q \leftarrow not\ p]$.

Example 3 *Consider* P_3 *in Figure 1. Let* x *be a notion of attack. Then* $J_{sa/x} = \emptyset$, *because every argument is strongly attacked.*

$J_{su/u} = J_{su/su} = \{[\neg p]\}$, *because all arguments except* $[\neg p]$ *are strongly undercut, but* $[\neg p]$ *does not undercut any argument. And* $J_{u/a} = J_{su/sa} = J_{su/a} = \{[\neg p], [q \leftarrow not\ r], [s \leftarrow not\ p]\}$.

Example 4 *Consider* P_4 *in Figure 1. Let* x *be a notion of attack. Then* $J_{u/x} = J_{d/x} = J_{a/x} = \emptyset$, *because every argument is undercut.* $J_{su/su} = J_{su/sa} = J_{sa/su} = J_{sa/sa} = \{[p \leftarrow not\ q], [q \leftarrow not\ p]\}$ *In this case, the strong attacks are precisely the strong undercuts; The argument* $[r \leftarrow not\ p]$ *is not justified, because the strong undercut* $[p \leftarrow not\ q]$ *is undercut, but not strongly undercut, by* $[q \leftarrow not\ p]$. $J_{su/u} = J_{su/a} = J_{sa/u} = J_{sa/a} = \{[p \leftarrow not\ q], [q \leftarrow not\ p], [r \leftarrow not\ p]\}$ *Again, undercuts and attacks, and strong undercuts and strong attacks, coincide; but now* $[r \leftarrow not\ p]$ *is justified, because non-strong undercuts are allowed as defence.*

Example 5 *Consider* P_5 *in Figure 1. Then* $J_{a/x} = \emptyset$, *because both arguments attack each other, while* $J_{d/x} = \{[\neg p]\}$, *because* $[\neg p]$ *defeats* $[p \leftarrow not\ \neg p]$, *but not vice versa.*

Example 6 *Consider* P_6 *in Figure 1. Let* x *be a notion of attack. Then* $J_{sa/x} = J_{d/x} = J_{a/x} = \emptyset$, *because every argument is strongly attacked (hence defeated and attacked), while* $J_{u/x} = J_{su/x} = \{[p], [q]\}$.

Theorem 7 *The notions of justifiability are ordered (by set inclusion) according to the Hasse diagram in Figure 2.*

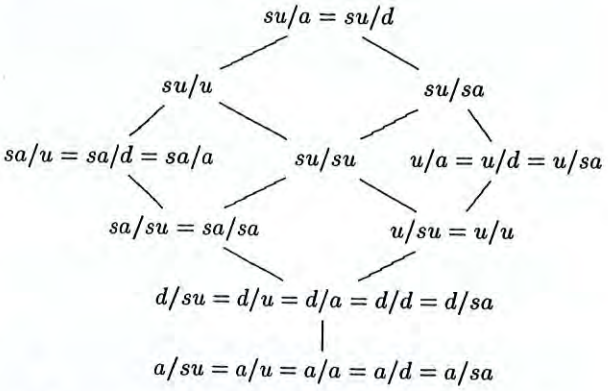


Figure 2: Hierarchy of Notions of Justifiability

By definition, Dung's grounded argumentation semantics [6] is exactly a/u-justifiability, while Prakken and Sartor's semantics [17], if we disregard priorities, amounts to d/su-justifiability. As corollaries to Theorem 7, we obtain relationships of these semantics to the other notions of justifiability.

Corollary 8 *Let* J_{Dung} *be the set of justified arguments according to Dung's grounded argumentation semantics [6]. Then* $J_{Dung} = J_{a/su} = J_{a/u} = J_{a/a} = J_{a/d} = J_{a/sa}$ *and* $J_{Dung} \subseteq J_{x/y}$ *for all notions of attack* x *and* y.

Corollary 9 *Let* J_{PS} *be the set of justified arguments according to Prakken and Sartor's argumentation semantics [17], where all arguments have the same priority. Then* $J_{PS} = J_{d/su} = J_{d/u} = J_{d/a} = J_{d/d} = J_{d/sa}$, $J_{PS} \subseteq J_{x/y}$ *for all notions of attack* $x \neq a$ *and* y, *and* $J_{PS} \supseteq J_{a/y}$ *for all notions of attack* y.

Remark 1 *1. The notions of* a/x-, d/x- *and* sa/x-*justifiability are very sceptical in that a* fact p *may not be justified, if there is a rule* $\neg p \leftarrow B$ *(where not* $p \notin B$*) that is not* x-*attacked. On the other hand this is useful in terms of avoiding inconsistency.*

2. sx/y-*justifiability is very credulous, because it does not take into account non-strong attacks, so e.g. the program* $\{p \leftarrow not\ q, q \leftarrow not\ p\}$ *has the justified arguments* $[p \leftarrow not\ q]$ *and* $[q \leftarrow not\ p]$.

Remark 2 *One might ask whether any of the semantics in Figure 2 are equivalent for* non-contradictory

programs, i.e. programs for which there is no literal L such that there exist justified arguments for both L and $\neg L$. The answer to this question is no: all the examples above distinguishing different notions of justifiability involve only non-contradictory programs.

In particular, even for non-contradictory programs, Dung's and Prakken and Sartor's semantics differ, and both differ from u/a-justifiability, which will be shown equivalent to the well-founded semantics WFSX [16, 2] in the following section.

4 Well-founded semantics

We recollect the definition of the well-founded semantics for extended logic programs, WFSX. We use the definition of [1], because it is closer to our definition of argumentation semantics than the original definition of [16, 2].

Definition 8 *The set of all objective literals of a program P is called the* Herbrand base *of P and denoted by $\mathcal{H}(P)$. A* pseudo-interpretation *of a program P is a set $T \cup not\ F$ where T and F are subsets of $\mathcal{H}(P)$. An* interpretation *is a pseudo-interpretation where the sets T and F are disjoint. An interpretation is called* two-valued *if $T \cup F = \mathcal{H}(P)$.*

Definition 9 *Let P be an extended logic program, I an interpretation, and let P' (resp. I') be obtained from P (resp. I) by replacing every literal $\neg A$ by a new atom, say $\neg_A$. The GL-transformation $\frac{P'}{I'}$ is the program obtained from P' by removing all rules containing a default literal not A such that $A \in I'$, and then removing all remaining default literals from P', obtaining a definite program P''. Let J be the least model of P''. $\Gamma_P I$ is obtained from J by replacing the introduced atoms $\neg_A$ by $\neg A$.*

Definition 10 *The* semi-normal *version of a program P is the program P_s obtained from P by replacing every rule $L \leftarrow Body$ in P by the rule $L \leftarrow not\ \neg L, Body$.*

If the program P is clear from the context, we write ΓI for $\Gamma_P I$ and $\Gamma_s I$ for $\Gamma_{P_s} I$.

Definition 11 *Let P be a program whose least fixpoint of $\Gamma\Gamma_s$ is T. Then the* paraconsistent well-founded model of P *is the pseudo-interpretation $WFM_p(P) = T \cup not\ (\mathcal{H}(P) - \Gamma_s T)$. If $WFM_p(P)$ is an interpretation, then P is called* non-contradictory, *and $WFM_p(P)$ is the* well-founded model of P, *denoted by $WFM(P)$.*

The paraconsistent well-founded model can by defined iteratively by the transfinite sequence $\{I_\alpha\}$:

$$
\begin{array}{lcll}
I_0 & = & \emptyset & \\
I_{\alpha+1} & = & \Gamma\Gamma_s I_\alpha & \text{for successor ordinal } \alpha + 1 \\
I_\lambda & = & \bigcup_{\alpha<\lambda} I_\alpha & \text{for limit ordinal } \lambda
\end{array}
$$

There exists a smallest ordinal λ_0 such that I_{λ_0} is the least fixpoint of $\Gamma\Gamma_s$, and $WFM_p(P) = I_{\lambda_0} \cup not\ (\mathcal{H}(P) - \Gamma_s I_{\lambda_0})$.

5 Equivalence of argumentation semantics and WFSX

In this section, we will show that the argumentation semantics $J_{u/a}$ and the well-founded model coincide. That is, the conclusions of justified arguments are exactly the objective literals which are true in the well-founded model; and those objective literals all of whose arguments are overruled are exactly the literals which are false in the well-founded model. The result holds also for contradictory programs under the *paraconsistent* well-founded semantics. This is important, because it shows that contradictions in the argumentation semantics are precisely the contradictions under the well-founded semantics, and allows the application of contradiction removal (or avoidance) methods to the argumentation semantics. Because for non-contradictory programs, the well-founded semantics coincides with the paraconsistent well-founded semantics [1], we obtain as a corollary that argumentation semantics and well-founded semantics coincide for non-contradictory programs.

In order to compare the argumentation semantics with the well-founded semantics, we define the set of literals which are a consequence of the argumentation semantics.

Definition 12 *$A(P) = T \cup not\ F$, where $T = \{L \mid$ there is a justified argument for $L\}$ and $F = \{L \mid$ all arguments for L are overruled $\}$.*

The following Proposition shows a precise connection between arguments and consequences of a program $\frac{P}{I}$.

Proposition 10 *Let I be a two-valued interpretation.*

1. *$L \in \Gamma(I)$ iff $\exists$ argument A with conclusion L such that $assm(A) \subseteq I$.*
2. *$L \in \Gamma_s(I)$ iff $\exists$ argument A with conclusion L such that $assm(A) \subseteq I$ and $\neg conc(A) \cap I = \emptyset$.*
3. *$L \notin \Gamma(I)$ iff $\forall$ arguments A with conclusion L, $assm(A) \cap I \neq \emptyset$.*
4. *$L \notin \Gamma_s(I)$ iff $\forall$ arguments A with conclusion L, $assm(A) \cap I \neq \emptyset$ or $\neg conc(A) \cap I \neq \emptyset$.*

Proof. See Appendix. □

Theorem 11 *Let P be an extended logic program. Then $WFM_p(P) = A(P)$.*

Proof. First, note that A undercuts B iff $\exists\ L$ s.t. *not* $L \in assm(A)$ and $L \in conc(B)$; and A rebuts B iff $\exists\ L \in conc(A) \cap \neg conc(B)$.

We show that for all ordinals α, $I_\alpha = A_\alpha$, by transfinite induction on α.

Base case $\alpha = 0$: $I_\alpha = \emptyset = A_\alpha$

Successor ordinal $\alpha \rightsquigarrow \alpha + 1$:

$L \in I_{\alpha+1}$

iff (Def. of $I_{\alpha+1}$)

$L \in \Gamma\Gamma_s I_\alpha$

iff (Prop. 10(1))

$\exists$ argument A for L such that $assm(A) \subseteq \Gamma_s I_\alpha$

iff (Def. of $\subseteq$, and $\Gamma_s I_\alpha$ is two-valued)

$\exists$ argument A for L such that $\forall$ *not* $L \in assm(A), L \notin \Gamma_s I_\alpha$

iff (Prop. 10(4))

$\exists$ argument A for L such that $\forall$ *not* $L \in assm(A)$, for any argument B for L,
($\exists$ *not* $L' \in assm(B)$ *s.t.* $L' \in I_\alpha$ or
$\exists\ L'' \in conc(B)$ *s.t.* $\neg L'' \in I_\alpha$)

iff (Induction hypothesis)

$\exists$ argument A for L such that $\forall$ *not* $L \in assm(A)$, for any argument B for L, ($\exists$ *not* $L' \in assm(B)$ *s.t.* $\exists$ argument $C \in J_\alpha$ for L', or
$\exists\ L'' \in conc(B)$ *s.t.* $\exists$ argument $C \in J_\alpha$ for $\neg L''$)

iff (Def. of undercut and rebut)

$\exists$ argument A for L such that for any undercut B to A, ($\exists$ argument $C \in J_\alpha$ s.t. C undercuts B, or
$\exists$ argument $C \in J_\alpha$ s.t. C rebuts B)

iff

$\exists$ argument A for L such that for any undercut B to A, $\exists$ argument $C \in J_\alpha$ s.t. C attacks B

iff (Def. of $J_{\alpha+1}$)

$\exists$ argument $A \in J_{\alpha+1}$ for L

iff (Def. of $A_{\alpha+1}$)

$L \in A_{\alpha+1}$

Limit ordinal λ:

$I_\lambda = \bigcup_{\alpha<\lambda} I_\alpha$ and $A_\lambda = \bigcup_{\alpha<\lambda} A_\alpha$, so by induction hypothesis ($I_\alpha = A_\alpha$ for all $\alpha < \lambda$), $I_\lambda = A_\lambda$.

Now, we show that a literal *not* L is in the well-founded semantics iff every argument for L is overruled.

not $L \in WFM_p(P)$

iff (Def. of $WFM_p(P)$)

$L \notin \Gamma_s I$

iff (Prop. 10(4))

for all arguments A for L,
($\exists$ *not* $L' \in assm(A)$ *s.t.* $L' \in I$, or
$\exists\ L'' \in conc(A)$ *s.t.* $\neg L'' \in I$)

iff (I=A)

for all arguments A for L,
($\exists$ *not* $L' \in assm(A)$ *s.t.* $\exists$ argument $B \in J$ for L', or $\exists\ L'' \in conc(A)$ *s.t.* $\exists$ argument $B \in J$ for $\neg L''$)

iff (Def. of undercut and rebut)

for all arguments A for L, ($\exists$ argument $B \in J$ s.t. B undercuts A, or $\exists$ argument $B \in J$ s.t. B rebuts A)

iff

every argument for L is attacked by a justified argument in J

iff (Def. of overruled)

every argument for L is overruled

iff (Def. of $A(P)$)

not $L \in A(P)$ □

Corollary 12 *Let P be a non-contradictory program. Then $WFM(P) = A(P)$.*

Remark 3 *In a similar way, one can show that the Γ operator corresponds to undercuts, while the Γ_s operator corresponds to attacks, and so the least fixpoints of $\Gamma\Gamma$, $\Gamma_s\Gamma$, and $\Gamma_s\Gamma_s$ correspond to $J_{u/u}$, $J_{a/u}$, and $J_{a/a}$, respectively. In [1], the least fixpoints of these operators are shown to be ordered as $lfp(\Gamma_s\Gamma) \subseteq lfp(\Gamma_s\Gamma_s) \subseteq lfp(\Gamma\Gamma_s)$, and $lfp(\Gamma_s\Gamma) \subseteq lfp(\Gamma\Gamma) \subseteq lfp(\Gamma\Gamma_s)$. Because $J_{a/u} = J_{a/a} \subseteq J_{u/u} \subseteq J_{u/a}$ by Theorem 7, we can strengthen this statement to $lfp(\Gamma_s\Gamma) = lfp(\Gamma_s\Gamma_s) \subseteq lfp(\Gamma\Gamma) \subseteq lfp(\Gamma\Gamma_s)$.*

6 Proof theory

One of the benefits of relating the argumentation semantics $J_{u/a}$ to WFSX is the existence of an efficient top-down proof procedure for WFSX [1], which we can use to compute justified arguments in $J_{u/a}$. On the other hand, *dialectical* proof theories, based on dialogue trees, have been defined for a variety of argumentation semantics [17, 11, 13]. In this section, we present a sound and complete dialectical proof theory for the least fixpoint argumentation semantics $J_{x/y}$ for any notions of attack x and y. Our presentation closely follows [17]. As a further consequence, we obtain an equivalence of the proof theory for WFSX and the dialectical proof theory for arguments.

Definition 13 *An* x/y-dialogue *is a finite nonempty sequence of moves $move_i = (Player_i, Arg_i)(i > 0)$, such that*

1. *$Player_i = P$ iff i is odd; and $Player_i = O$ iff i is even.*

2. *If* $Player_i = Player_j$ *and* $i \neq j$, *then* $Arg_i \neq Arg_j$.
3. *If* $Player_i = P$ *and* $i > 1$, *then* Arg_i *is a minimal argument such that* $(Arg_i, Arg_{i-1}) \in y$.
4. *If* $Player_i = O$, *then* $(Arg_i, Arg_{i-1}) \in x$.

The first condition states that the players P (Proponent) and O (Opponent) take turns, and P starts. The second condition prevents the proponent from repeating a move. The third and fourth conditions state that both players have to attack the other player's last move, where the opponent is allowed to use the notion of attack x, while the proponent may use y to defend its arguments.

Definition 14 *An* x/y-dialogue tree *is a tree of moves such that every branch is a dialogue, and for all moves* $move_i = (P, Arg_i)$, *the children of* $move_i$ *are all those moves* (O, Arg_j) *such that* $(Arg_j, Arg_i) \in x$.

Definition 15 *A player* wins an x/y-dialogue *iff the other player cannot move. A player* wins an x/y-dialogue tree *iff it wins all branches of the tree. An* x/y*-dialogue tree which is won by the proponent is called a* winning x/y-dialogue tree. *An argument* A *is* provably x/y-justified *iff there exists a* x/y*-tree with* A *as its root, and won by the proponent. A literal* L *is a* provably justified conclusion *iff it is a conclusion of a provably* x/y*-justified argument. The* height *of a dialogue tree is* 0 *if it consists only of the root, and otherwise* $height(t) = \bigcup height(t_i) + 1$ *where* t_i *are the trees rooted at the grandchildren of* t.

We show that the proof theory of x/y-dialogue trees is sound and complete for any notions of attack x and y.

Theorem 13 *An argument is provably* x/y*-justified iff it is* x/y*-justified.*

Proof. "If"-direction. We show by transfinite induction: If $A \in J^{\alpha}_{x/y}$, then there exists a winning x/y-dialogue tree of height $< \alpha$ for A.

Base case $\alpha = 0$*:*
Then there exists no argument B such that $(B, A) \in x$, and so A is a winning x/y-dialogue tree for A of height 0.

Successor ordinal $\alpha + 1$*:*
If $A \in J^{\alpha+1}_{x/y}$, then for any B_i such that $(B_i, A) \in x$ there exists a $C_i \in J^{\alpha}_{x/y}$ such that $(C_i, B_i) \in y$. By induction hypothesis, there exist winning x/y-dialogue trees for the C_i. Thus, we have a winning tree rooted for A, with children B_i, whose children are the winning trees for C_i.

Limit ordinal λ*:*
If $A \in J^{\lambda}_{x/y}$, then there exists an $\alpha < \lambda$ such that $A \in J^{\alpha}_{x/y}$; by induction hypothesis, there exists a winning x/y-dialogue tree of height α for A.

"Only-if"-direction. We prove by transfinite induction: If there exists a winning tree of height α for A, then $A \in J^{\alpha}_{x/y}$.

Base case $\alpha = 0$*:*
Then there are no arguments B such that $(B, A) \in x$, and so $A \in J^{0}_{x/y}$.

Successor ordinal $\alpha + 1$*:*
Let T be a tree with root A, whose children are B_i, and the children of B_i are winning trees rooted at C_i. By induction hypothesis, $C_i \in J^{\alpha}_{x/y}$. Because the B_i are all those arguments such that $(B_i, A) \in x$, then A is defended against each B_i by C_i, and so $A \in J^{\alpha+1}_{x/y}$. □

As a corollary, we can relate the proof theory of WFSX and the u/a-proof theory.

Corollary 14 *L is a provably u/a-justified conclusion iff there exists a successful T-tree [2] for L.*

Proof. Follows from the fact that u/a-dialogue trees are sound and complete for u/a-justifiability (Theorem 13), that T-trees are sound and complete for WFSX [2], and that u/a-justifiability and WFSX are equivalent (Theorem 11). □

7 Conclusion and Further Work

We have identified various notions of attack for extended logic programs. Based on these notions of attack, we defined notions of acceptability and least fixpoint semantics. These fixpoint semantics were related by establishing a lattice of justified arguments, based on set inclusion. We identified an argumentation semantics $J_{u/a}$ equal to the well-founded semantics for logic programs with explicit negation, $WFSX$ [2], and established that $J_{Dung} \subseteq J_{PS} \subseteq J_{u/a} = WFSX$, where J_{Dung} and J_{PS} are the least fixpoint argumentation semantics of Dung [6] and Prakken and Sartor [17]. We have defined a dialectical proof theory for argumentation. For all notions of justified arguments introduced, we prove that the proof theory is sound and complete wrt. the corresponding fixpoint argumentation semantics. In particular, we showthe equivalence of successful T-trees [2] in WFSX to provably u/a justified arguments.

Finally, it remains to be seen whether a variation in the notion of attack yields interesting variations of alternative argumentation semantics for extended logic

programs such as preferred extensions or stable extensions [6]. It is also an open question how the hierarchy changes when priorities are added as defined in [17, 20].

Acknowledgement This work has been supported by EPSRC grant GRM88433.

References

[1] J. J. Alferes, C. V. Damásio, and L. M. Pereira. A logic programming system for non-monotonic reasoning. *Journal of Automated Reasoning*, 14(1):93–147, 1995.

[2] J. J. Alferes and L. M. Pereira. *Reasoning with Logic Programming.* (LNAI 1111), Springer-Verlag, 1996.

[3] Garrett Birkhoff. *Lattice Theory.* American Mathematical Society, 3rd edition, 1967.

[4] A. Bondarenko, P.M. Dung, R.A. Kowalski, and F. Toni. An abstract, argumentation-theoretic approach to default reasoning. *Artificial Intelligence*, 93(1-2):63–101, 1997.

[5] Carlos Iván Chesñevar, Ana Gabriela Maguitman, and Ronald Prescott Loui. Logical models of argument. *ACM Computing Surveys*, 32(4):337–383, December 2000.

[6] P. M. Dung. An argumentation semantics for logic programming with explicit negation. In *Proc. of the 10th International Conference on Logic Programming*, pages 616–630. MIT Press, 1993.

[7] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2):321–357, 1995.

[8] Allen Van Gelder, Kenneth A. Ross, and John S. Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM*, 38(3):620–650, July 1991.

[9] M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In R. A. Kowalski and K. A. Bowen, editors, *5th International Conference on Logic Programming*, pages 1070–1080. MIT Press, 1988.

[10] M. Gelfond and V. Lifschitz. Logic programs with classical negation. In *Proc. of ICLP90*, pages 579–597. MIT Press, 1990.

[11] H. Jakobovits and D. Vermeir. Dialectic semantics for argumentation frameworks. In *Proceedings of the Seventh International Conference on Artificial Intelligence and Law (ICAIL '99)*, pages 53–62, 1999.

[12] H. Jakobovits and D. Vermeir. Robust semantics for argumentation frameworks. *Journal of Logic and Computation*, 9(2):215–261, 1999.

[13] A.C. Kakas and F. Toni. Computing argumentation in logic programming. *Journal of Logic and Computation*, 9:515–562, 1999.

[14] S. Kraus, K. Sycara, and A. Evenchik. Reaching agreements through argumentation: a logical model and implementation. *Artificial Intelligence*, 104(1-2):1–69, 1998.

[15] Simon Parsons, Carlos Sierra, and Nick Jennings. Agents that reason and negotiate by arguing. *Journal of Logic and Computation*, 8(3):261–292, 1998.

[16] Luis Moniz Pereira and José Júlio Alferes. Well founded semantics for logic programs with explicit negation. In *B. Neumann (Ed.), European Conference on Artificial Intelligence*, pages 102–106. Wiley, 1992.

[17] Henry Prakken and Giovanni Sartor. Argument-based extended logic programming with defeasible priorities. *Journal of Applied Non-Classical Logics*, 7(1), 1997.

[18] Michael Schroeder. An efficient argumentation framework for negotiating autonomous agents. In *Proceedings of Modelling Autonomous Agents in a Multi-Agent World MAAMAW99.* LNAI1647, Springer-Verlag, 1999.

[19] A. Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics*, 5:285–309, 1955.

[20] Gerard A. W. Vreeswijk. Abstract argumentation systems. *Artificial Intelligence*, 90(1–2):225–279, 1997.

[21] Gerd Wagner. *Vivid Logic – Knowledge-Based Reasoning with Two Kinds of Negation*, volume LNAI 764. Springer–Verlag, 1994.

Appendix

Proof of Proposition 10

1. "If"-direction: Induction on the length n of the derivation of $L \in \Gamma(I)$.

   Base case: $n = 1$:
   Then there exists a rule $L \leftarrow not\ L_1, \ldots, not\ L_n$ in P s.t. $L_1, \ldots, L_n \notin I$, and $[L \leftarrow not\ L_1, \ldots, not\ L_n]$ is an argument for L whose assumptions are contained in I.

   Induction step: $n \rightsquigarrow n+1$:
   Let $L \in \Gamma^{n+1}(I)$. Then there exists a rule $r = L \leftarrow L_1, \ldots, L_n, not\ L'_1, \ldots, L'_m$ in P s.t. $L_i \in \Gamma^n(I)$, and $L'_i \notin I$. By induction hypothesis, there exists arguments $A_1, \ldots, A_n$ for $L_1, \ldots, L_n$ with $assm(A_i) \subseteq I$. Then $A = [r] \cdot A_1 \cdots A_n$ is an argument for L such that $assm(A) \subseteq I$.

   "Only-if" direction: Induction on the length of the argument.

   Base case: $n = 1$:
   Then $A = [L \leftarrow not\ L_1, \ldots, not\ L_n]$, and $L_1, \ldots, L_n \notin I$. Then $L \leftarrow\in \frac{P}{I}$, and $L \in \Gamma^1(I)$.

   Induction step: $n \rightsquigarrow n+1$:
   Let $A = [L \leftarrow L_1, \ldots, L_n, not\ L'_1, \ldots, not\ L'_m; r_2, \ldots, r_n]$ be an argument s.t. $assm(A) \subseteq I$. A contains subarguments $A_1, \ldots, A_n$ for $L_1, \ldots, L_n$, with $assm(A_i) \subseteq I$. Because $L'_1, \ldots, L'_m \notin I$, then $L \leftarrow L_1, \ldots, L_n \in \frac{P}{I}$. By induction hypothesis, $L_i \in \Gamma(I)$. so also $L \in \Gamma(I)$.

2. "If"-direction: Induction on the length n of the derivation of $L \in \Gamma_s(I)$.

   Base case: $n = 1$:
   Then there exists a rule $L \leftarrow not\ L_1, \ldots, not\ L_n$ in P s.t. $\neg L, L_1, \ldots, L_n \notin I$, and $[L \leftarrow not\ L_1, \ldots, not\ L_n]$ is an argument for L whose assumptions are contained in I, and $\neg L \notin I$.

   Induction step: $n \rightsquigarrow n+1$:
   Let $L \in \Gamma^{n+1}(I)$. Then there exists a rule $r = L \leftarrow L_1, \ldots, L_n, not\ L'_1, \ldots, L'm$ in P s.t. $L_i \in \Gamma^n(I)$, $L'_i \notin I$, and $\neg L \notin I$. By induction hypothesis, there exists arguments $A_1, \ldots, A_n$ for $L_1, \ldots, L_n$ with $assm(A_i) \subseteq I$ and $\neg conc(A_i) \cap I = \emptyset$. Then $A = [r] \cdot A_1 \cdots A_n$ is an argument for L such that $assm(A) \subseteq I$, and $\neg conc(A) \cap I = \emptyset$.

   "Only-if" direction: Induction on the length of the argument.

   Base case: $n = 1$:
   Then $A = [L \leftarrow not\ L_1, \ldots, not\ L_n]$, and $\neg L, L_1, \ldots, L_n \notin I$. Then $L \leftarrow\in \frac{P_s}{I}$, and $L \in \Gamma^1(I)$.

   Induction step: $n \rightsquigarrow n+1$:
   Let $A = [L \leftarrow L_1, \ldots, L_n, not\ L'_1, \ldots, not\ L'_m; \ldots]$ be an argument s.t. $assm(A) \subseteq I$, and $\neg conc(A) \cap I = \emptyset$. A contains subarguments $A_1, \ldots, A_n$ for $L_1, \ldots, L_n$, with $assm(A_i) \subseteq I$, and $\neg conc(A_i) \cap I = \emptyset$. Because $L'_1, \ldots, L'_m \notin I$, and $\neg L \notin I$, then $L \leftarrow L_1, \ldots, L_n \in \frac{P}{I}$. By induction hypothesis, $L_i \in \Gamma(I)$. so also $L \in \Gamma(I)$.

3. and 4. follow immediately from 1. and 2., because I is two-valued.

□

Collective Argumentation

Alexander Bochman
Computer Science Department,
Holon Academic Institute of Technology, Israel
e-mail: bochmana@hait.ac.il

Abstract

An extension of an abstract argumentation framework, called collective argumentation, is introduced in which the attack relation is defined directly among sets of arguments. The extension turns out to be suitable, in particular, for representing semantics of disjunctive logic programs. Two special kinds of collective argumentation are considered in which the opponents can share their arguments.

Introduction

The general argumentation theory [Bondarenko *et al.*, 1997; Dung, 1995b] has proved to be a powerful framework for representing nonmonotonic formalisms in general, and semantics for normal logic programs, in particular. Thus, it has been shown that the main semantics for the latter, suggested in the literature, are naturally representable in this framework (see, e.g., [Dung, 1995a; Kakas and Toni, 1999]).

In this report we suggest a certain extension of the general argumentation theory in which the attack relation is defined directly among sets of arguments. In other words, we will permit situations in which a set of arguments 'collectively' attacks another set of arguments in a way that is not reducible to attacks among particular arguments from these sets. It turns out that this extension is suitable for providing semantics for disjunctive logic programs in which the rules have multiple heads. In addition, it suggests a natural setting for studying kinds of argumentation in which the opponents could provisionally share their arguments. Moreover, the original argumentation theory can be 'reconstructed' in this framework by requiring, in addition, that the attack relation should be *local* in the sense that a set of arguments can attack another set of arguments only if it attacks a particular argument in this set.

The plan of the paper is as follows. After a brief description of the abstract argumentation theory, we suggest its generalization in which the attack relation is defined on sets of arguments. It is shown that the suggested collective argumentation theory is adequate for representing practically any semantics for disjunctive logic programs. As an application of the general theory, we consider two special cases of the general framework in which the opponents can share their arguments. The semantics obtained in this way will correspond to some familiar proposals, given in the literature.

1 Abstract Argumentation Theory

We give first a brief description of the argumentation theory from [Dung, 1995b].

Definition 1.1. An *abstract argumentation theory* is a pair $\langle \mathcal{A}, \hookrightarrow \rangle$, where $\mathcal{A}$ is a set of *arguments*, while $\hookrightarrow$ a binary relation of an *attack* on $\mathcal{A}$.

A general task of the argumentation theory consists in determining 'good' sets of arguments that are safe in some sense with respect to the attack relation. To this end, we should extend first the attack relation to sets of arguments: if Γ, Δ are sets of arguments, then $\Gamma \hookrightarrow \Delta$ is taken to hold iff $\alpha \hookrightarrow \beta$, for some $\alpha \in \Gamma$, $\beta \in \Delta$.

An argument α will be called *allowable* for the set of arguments Γ, if Γ does not attack α. For any set of arguments Γ, we will denote by $[\Gamma]$ the set of all arguments allowable by Γ, that is

$$[\Gamma] = \{\alpha \mid \Gamma \not\hookrightarrow \alpha\}$$

An argument α will be called *acceptable* for the set of

arguments Γ, if Γ attacks any argument against α. As can be easily checked, the set of arguments that are acceptable for Γ coincides with $[[\Gamma]]$.

Using the above notions, we can give a quite simple characterization of the basic objects of an abstract argumentation theory.

Definition 1.2. A set of arguments Γ will be called

- *conflict-free* if $\Gamma \subseteq [\Gamma]$;
- *admissible* if it is conflict-free and $\Gamma \subseteq [[\Gamma]]$;
- a *complete extension* if it is conflict-free and $\Gamma = [[\Gamma]]$;
- a *preferred extension* if it is a maximal complete extension;
- a *stable extension* if $\Gamma = [\Gamma]$.

A set of arguments Γ is conflict-free if it does not attack itself. A conflict-free set Γ is admissible iff any argument from Γ is also acceptable for Γ, and it is a complete extension if it coincides with the set of arguments that are acceptable with respect to it. Finally, a stable extension is a conflict-free set of arguments that attacks any argument outside it. Clearly, any stable extension is also a preferred extension, any preferred extension is a complete extension, and any complete extension is an admissible set. Moreover, as has been shown in [Dung, 1995b], any admissible set is included in some complete extension. Consequently, preferred extensions coincide with maximal admissible sets. In addition, the set of complete extensions forms a complete lower semi-lattice: for any set of complete extensions, there exists a unique greatest complete extension that is included in all of them. In particular, there always exists a least complete extension of an argumentation theory.

As has been shown in [Dung, 1995a], under a suitable translation, the above objects correspond to well-known semantics suggested for normal logic programs. Thus, stable extensions correspond to stable models (answer sets), complete extensions correspond to partial stable models, preferred extensions correspond to regular models, while the least complete extension corresponds in this sense to the well-founded semantics (WFS). These results have shown, in effect, that the abstract argumentation theory successfully captures the essence of logical reasoning behind normal logic programs.

Unfortunately, the above argumentation theory cannot be extended directly to disjunctive logic programs. The reasons for this shortcoming, as well as a way of modifying the argumentation theory are discussed in the next section.

2 Collective Argumentation

We begin with pointing out a peculiar discrepancy between the abstract argumentation theory, on the one hand, and the general abductive framework used for interpreting semantics for logic programs, on the other hand (see, e.g., [Bondarenko *et al.*, 1997; Dung, 1995a; Kakas and Toni, 1999]). The main objects of the abductive argumentation theory are sets of assumptions (abducibles) of the form $\mathbf{not}\, p$ that play the role of arguments in the associated argumentation theory. In addition, the attack relation is defined in this framework as a relation between sets of abducibles and particular abducibles they attack. For example, the program rule $r \leftarrow \mathbf{not}\, p, \mathbf{not}\, q$ is interpreted as saying that the set of assumptions $\{\mathbf{not}\, p, \mathbf{not}\, q\}$ attacks the assumption $\mathbf{not}\, r$.

The above attack relation is employed for defining the basic objects (such as extensions) of the source abductive framework. The abstract argumentation theory defines its main objects, however, as *sets* of arguments. Consequently, they should correspond to *sets of sets* of assumptions in the abductive framework. The abductive theory defines such objects, however, as certain plain sets of assumptions! In other words, we have a certain discrepancy between the levels of representations of intended objects in these two theories.

The above discrepancy will disappear once we notice that all the basic objects of the abstract argumentation theory are definable, in effect, in terms of the derived attack relation $\Gamma \hookrightarrow \alpha$ between sets of arguments and particular arguments; only the latter was used in defining the above operator $[\Gamma]$. As a result, the abductive argumentation theory can be constructed in the same way as the abstract theory, with the only distinction that the attack relation between sets of arguments and particular arguments is not reducible to the attack relation among individual arguments.

The above construction of abductive argumentation naturally suggests that assumptions, or abducibles, can be considered as full-fledged arguments, while the attack relation is best describable as a relation among sets of arguments. Indeed, once we allow for a possibility that a set of arguments can produce a nontrivial attack that is not reducible to attacks among particular arguments, it is only natural to allow also for a possibility that a set of arguments could be attacked in such a way that we cannot single out a particular argument in the attacked set that could be blamed

for it. In a quite common case, for example, we can disprove some conclusion jointly supported by the disputed set of arguments. The following generalization of the abstract argumentation framework reflects this idea.

Definition 2.1. A *collective argumentation theory* is a pair $\langle \mathcal{A}, \hookrightarrow \rangle$, where $\mathcal{A}$ is a set of arguments, and $\hookrightarrow$ is an attack relation on finite subsets of $\mathcal{A}$ satisfying the following *monotonicity condition*:

(Monotonicity) If $\Gamma \hookrightarrow \Delta$, then $\Gamma \cup \Gamma' \hookrightarrow \Delta \cup \Delta'$.

Though the attack relation is defined above only on finite sets of arguments, it can be naturally extended to arbitrary such sets by imposing the following *compactness property*:

(Compactness) $\Gamma \hookrightarrow \Delta$ only if $\Gamma' \hookrightarrow \Delta'$, for some finite $\Gamma' \subseteq \Gamma$, $\Delta' \subseteq \Delta$.

As the reader may notice, the suggested argumentation framework has many properties in common with ordinary sequent calculus, or consequence relations. Moreover, we will use in what follows the same agreements for the attack relation as that commonly accepted for consequence relations. Thus, $\Gamma, \Gamma_1 \hookrightarrow \Delta, \Delta_1$ will have the same meaning as $\Gamma \cup \Gamma_1 \hookrightarrow \Delta \cup \Delta_1$. Similarly, $\alpha, \Gamma \hookrightarrow \Delta, \beta$ will be an alternative notation for $\{\alpha\} \cup \Gamma \hookrightarrow \{\beta\} \cup \Delta$, etc.

The argumentation theory from [Dung, 1995a] satisfies all the above properties. Moreover, the above modification of the abstract argumentation theory has already been suggested, in effect, in [Kakas and Toni, 1999]. However, the attack relation defined in the latter paper satisfied also a couple of further properties described in the following definition.

Definition 2.2. A collective argumentation theory will be called

- *affirmative* if no set of arguments attacks the empty set $\emptyset$;
- *local* if it satisfies the following condition:

  (Locality) If $\Gamma \hookrightarrow \Delta, \Delta'$, then either $\Gamma \hookrightarrow \Delta$ or $\Gamma \hookrightarrow \Delta'$.

- *normal* if it is both affirmative and local.

If a collective argumentation theory is normal, then it can be easily shown that $\Gamma \hookrightarrow \Delta$ will hold if and only if $\Gamma \hookrightarrow \alpha$, for some $\alpha \in \Delta$. Consequently, the attack relation in such argumentation theories is reducible to the relation $\Gamma \hookrightarrow \alpha$ between sets of arguments and single arguments, and the resulting argumentation theory will coincide, in effect, with that given already in [Dung, 1995a].

It turns out, however, that the general, non-local framework of collective argumentation is precisely what is needed in order to represent semantics of disjunctive logic programs.

2.1 Collective Argumentation and Disjunctive Programs

Despite an obvious success, the abstract argumentation theory is still not abundant with intuitions and principles that could guide its development independently of applications. In this respect, logic programming and its semantics constitute one of the crucial sources and driving forces behind development of argumentation theories. Consequently, as a first step in studying collective argumentation, we consider its representation capabilities in describing semantics for disjunctive logic programs.

In what follows, given a set of propositional atoms C, we will denote by $\overline{C}$ the complement of C in the set of all atoms. In addition, $\mathbf{not}\, C$ will denote the set of all negative literals (abducibles) $\mathbf{not}\, p$, for $p \in C$.

By the general correspondence between normal logic programs and abductive argumentation frameworks, a set of abducibles $\mathbf{not}\, C$ attacks an abducible $\mathbf{not}\, p$ in the abductive theory associated with a normal logic program P if P, taken together with $\mathbf{not}\, C$ as a set of additional assumptions, allows to derive p.

The above description immediately suggests a generalization according to which any disjunctive logic program P determines an attack relation among sets of abducibles as follows:

$$\mathbf{not}\, C \text{ attacks } \mathbf{not}\, D \text{ iff } P \cup \mathbf{not}\, C \text{ derives } \bigvee D.$$

As can be easily verified, the above defined attack relation satisfies all the properties of collective argumentation. However, it is in general not local: $P \cup \mathbf{not}\, C$ may support $p \vee q$ without supporting either p or q. Still, it will be affirmative for disjunctive logic programs without constraints.

The appropriateness of the original argumentation theory for representing semantics of normal logic programs was based, ultimately, on the fact that these semantics are completely determined by rules of the form $p \leftarrow \mathbf{not}\, C$ that are derivable from a program. A similar principle, called *the principle of partial deduction, or evaluation* is valid also for the majority of se-

mantics suggested for disjunctive logic programs. According to this principle, semantics of such programs should be completely determined by rules $C \leftarrow \mathbf{not}\, D$ without positive atoms in bodies that are derivable from the source program. See also [Bochman, 1998] for the role of this principle in determining semantics of logic programs of a most general kind.

The above considerations indicate that practically all 'respectable' semantics for disjunctive programs should be expressible in terms of collective argumentation theories associated with such programs.

It turns out, however, that the actual semantics suggested for disjunctive programs do not fit easily into the general constructions of Dung's argumentation theory. A most immediate reason for this is that the operator $[\Gamma]$ of the abstract argumentation theory is no longer suitable for capturing the main content of the collective attack relation, since the latter is defined as holding between sets of arguments. Accordingly, it seems reasonable to generalize it to an operator that outputs a set of sets of arguments:

$$\langle\Gamma\rangle = \{\Delta \mid \Gamma \not\rightsquigarrow \Delta\}$$

As in the abstract argumentation theory, $\langle\Gamma\rangle$ will collect argument sets that are *allowable* with respect to Γ. Notice that, due to monotonicity of the attack relation, $\langle\Gamma\rangle$ will be closed with respect to subsets, that is, if $\Delta \in \langle\Gamma\rangle$ and $\Phi \subseteq \Delta$, then $\Phi \in \langle\Gamma\rangle$. Consequently, any set $\langle\Gamma\rangle$ will be completely determined by maximal argument sets belonging to it. As can be easily verified, such maximal sets will always exist due to compactness of the attack relation.

2.2 Stable and Partial Stable Argument Sets

Using the above generalized operator of allowability, we can give a rather simple description of stable and partial stable models for disjunctive programs (see, e.g., [Gelfond and Lifschitz, 1991; Przymusinski, 1991]) in terms of collective argumentation.

Definition 2.3.
- A set of arguments Γ will be said to be *stable* with respect to a collective argumentation theory if it is a maximal set in $\langle\Gamma\rangle$.
- A pair of sets (Γ, Δ) will be called *p-stable* if $\Gamma \subseteq \Delta$, Γ is a maximal set in $\langle\Delta\rangle$, and Δ is a maximal set in $\langle\Gamma\rangle$.

The following lemmas give more direct, and often more convenient, descriptions of the above objects. The proofs are immediate, so we omit them.

Lemma 2.1. *Γ is a stable set iff* $\Gamma = \{\alpha \mid \Gamma \not\rightsquigarrow \Gamma, \alpha\}$.

The above equation says that a stable set is a set Γ consisting of all arguments α such that Γ does not attack $\Gamma \cup \{\alpha\}$. A similar description can be given for partial stable sets:

Lemma 2.2. *(Γ, Δ) is p-stable iff* $\Gamma \subseteq \Delta$, $\Delta = \{\alpha \mid \Gamma \not\rightsquigarrow \Delta, \alpha\}$, *and* $\Gamma = \{\alpha \mid \Delta \not\rightsquigarrow \Gamma, \alpha\}$.

Recall that normal collective argumentation theories could be identified with abstract Dung's argumentation theories. Moreover, the above descriptions can be used to show that if a collective argumentation theory is normal, then stable argument sets will coincide with stable extensions, while p-stable pairs will correspond exactly to complete extensions of the abstract argumentation theory. These facts could also be obtained as a by-product of the correspondence between such objects and relevant semantics of disjunctive programs stated below.

The correspondence between the above descriptions and (partial) stable models of disjunctive logic programs is established in the following theorem.

Theorem 2.3. *If $\mathcal{A}_P$ is a collective argumentation theory corresponding to a disjunctive program P, then*

- *C is a stable model of P iff* $\mathbf{not}\, \overline{C}$ *is a stable set in $\mathcal{A}_P$.*
- *(C, D) is a p-stable model of P iff* $(\mathbf{not}\, \overline{C}, \mathbf{not}\, \overline{D})$ *is p-stable in $\mathcal{A}_P$.*

P-stable models have been introduced in [Bochman, 1998] as a slight modification of Przymusinski's partial stable models for disjunctive programs from [Przymusinski, 1991]; the reason for the modification was that the original Przymusinski's semantics violated the above-mentioned principle of partial deduction. In our present context, this means that it is not definable directly in terms of the collective argumentation theory associated with a disjunctive program. Note, however, that the modification does not change the correspondence with partial stable models for normal logic programs.

The above results could serve as an instance of our earlier claim that semantics of disjunctive programs are representable in the framework of collective argumentation. These results reveal, however, that the relevant objects are significantly different from the corresponding objects of the abstract argumentation theory. In order to get a further insight on the differences, we will consider now various notions of admissibility for argument sets that are definable in the framework of collective argumentation.

2.3 Argument sharing

In ordinary disputation and argumentation the parties can provisionally accept some of the arguments defended by their adversaries in order to disprove the latter. Two basic cases of such an 'argument sharing' in attacking the opponents are described in the following definition (see also [Bondarenko *et al.*, 1997]).

Definition 2.4. • Γ *positively attacks* Δ (notation $\Gamma \hookrightarrow^{+} \Delta$) if $\Gamma, \Delta \hookrightarrow \Delta$;

• Γ *negatively attacks* Δ (notation $\Gamma \hookrightarrow^{-} \Delta$) if $\Gamma \hookrightarrow \Gamma, \Delta$.

In a positive attack, the proponent temporarily accepts opponent's arguments in order to disprove the latter, while in a negative attack she shows that her arguments are sufficient for challenging an addition of opponent's arguments. Clearly, if Γ attacks Δ directly, then it attacks the latter both positively and negatively. The reverse implications do not hold, however.

Note that the above defined notion of a stable argument set was formulated, in effect, in terms of negative attacks. Indeed, it is easy to see that a set Γ is stable iff it negatively attacks any argument outside it:

$$\Gamma = \{\alpha \mid \Gamma \not\hookrightarrow^{-} \alpha\}$$

As can be seen, the above definition is equivalent to the definition of stable extensions in the abstract argumentation theory, given earlier, so stable extensions and stable sets of collective argumentation are indeed close relatives.

Recall now that admissible argument sets in Dung's argumentation theory are definable as conflict-free sets that counterattack any argument against them. Given the above proliferation of the notion of an attack in collective argumentation, however, we can obtain a number of possible definitions of admissibility by allowing different kinds of attack and/or counterattack among sets of arguments. Three such notions turns out to be of special interest.

Definition 2.5. A conflict-free set of arguments Γ will be called

• *admissible* if $\Gamma \hookrightarrow \Delta$ whenever $\Delta \hookrightarrow \Gamma$;

• *positively admissible* if $\Gamma \hookrightarrow^{+} \Delta$ whenever $\Delta \hookrightarrow^{+} \Gamma$;

• *negatively admissible* if $\Gamma \hookrightarrow^{-} \Delta$ whenever $\Delta \hookrightarrow^{-} \Gamma$.

Plain admissibility is a direct counterpart of the corresponding notion from the abstract argumentation theory. Unfortunately, in the context of collective argumentation it lacks practically all the properties it has in the latter. Notice, in particular, that stable sets as defined above need not be admissible in this sense.

As can be seen, positive and negative admissibility coincide with plain admissibility for respective 'extended' attack relations. The latter have some specific features that make the overall structure simpler and more regular. They will be described in the following sections.

3 Negative Argumentation

The definition below provides a general description of collective argumentation based on a negative attack. Such argumentation theories will be shown to be especially suitable for studying stable argument sets.

Definition 3.1. A collective argumentation theory will be called *negative* if $\Gamma \hookrightarrow \Gamma, \Delta$ always implies $\Gamma \hookrightarrow \Delta$.

As can be easily verified, any collective argumentation theory will be negative with respect to the negative attack relation $\hookrightarrow^{-}$. Moreover, the latter determines a least negative 'closure' of the source attack relation.

The following result gives an important alternative characterization of negative argumentation; it establishes a correspondence between negative argumentation and *shift* operations studied in a number of papers on disjunctive logic programming [Dix *et al.*, 1994; Schaerf, 1995; You *et al.*, 2000].

Lemma 3.1. *An argumentation theory is negative iff it satisfies:*

(Importation) *If* $\Gamma \hookrightarrow \Delta, \Phi$, *then* $\Gamma, \Delta \hookrightarrow \Phi$.

Proof. If the argumentation theory is negative and $\Gamma \hookrightarrow \Delta, \Phi$, then $\Gamma, \Delta \hookrightarrow \Gamma, \Delta, \Phi$ by monotonicity, and hence $\Gamma, \Delta \hookrightarrow \Phi$. The reverse implication is immediate. □

As an important special case of Importation, we have that if $\Gamma \hookrightarrow \Delta$, then $\Gamma, \Delta \hookrightarrow \emptyset$. Thus, any nontrivial negative argumentation theory is bound to be non-affirmative. Furthermore, this implies that self-contradictory arguments attack any argument:

$$\text{If } \Delta \hookrightarrow \Delta, \text{ then } \Delta \hookrightarrow \Gamma$$

These 'classical' properties indicate that negative attack is similar to a rule $A \vdash \neg B$ holding in a supra-classical consequence relation. Though the latter does

not admit contraposition, we nevertheless have that if $A \vdash \neg(B \wedge C)$, then $A, B \vdash \neg C$.

The connection between negative argumentation and stable argument sets is based on the following facts about general collective argumentation.

Theorem 3.2. *1. If Γ is negatively admissible, and Δ is a conflict-free set that includes Γ, then Δ is also negatively admissible.*

2. Stable sets coincide with maximal negatively admissible sets.

Proof. (1) Assume that Γ is negatively admissible, $\Gamma \subseteq \Delta$ and $\Phi \hookrightarrow^- \Delta$. Then $\Phi \hookrightarrow^- \Delta, \Gamma$, and hence $\Phi, \Delta \hookrightarrow^- \Gamma$ by Importation. Since Γ is negatively admissible, we obtain $\Gamma \hookrightarrow^- \Phi, \Delta$, and hence $\Gamma, \Delta \hookrightarrow^- \Phi$ by Importation. But the latter amounts to $\Delta \hookrightarrow^- \Phi$, which shows that Δ is also negatively admissible.

(2) It is easy to check that any stable set is negatively admissible. Moreover, any superset of a stable set will not already be conflict-free. Consequently stable sets will be maximal negatively admissible sets. In the other direction, if Γ is a maximal negatively admissible set and $\alpha \notin \Gamma$, then $\Gamma \cup \{\alpha\}$ will not be conflict-free by the previous claim, and hence $\Gamma, \alpha \hookrightarrow \Gamma, \alpha$. Consequently $\Gamma, \alpha \hookrightarrow^- \Gamma$, and therefore $\Gamma \hookrightarrow^- \Gamma, \alpha$ (since Γ is negatively admissible). But the latter implies $\Gamma \hookrightarrow^- \alpha$, which shows that Γ is actually a stable set. □

Recall now that negatively admissible sets are precisely admissible sets with respect to the negative attack $\hookrightarrow^-$. Moreover, in negative argumentation theories admissible sets will coincide with negatively admissible ones, while any conflict-free set will already be positively admissible. Accordingly, all nontrivial kinds of admissibility in such theories will boil down to (negative) admissibility; furthermore, maximal admissible sets in such argumentation theories will coincide with stable sets.

It can also be easily verified that any collective argumentation theory has the same stable sets as its negative closure. So, Importation is an admissible rule for argumentation systems based on stable sets. As a result, negative argumentation theories suggest themselves as a natural framework for describing stable sets.

Though the above results demonstrate that negatively admissible sets behave much like logically consistent sets, there is a crucial difference: the empty set $\emptyset$ is not, in general, negatively admissible. Moreover, an argumentation theory may be 'negatively inconsistent', that is, it may have no negatively admissible sets at all; this happens precisely when it has no stable sets.

Unfortunately, the above considerations indicate also that negative argumentation is inappropriate for studying argument sets beyond stable ones. Recall that one of the main incentives for introducing partial stable and well-founded models for normal programs was the desire to avoid taking stance on each and every literal and argument. However, negativity implies that self-contradictory arguments attack any argument whatsoever, so any admissible set is forced now to counter-attack any such argument. In particular, if $\Delta \hookrightarrow \Phi$, then any admissible set should attack $\Delta \cup \Phi$. This means that complete extensions (and partial stable models) are no longer a viable alternative for such argumentation systems. This means as well that Importation (and corresponding shift operations in logic programming) is an appropriate operation only for describing stable models[1].

4 Positive Argumentation

The following definition provides a description of argumentation based on a positive attack.

Definition 4.1. A collective argumentation theory will be called *positive* if $\Gamma, \Delta \hookrightarrow \Delta$ always implies $\Gamma \hookrightarrow \Delta$.

Any collective argumentation theory will be positive with respect to the positive attack relation $\hookrightarrow^+$. Moreover, the latter determines a least positive extension of the source attack relation.

Example. Consider an argumentation theory containing only two arguments α and β such that $\alpha \hookrightarrow \alpha$ and $\alpha \hookrightarrow \beta$. As can be seen, this argumentation theory has no extensions, while the corresponding positive theory has a unique extension $\{\beta\}$.

Similarly to negative argumentation, positive argumentation can be characterized by the 'exportation' property described in the lemma below:

Lemma 4.1. *An argumentation theory is positive iff it satisfies:*

(Exportation) *If $\Gamma, \Delta \hookrightarrow \Phi$, then $\Gamma \hookrightarrow \Delta, \Phi$.*

The above characterization implies, in particular, that self-conflicting arguments are attacked by any argument:

$$\text{If } \Delta \hookrightarrow \Delta, \text{ then } \Gamma \hookrightarrow \Delta$$

So, in positive argumentation we are relieved, in effect, from the obligation to refute self-contradictory argu-

[1] despite some attempts made in this direction - see, e.g., [You *et al.*, 2000]. Actually, the same difficulty plagues attempts to define partial stable semantics for default logic.

ments. In particular, no allowable argument will be self-contradictory.

It is interesting to note that positive and negative argumentation are, in a sense, incompatible on pain of trivialization. Namely, if we combine positive and negative argumentation, we obtain a symmetric attack relation:

Lemma 4.2. *If an argumentation theory is both positive an negative, then $\alpha \hookrightarrow \beta$ always implies $\beta \hookrightarrow \alpha$.*

Proof. If $\alpha \hookrightarrow \beta$, then $\alpha, \beta \hookrightarrow \alpha, \beta$ by monotonicity. Consequently, $\alpha, \beta \hookrightarrow \alpha$ by negativity and hence $\beta \hookrightarrow \alpha$ by positivity. □

In this case we can consider the attack relation $\alpha \hookrightarrow \beta$ as expressing plain *incompatibility* of the arguments α and β in a classical logical sense. In other words, we could treat arguments as propositions and define $\alpha \hookrightarrow \beta$ as $\alpha \models \neg\beta$.

4.1 Local Positive Argumentation

As a matter of fact, positively admissible sets were introduced for normal programs in [Kakas and Mankarella, 1991] (see also [Kakas and Toni, 1999]) under the name *weakly stable sets*; maximal such sets were termed *stable theories*. Accordingly, a study of such objects will amount to a study of admissible sets in collective argumentation theories that are positive closures of normal argumentation theories.

Note first that a positive closure of a local argumentation theory need not, in general, be local. Still, the following property provides a characterization of positive theories arising from local argumentation theories.

Definition 4.2. A collective argumentation theory will be called *l-positive* if it is positive and satisfies

(Semi-locality) If $\Gamma \hookrightarrow \Delta, \Phi$, then either $\Gamma, \Delta \hookrightarrow \Phi$ or $\Gamma, \Phi \hookrightarrow \Delta$

The following basic result shows that l-positive argumentation theories are precisely positive closures of local theories. Though the proof is not trivial, we omit it due to the lack of space.

Theorem 4.3. *An argumentation theory is l-positive iff it is a positive closure of some local argumentation theory.*

Due to the above result, stable theories from [Kakas and Mankarella, 1991] are exactly representable as maximal admissible sets in l-positive argumentation theories. It turns out that many properties of stable theories can be obtained in this abstract setting; the corresponding descriptions will be given in an extended version of this report. Still, we should mention that, since l-positive argumentation theories are not local, the corresponding structure of admissible sets is more complex than in the local case. For example, the set of admissible sets no longer forms a lower semilattice.

5 Preliminary Conclusions

Collective argumentation suggests itself as a natural extension of the abstract argumentation theory. It allows, in particular, to represent and study semantics for disjunctive logic programs. Speaking generally, it constitutes an argumentation framework in which the attack relation has structural properties allowing to represent cooperation and sharing of arguments among the parties.

The present report is very preliminary, however; it only barely scratches the surface of the vast number of problems and issues that arise in this setting. One of such issues consists in extending the approach to other, weaker semantics suggested for disjunctive programs. A more general task amounts, however, to determining general argumentation principles underlying collective argumentation. This is a subject of an ongoing research.

References

[Bochman, 1998] A. Bochman. A logical foundation for logic programming, I and II. *Journal of Logic Programming*, 35:151–194, 1998.

[Bondarenko *et al.*, 1997] A. Bondarenko, P. M. Dung, R. A. Kowalski, and F. Toni. An abstract, argumentation-theoretic framework for default reasoning. *Artificial Intelligence*, 93:63–101, 1997.

[Dix *et al.*, 1994] J. Dix, G. Gottlob, and V. Marek. Causal models for disjunctive logic programs. In P.Van Hentenryck, editor, *Proc. 11th Int. Conf. On Logic Programming, S. Margherita Ligure*, pages 290–302. MIT Press, 1994.

[Dung, 1995a] P. H. Dung. An argumentation-theoretic foundation for logic programming. *J. of Logic Programming*, 22:151–177, 1995.

[Dung, 1995b] P. M. Dung. On the acceptability of arguments and its fundamental role in non-monotonic reasoning, logic programming and n-persons games. *Artificial Intelligence*, 76:321–358, 1995.

[Gelfond and Lifschitz, 1991] M. Gelfond and V. Lifschitz. Classical negation and disjunctive databases. *New Generation Computing*, 9:365–385, 1991.

[Kakas and Mankarella, 1991] A. C. Kakas and P. Mankarella. Stable theories for logic programs. In V. Saraswat and K. Ueda, editors, *Proc. Int. Symp. On Logic Programming*, pages 85–100. MIT Press, 1991.

[Kakas and Toni, 1999] A. C. Kakas and F. Toni. Computing argumentation in logic programming. *J. of Logic and Computation*, 9:515–562, 1999.

[Przymusinski, 1991] T. C. Przymusinski. Stable semantics for disjunctive programs. *New Generation Computing*, 9:401–424, 1991.

[Schaerf, 1995] M. Schaerf. Negation and minimality in disjunctive databases. *J. of Logic Programming*, 23:63–86, 1995.

[You *et al.*, 2000] J.-H. You, L. Y. Yuan, and R. Goebel. An abductive apprach to disjunctive logic programming. *J. of Logic Programming*, 44:101–127, 2000.

Value-Based Argumentation Frameworks

Trevor Bench-Capon
Department of Computer
Science
The University of Liverpool,
Liverpool,
UK

Abstract

In this paper I introduce the notion of Value-Based Argumentation Frameworks (VAF). I start from a standard notion of argumentation frameworks which has been widely used to analyse logics for defeasible argument, and extend it so as to make defeat dependent on the relative importance of the values the arguments advance or protect. The idea is to re-introduce an element which has been abstracted away in standard argumentation frameworks, and which can be used to ground a rational choice between alternatives which are equally tenable from the more abstract point of view.

The key result of the paper is to show that in a VAF it may be possible to force rational acceptance of particular arguments within the VAF, irrespective of how the values are ranked. *Such arguments can be seen as objectively acceptable. In addition to the definition of VAFs, and exploration of their properties, particularly for the special, but rather common, case where there are precisely two values involved, the paper gives a detailed analysis of a well known moral dispute in terms of a VAF, and presents some heuristics for extending a given VAF to change the status of particular arguments.*

1 INTRODUCTION

In many of the more interesting situations where humans are called upon to reason it seems that finding an argument is not enough. For there are often many arguments pertinent to the situation which conflict with and support one another in a variety of ways. Instances of this abound in the context of moral and legal argumentation. In recognition of this, in recent years there has being a growing interest in *logics for defeasible argumentation* (e.g. Dung 1995, Bonderenko et al 1997). Such logics allow us to analyse situations in which we have arguments that may be overturned by other arguments, to see what, if anything, may safely be concluded. In particular the notion of an *Argumentation Framework* (*AF*), introduced by Dung, allows us to explore a system of conflicting arguments, in which arguments are abstracted to entities whose role is solely determined by their relation to other arguments. In moral and legal debate, however, an argument may attack another argument without defeating it. Suppose two people are debating whether income tax should be raised. One argues that it should because to do so would promote social equality, while the other argues that it should not, since enterprise should be rewarded. Certainly these arguments conflict, but neither party is likely to withdraw his argument in the face of the other. The point is that one believes that it is more important to promote equality than to reward enterprise, and the other ranks these purposes differently. Note also that both may accept *both* arguments: the first party can accept that it is right to reward enterprise, *but not at the expense of equality*. The point is that in many contexts the soundness of an argument is not the only consideration: arguments also have a force which derives from the value they advance or protect. Thus an argument may be defended not only by defeating its attacker, but also by ranking its value more highly than that of its attacker. That moral and legal disagreement should be seen in terms of different preferences for the values which the conflicting arguments defend or promote is the major insight of work in jurisprudence such as that of Perelman (e.g. 1980). If we want to model this kind of debate, the notion of an argument found in *AFs* is too abstract: we need also be able to relate arguments to their supporting values, and to allow these values to be ranked to reflect the preferences of the audience to which the arguments are addressed.

Accordingly, in this paper I extend Dung's *AF* into a *Value-Based Argumentation Framework* (*VAF*), by allowing this information to be represented. *VAFs* are defined in Section 2. When we have this notion, we can see that *VAFs* have some different properties from

standard *AFs*, which can prove useful in the resolution of disputes. Important among these properties is that some arguments in a *VAF* will be in the preferred extension *independent of how the values are ranked.* These properties are discussed in Section 3. In Section 4 the use of a *VAF* is illustrated through an analysis of a well known problem from moral philosophy. Typically on entering a dispute a person will have some argument or arguments which they wish to be accepted. If the *VAF* does not produce this result, they will wish to extend it so that it does. Section 5 presents some strategic heuristics which will identify the kind of arguments with which they can usefully extend the *VAF*. Finally Section 6 offers some concluding remarks.

2 VALUE-BASED ARGUMENTATION FRAMEWORKS

Dung's definition of an Argumentation Framework (*AF*), given in Dung (1995), is:

Definition 1: An *argumentation framework* is a pair

$AF = \langle AR, attacks \rangle$

Where *AR* is a set of arguments and *attacks* is a binary relation on AR, i.e. $attacks \subseteq AR \times AR$. For two arguments *A* and *B*, the meaning of *attacks(A,B)* is that *A* represents an attack on *B*. We also say that a set of arguments *S* attacks an argument *B* if *B* is attacked by an argument in *S*.

In order to represent the values to which arguments relate and the ranking of the values we extend this definition to a Value-Based Argument Framework (*VAF*).

Definition 2: A *value-based argumentation framework* (*VAF*) is a 5-tuple:

VAF = <AR, attacks,V,val, valpref>

Where *AR*, and *attacks* are as for a standard argumentation framework, *V* is a non-empty set of values, *val* is a function which maps from elements of *AR* to elements of *V*, and *valpref* is a preference relation (transitive, irreflexive and asymmetric) $valpref \subseteq V \times V$. We say that an argument *A* relates to value *v* if accepting *A* promotes or defends *v*: the value in question is given by *val(A)*. For every $A \in AF$, $val(A) \in V$.

Our purpose in extending the *AF* was to allow us to distinguish between one argument attacking another, and that attack succeeding, so that the attacked argument is defeated. We therefore define the notion of *defeat:*

Definition 3: *An argument* $A \in AF$ *defeats an argument* $B \in AF$ *if and only if both attacks(A,B) and not valpref(val(B),val(A)).*

Note that an attack succeeds if both arguments relate to the same value, or if no preference between the values has been defined. If *V* contains a single value, the *VAF* becomes a standard *AF*. If each argument maps to a different value, we have a Preference Based Argument Framework (Amgoud and Cayroll 1998).

Dung (1997) introduces the important notions *acceptability, conflict free set, admissible set, preferred extension* and *stable extension* for *AFs*. We also need to define these notions for a *VAF*.

Definition 4: An argument $A \in AR$ is *acceptable* with respect to set of arguments *S, (acceptable(A,S))* if:

$(\forall x)((x \in AR \ \& \ defeats(x,A)) \rightarrow (\exists y)((y \in S) \ \& \ defeats(y,x)))$.

Definition 5: A set *S* of arguments is *conflict-free* if

$(\forall x)\ (\forall y)((\ x \in S \ \& \ y \in S) \rightarrow (\neg attacks(x,y) \vee valpref(val(y),val(x))))$.

Definition 6: A conflict-free set of arguments *S* is *admissible* if

$(\forall x)(x \in S \rightarrow acceptable(x,S))$.

Definition 7: A set of arguments *S* in a value-based argumentation framework *AF* is a *preferred extension* if it is a maximal (with respect to set inclusion) admissible set of *AR*.

Definition 8: A conflict-free set of arguments *S* is a *stable extension* if and only if *S* attacks each argument in *AR* which does not belong to *S*.

The significance of these notions is with reference to which arguments in the *VAF* it is possible consistently to accept. If an argument is acceptable with respect to *AR*, it cannot be defeated and thus must be accepted. Similarly if an argument is acceptable with respect to a subset *S* of *AR*, it is possible to accept it, provided *S* is accepted. *S* cannot, of course, be consistently accepted unless it is conflict free. Thus an admissible set is one that can be consistently accepted given a *VAF*. If, however, a admissible set can be further extended, it is possible to accept further arguments. Thus a preferred extension represents a set of acceptable arguments to which no more arguments can be added. For standard *AFs* it is known that an *AF* always has a preferred extension (although it may be the empty set), and that preferred extensions are not in general unique. This last point means that it is possible for two parties to a dispute to adopt different positions which cannot be resolved by the *AF*. A stable extension cannot be empty: hence not every preferred extension is stable, and an *AF* may have no stable extension. Stable extensions are also not, in general, unique. In section 3 we will consider if these properties also hold true of a *VAF*.

We can now relate these notions to the ideas of credulous and sceptical acceptance in non-monotonic

logic. An argument is *credulously acceptable* it appears in *at least one* preferred extension, and *sceptically acceptable* if it appears in *every* preferred extension.

Given a dispute, the ideal situation is that everyone concerned comes to accept precisely the same arguments. This requires that the preferred extension be unique. We say such a dispute is *resoluble*. Failing this ideal situation, we want to identify the consensus between the parties. This is represented by the sceptically acceptable arguments. If, however, an argument is not in this consensus, one of the parties may wish to show that it is at least defensible. For this to be so it must be credulously acceptable. We therefore say that an argument is *defensible* if it appears in at least one preferred extension, and *indefensible* otherwise. For a standard *AF* determining such things is not, in general easy. For example determining whether a dispute is resoluble is Co-NP complete, credulous acceptance is NP-complete, and sceptical acceptance even harder. See Dunne and Bench-Capon (2001) for a discussion of these and other similar results. The next Section will discuss how these properties are affected by moving to a VAF

3. PROPERTIES OF VAFS

A useful way to picture an *AF* is as a directed graph in which the arguments are vertices and the attacks are represented as edges, directed from attacker to attacked. For a *VAF* we may picture the nodes being coloured differently to represent different values. Most of the complexity problems with *AFs* arise from the possibility of cycles in such a graph. Consider first a cycle with an odd number of vertices, say a three cycle. Such a cycle will have an empty preferred extension: If we suppose a node to be in the preferred extension its successor will be defeated. But this means that the third argument will not be defeated, and this will then defeat the argument that we originally supposed to be in the preferred extension. If, on the other hand, we consider a cycle with an even number of vertices, we will get two disjoint preferred extensions. Let us now refer to cycles in a *VAF* as *monochromatic* if they contain arguments relating to a single value, *dichromatic* if they contain arguments relating to exactly two values, and *polychromatic* if they have two or more values.

In a *VAF*, monochromatic cycles of odd length (*odd-cycles*) will have the empty set as their preferred extension and monochromatic cycles of even length (*even cycles*) will have two preferred extensions, as in a standard *AF*. We should, however, feel somewhat uncomfortable with the existence of monochromatic cycles in our graph. An odd cycle will have the nature of a paradox, in that we cannot consistently accept anything, and an even cycle the nature of a dilemma, in that we must choose between two alternatives on no rational grounds. Note that this criticism does not apply to cycles in a standard *AF*: there we can reasonably construct cycles since the rational resolution of the dilemmas and paradoxes lies in the ability to prefer an argument on the basis of information not included in a standard *AF*.

If, however, we consider polychromatic cycles, we get a different picture. If the cycle contains arguments of two different values, call them *red* and *blue*, at some point in the cycle a red argument must have a blue attacker, and a blue argument must have a red attacker. Now either red is preferred to blue, or vice versa: hence either the red argument attacked by the blue argument or the blue argument attacked by the red argument is not defeated. But if this is so, the preferred extension is not empty. Having established that one argument is in the preferred extension, we can now proceed round the cycle and determine the status of the remaining arguments. Similar reasoning can be applied to any polychromatic cycle. The actual status of an argument will, of course, depend on which argument was first decided to be in, which depends on the order of the values. We can, however, say that *for a given order on values* there will be a unique preferred extension. Thus we can say:

Property 9: *Given an order on values, a polychromatic cycle in a VAF has a unique, non-empty preferred extension.*

But cycles were the source of non-empty and multiple preferred extensions in *AFs*. See Dunne and Bench-Capon 2001 for a proof that an *AF* with more than one preferred extension must have an even cycle. Therefore we can say that a *VAF* containing no monochromatic cycles will as a whole have a unique, non-empty preferred extension, given an order on values. We can offer the following algorithm as a means of calculating this extension.

Algorithm 10: *EXTEND(AF,attacks)*.

1) $S := \{s \in AR: (\forall y)(\text{not defeats}(y,s))\}$
2) $R := \{r \in AR: \exists s \in S$ *for which* $\text{defeats}(s,r)\}$
3) *If* $S = \varnothing$ *then return S and Halt*
4) $AR' := AR / (S \cup R)$
5) $\text{attacks}' := \text{attacks} / ((S \times R) \cup (R \times AR') \cup (AR' \times R))$
6) *Return* $S \cup EXTEND(AR', \text{attacks}')$

The correctness of this algorithm is shown in Bench-Capon and Dunne (2002).

Note now the following: we can determine whether a dispute represented as a *VAF* is resoluble, and, given an order on values, credulous and sceptical acceptance are

the same. The contents of the preferred extension, however, do depend on the value order. Now it is possible that some arguments may be in the preferred extension whatever the order on values and some in the preferred extension only for particular value orders. Let us call arguments in the preferred extension irrespective of value order *objectively acceptable*, those in the preferred extension for some value orders *subjectively acceptable*, and those in the preferred extension for no ordering on values *indefensible*. First, we may ask whether there are objectively acceptable arguments. Let us first consider the case of a dichromatic three-cycle. We may state that:

Property 11: *In a dichromatic three cycle the argument coloured differently from the other two will be objectively acceptable.*

Proof: First we define the notion of an *argument chain*.

Definition 12: An *argument chain* in a *VAF, C* is a set of n arguments $\{a_1 \ldots a_n\}$ such that:

i. $(\forall a)(\forall b)(a \in C \ \& \ b \in C) \rightarrow val(a) = val(b))$;
ii. a_1 *has no attacker in C;*
iii. *For all* $a_i \in C$ *if* $i > 1$, *then* a_i *is attacked and the sole attacker of* a_i *is* a_{i-1} .

In an argument chain C it is obvious that, since all attacks will succeed because all arguments have the same value, if a_1 is accepted, then *every odd* argument of C is also accepted and *every even* argument of C is defeated. Similarly if a_1 is defeated, every odd argument of C is defeated and every even argument of C is accepted.

A three cycle will consist of a chain of two arguments with one colour (say red) and a chain of one argument of the other colour (say blue). Suppose blue > red. Now the blue argument is attacked by a red argument and so undefeated. Suppose on the other hand that red > blue. Then the first argument in the red chain is undefeated and so the second argument is defeated. But this attacks the blue argument, which is therefore now defended by the first red argument and so undefeated. Thus the preferred extension contains the blue argument, irrespective of whether red > blue or blue > red.

As demonstrated in Bench-Capon and Dunne (2002) this result can be generalised for dichromatic odd cycles of any length to:

Property 13: *In a dichromatic odd cycle, the odd numbered arguments of any chain preceded by an even chain are objectively acceptable.*

Next consider even cycles. An even cycle must comprise either (1) an even number of odd chains, or (2) any number of even chains, or (3) a mixture of an even number of odd chains together with any number of even chains.

In case (1) the arguments in the preferred extension will depend on the ordering of the values. To be precise, it will comprise the odd numbered arguments from the chains with the preferred value and the even numbered arguments of the chains with the other value. This is observable from the fact that the first argument of the chain with the preferred value must be in, since the attack on it does not succeed. Hence all odd numbered arguments in that chain are also undefeated. In particular the last argument in the chain is not defeated, and so it defeats the first argument in the next chain, since its value is preferred. Hence the even numbered arguments in that chain will not be defeated.

In case (2) the preferred extension is independent of the ordering of the values, and will comprise the odd numbered arguments from each chain. The first argument of a chain with the preferred value will not be defeated. But this means that the last argument of that chain is defeated: hence the first argument of the succeeding chain is not defeated.

In case (3) the arguments in the preferred extension will depend on the value ordering, but the odd numbered arguments of any chain preceded by an even chain will be included irrespective of the value ordering. If the value of the odd chain is preferred, the first argument (and hence all odd numbered arguments) of this chain will be in the preferred extension. Suppose the value of the even chain is preferred. Now the first argument in this chain is not defeated since it must be attacked by the other, lesser, value. Hence the last argument will be defeated, since it is an even numbered member of a chain whose first argument is not defeated. Thus the first argument of the succeeding odd chain has a defence, and is not defeated.

So for even cycles we get:

Property 14: *In a dichromatic even cycle, the odd numbered arguments of any chain preceded by an even chain will be objectively acceptable.*

Note, however, that there may be no such argument, since the cycle may contain only odd chains. We may generalise these observations to:

Property 15: *In any dichromatic cycle, the odd numbered arguments of any chain preceded by an even chain will be objectively acceptable.*

But what of those chains preceded by odd chains? If the value of the preceding chain is preferred, the first argument will be defeated and the even numbered arguments will be in the preferred extension. If on the

other hand the value of the preceding chain is not preferred, then the odd numbered arguments of this chain will be in. We are now in a position to characterise fully the preferred extension of a dichromatic cycle:

Property 16: *The preferred extension of a dichromatic cycle comprises:*

(i) *the odd numbered arguments of all chains preceded by an even chain;*

(ii) *the odd numbered arguments of chains with the preferred value;*

(iii) *the even numbered arguments of all other chains.*

Note that those included under (i) are objectively acceptable and those included under (ii) and (iii) are subjectively acceptable. The even numbered arguments of a chain preceded by an even chain are indefensible.

Now let us view such dichromatic cycles as part of a fuller dichromatic *VAF*. Consider the *VAF* in Figure 1.

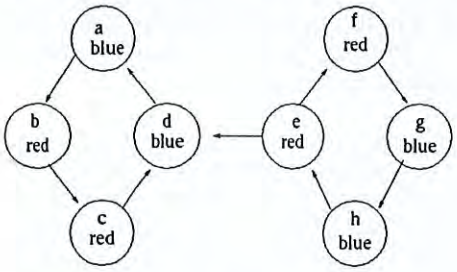


Figure 1: Example *VAF*

Now the blue argument *d* is the successor not only of the chain *bc*, but also of the one argument chain *e*. (Argument *e* is part of two chains, *e* and *ef*). There will be two preferred extensions, according to whether red > blue, or blue > red. If red > blue, the preferred extension will be {*e,g,a,b*}, and if blue > red, {*e,g,d,b*}. Now *e* and *g* and *b* are objectively acceptable, but *d*, which would have been objectively acceptable if *e* had not attacked *d*, is only subjectively acceptable, and *a*, which is indefensible if *d* is not attacked, is also subjectively acceptable Arguments *c* and *h* remain indefensible. So, to be objectively acceptable, an argument must be an odd numbered member of a chain preceded *only* by even chains. Suppose that instead of *e* attacking *d*, *h* attacked *d*. In this case *d* would be part of the even chain *ghdb*, preceded by the even chain *ef*. Now {*egdb*} is the only preferred extension, whatever the value ordering, since all chains are preceded only by even chains.

The fact that arguments can be part of two chains complicates matters. To try to impose some order let us consider some cases. Suppose that an argument is an even numbered member of a chain which is unattacked or preceded only by even chains. Will it then be indefensible? This depends on what we consider a chain. Consider Figure 2.

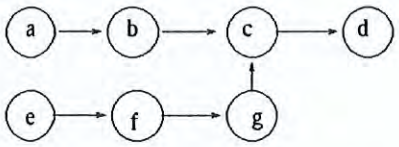


Figure 2: Monochromatic *VAF* with argument attacked twice

Suppose we see this as two chains, *abcd* and *efgcd*. Argument *d* is an even numbered member of the first. But in fact far from being indefensible, *d* is objectively acceptable, since *c* is always defeated because of its position in *efgcd*. In fact, we better construe this situation as three chains: *abc*, *d* and *efgc*. Now *d* is no longer an even numbered member of a chain attacked only by even chains, since it is now odd numbered. Thus an argument with more than one attacker needs to be seen as the last element of any chain where the attackers are of its own colour, and its successors will form their own new chain. If a node terminates two or more chains, this is understood, when considering chains attacked by it, as an even chain if any of its chains are even. An argument can also be part of two chains if it attacks two arguments with the same value: in this case, however, it will either be odd numbered or even numbered for all such chains.

So, an argument is indefensible if it is an even numbered member of any chain attacked only by even chains (construing an unattacked chain as attacked by a chain of length zero). But there is another way that an argument can be indefensible. Consider Figure 3.

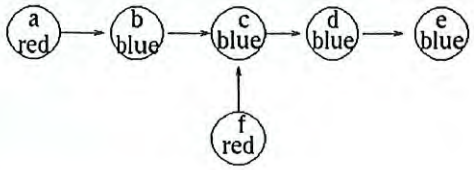


Figure 3: Dichromatic *VAF* with an argument attacked twice

Here we have the following chains: *a*, *bc*, *f*, *de*. If *f* was not present, we would have a single blue chain, *bcde*, and all the blue arguments would be subjectively acceptable: *b* and *d* if blue > red and *c* and *e* if red >

blue. Adding *f*, however, changes this, since it splits the blue chain into *bc* and *de*. Now *de* is a chain preceded only by an even chain, hence *d* is objectively acceptable and *e* is indefensible. Argument *c* will also be indefensible, since it will be defeated by *b* if blue > red and by *f* if red > blue. This is best seen as construing *c* as forming a fifth chain, of which it is the only element. Now an argument will be indefensible if it forms part of two chains attacked by odd chains in which it is even numbered in one chain and odd numbered in the other, which can happen only if an even numbered member of a chain is *directly* attacked by an odd chain. Again where an argument terminates two or more chains, it is considered to be the end of an even chain if any of these chains are even, for the purposes of attacking succeeding chains.

So if an argument has two (or more) attackers it will form part of two chains. If the attacking argument is of the same colour it will be the last element of that chain; if the attacker is of a different colour it will form a single element chain. If an argument is the member of any even chain, any chains it attacks will be considered attacked by an even chain. The objectively acceptable arguments will be the odd numbered members of chains preceded only by even chains.

We can summarise this :

Property 17: *In a dichromatic VAF:*

i. *an argument is indefensible if it an even numbered member of any chain preceded only by even chains; or if it an even numbered member of a chain attacked by an odd chain, and is directly attacked by an odd chain;*
ii. *an argument is objectively acceptable if it is only an odd numbered argument of a chain preceded only by even chains;*
iii. *an argument is subjectively acceptable otherwise.*

An unattacked argument is considered to be preceded by a chain of length zero, hence an even chain.

Dichromatic argument frameworks are of considerable importance since we can construe many disputes, such as that discussed in the next Section, as involving only two values. Some disputes, however, particularly in law, seem to involve more than two values. We should therefore ask if we can say anything about polychromatic *VAFs* in general. First we can say that a polychromatic *VAF* containing no monochromatic cycles will, given an ordering on values, have a unique preferred extension. Moreover the EXTEND algorithm remains applicable. We can, however, now construct examples of chains preceded by even chains in which the odd values are not objectively acceptable. Consider a seven cycle with three values, arranged as two blues, three reds, and two greens. Here the first and third red arguments will be objectively acceptable. But the blue chain also follows an even chain. The first argument in the blue chain will be defeated if green > red > blue (although undefeated for any other ordering). We now have a possible position in between objective and subjective acceptance: an argument may be acceptable with respect to a partial ordering on values, which means that parties can agree to accept some subjectively acceptable arguments even though they disagree as to the ranking of *some* values. Polychromatic *VAFs* give rise to a number of questions which require further work to explore. In the rest of the paper I will speak mainly of dichromatic *VAFs*.

4. AN EXAMPLE OF DISPUTES IN THE MORAL DOMAIN.

The scenario we will consider is taken from an example discussed in Coleman (1992), and further discussed in Christie (2000). Hal, a diabetic, loses his insulin in an accident through no fault of his own. Before collapsing into a coma he rushes to the house of Carla, another diabetic. She is not at home, but Hal enters her house and uses some of her insulin. Was Hal justified, and does Carla have a right to compensation? A diagrammatic representation of the *VAF* for this dispute is given as Figure 4.

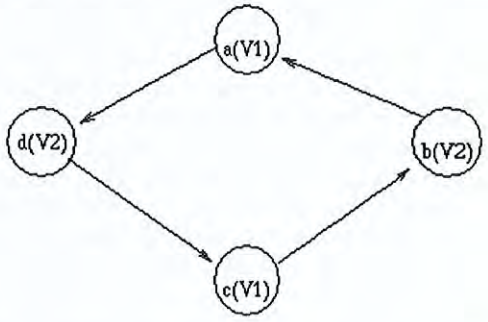


Figure 4: VAF for initial Hal and Carla scenario

The first argument is that Hal is justified, since a person has a privilege to use the property of others to save their life - the case of necessity. But should Hal compensate Carla? His justification can be attacked by an argument that if one infringes the property rights of another one should pay compensation. The first argument (a) is based on the value that life is important (V1), the second (b) on the value that property owners should be able to enjoy their property (V2). By valuing life over property we can accept both arguments: Hal was

justified in taking the insulin, but is obliged to compensate Carla. This appears to be Coleman's view. But now we may add a third argument, (c) attacking (b). This says that even if Hal were too poor to compensate Carla, then he should still be able to take the insulin: no one should die through their poverty. This in turn could be attacked by argument (d) which is based on the fact that starvation is not a recognised defence against theft, even of food. This argument is itself attacked by (a), so we have a four cycle with alternating values. In this case, as we have seen, we can choose either of two consistent positions, depending on which value we prefer. Either we can value human life over property, in which case we reject (b) and (d), and do not oblige Hal to compensate Carla, or we reverse our preference and have Hal in default if he cannot pay the compensation.

But suppose we argue the case differently. Instead of using argument (b) we attack (a) with an argument (e) to the effect that Hal is endangering Carla's life by consuming her insulin. On her return she may be in need of the insulin which is no longer available. Here both (e) and (a) are based on the same value, so (e) defeats (a). The threat to Carla can be removed by Hal replacing the insulin, so we can attack (e) by an argument that Hal should replace the insulin, effectively argument (b) above. Note that (b), which says that Hal is justified if and only if he compensates, can be used both to demand compensation and so attack (a) and to remove the threat to Carla, so attacking (e). We now plead Hal's poverty and attack (b) using (c). Now, however, instead of attacking (c) with a property based argument as above, suppose we use a life based argument, (f) to the effect that no one should preserve their own life at the cost of endangering another's. This argument is attacked by (a) which held that one may do anything to preserve one's own life. This framework now contains a five cycle (a,e,b,c,f,a) with a chain of four life based arguments, and one property based argument (b). In such a situation *property 13* shows that we have objectively to accept the property based argument, and conclude that Hal is not justified in taking the insulin without compensation - whichever value we prefer. Even if we rate life more highly than property, we must conclude that Hal should not value his own life over Carla's, and so it does seem fair that the first claim on the insulin should go to its rightful owner rather than anyone who can get possession of it.

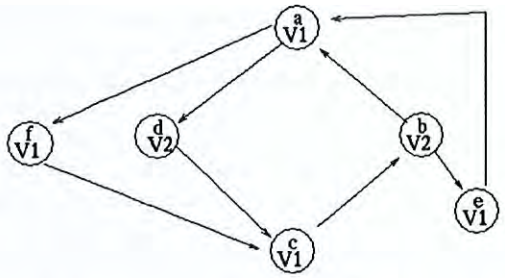


Figure 5: *VAF* for Hal and Carla case: V1 is life, V2 is property

The full *VAF* with all six arguments, as shown in Figure 5. Suppose V1 (life) is preferred to V2 (property). Now (e) is in the preferred extension because it is attacked only by (b) which has the inferior value. So (a) is out, because attacked by (e), so (f) and (d) are in, because attacked only by (a). Now (c) is defeated by (f), and (b), attacked only by (c), is in. Thus, given V1 > V2, the rational position is $\{e,f,d,b\}$. Now suppose V2 > V1: now (b) is in, because it is only attacked by the lower ranked (c). Therefore (a) and (e) are defeated by (b). With (a) out, (f) and (d) are in and (f) defeats (c). So, for V2 > V1, the rational position is $\{b,f,d\}$. Thus (b), (f) and (d) are objectively acceptable, and (e) is subjectively acceptable, if we value life over property. The conclusion is that Hal can take the insulin *only* if he replaces it before Carla needs it, hence only if he is able to replace it. Note that (a) and (c) are indefensible: thus it looks as if the poverty defence is a non-starter, as is the proposition that one may do anything (even endanger the life of an innocent person) to preserve one's life.

But this is so only because of the availability of (e) in the framework. Suppose in fact Carla was a pharmacist, and not a diabetic. Now Carla's life is not endangered by Hal's action, since Carla has no need of insulin. Removing (e) from the framework gives two even cycles. Now the choice of preferred value determines which of $\{a,c\}$ or $\{b,d,f\}$ is the rational position. Note that (b) is no longer objectively acceptable, so that Hal can take the insulin without compensation, given a preference for V1 over V2 Although (f) may be in general desired, it need not be included in the position of someone who prefers life over property, since without (e) it is irrelevant. With respect to (d), such a value order would seem to license theft by a starving person - provided the theft was not from another starving person. We may object to this and wish to analyse further.

The further analysis might run as follows. Suppose we attack (a) by an argument (g) saying that the life of

some other diabetic is endangered, since they may rely on Carla being in a position to supply them with insulin. We may attack this by a factual argument (h) to the effect that Carla is well stocked with insulin and will be able to meet any foreseeable demand. But we may again argue (k) that Hal cannot possibly know that this so: Carla may have allowed her stocks to run down. This gives the a *VAF* shown in Figure 6.

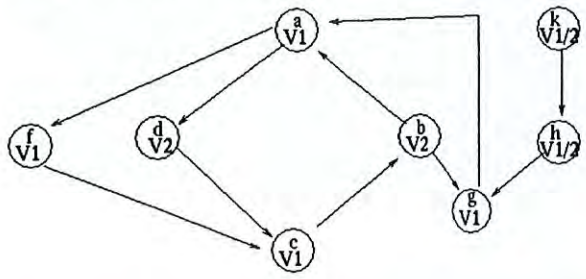


Figure 6: *VAF* for Hal and the extended chemist scenario

Note that a factual argument is taken as value free: it should therefore be capable of defeating any other argument irrespective of value. We model this within a dichromatic *VAF* by allowing it to take on the preferred value. Suppose here we allow it to take V1, since that will ensure that it always defeats (g). Now (k) is undefeated since it is a source. (h) is therefore defeated by (k). (a) will be defeated by (b) if V2 > V1 and by (g) if V1 > V2. Thus (f) and (d) are undefeated, and (c) is defeated by (f), leaving (b) undefeated. Thus all of (k), (b), (f) and (d) are objectively acceptable, and (g) is acceptable provided that V1 > V2. Arguments (a) and (c) are indefensible. This seems to be a reasonable resolution of the dispute: that the principle of necessity, that one may do anything to preserve one's own life, does not license endangering the lives of others.

5. STRATEGIC HEURISTICS IN *VAF*S

Given a *VAF* we can determine the status of an argument. But suppose we do not like the status so determined. We need not concede at this point: we can extend the *VAF* by adding another argument. But, of course, we want to extend the *VAF* at a point where it will have the impact we desire. The considerations above, suggest some heuristics for extending the *VAF* to produce a particular status for a given argument.

The status of an argument depends on its position in chains of which it is part, and on the parity of the chains attacking those chains. A decision graph showing the dependency of status on these factors is shown in Figure 7.

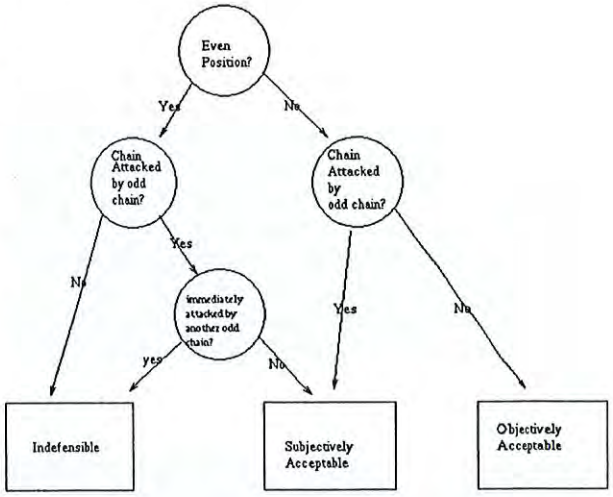


Figure 7: Decision process for status of an argument in a dichromatic *VAF*

First note that the availability of this decision process means that we need consider only the position of an argument within its chains, and the parity of chains attacking its chains. No attack on a successor argument, or any more remote argument need be considered.

To change the position of an argument within a chain we can either extend the chain (with an argument of the same colour), if the chain is unattacked; or attack some preceding *odd numbered* argument in the chain (with an argument of either colour). To make an argument part of an additional chain we must attack that argument itself (again with an argument of either colour). We can change the parity of an attacking chain either by extending that chain (with argument of the colour of that chain), or by attacking some odd numbered argument in that chain (with an argument of either colour). Finally if a chain is attacked only by even chains, we may prevent this by attacking the head of that chain with an argument of different colour.

If an argument is objectively acceptable it is an odd numbered member of a chain attacked only by even chains. To make it indefensible we must attack it, or an odd numbered predecessor in its chain, with an argument of the *same* colour. To make it subjectively acceptable we must change the parity of one of its attacking chains, by attacking an odd numbered argument of the attacking chain with an argument of *the*

colour of that chain, or adding a attacking chain by attacking itself, or an odd numbered predecessor in its chain with an argument of *different* colour.

If an argument is indefensible, it must be an even numbered member of a chain attacked only by even chains. To render it objectively acceptable we must attack an odd numbered predecessor in its chain with an argument of the *same* colour. It can be made subjectively acceptable by attacking an odd numbered argument of some attacking chain with an argument of *the colour of that chain*, or adding an attacker by attacking an odd numbered predecessor in its chain with an argument of *different* colour, giving its chain an odd attacker.

If an argument is subjectively acceptable, it must be part of a chain attacked by at least one odd chain. If even numbered it can be made objectively acceptable by attacking an odd numbered predecessor in its chain with an argument of the *same* colour. It can be made indefensible by making all the attackers of its chain of even length, by attacking some odd numbered member of each odd attacking chains with arguments of *their own colour*. Alternatively it can be made indefensible by attacking it, or an even numbered predecessor, with an argument of *either* colour. If it is odd numbered it can be made indefensible by attacking it, or an odd numbered predecessor with an argument of the *same* colour, and it can be made objectively acceptable by making all chains attacking its chain of even parity, by attacking odd numbered members of attacking odd chains with arguments of *their own* colour.

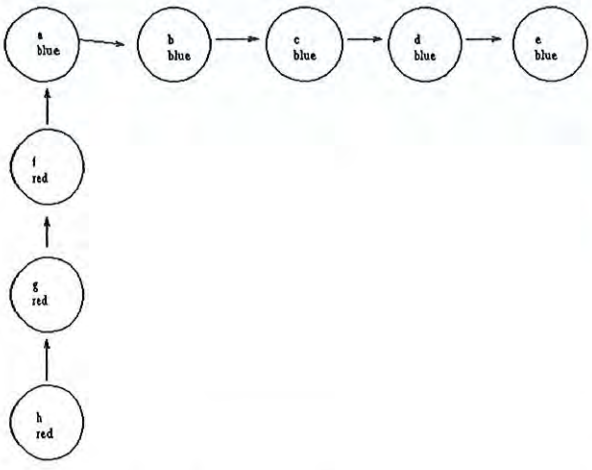


Figure 8: Example framework

For an example consider the VAF in Figure 8. Here we have a chain of five blue arguments attacked by a chain of three red arguments. All the blue arguments are subjectively acceptable. Suppose we wish to make (d), and even numbered member of the blue chain objectively acceptable. This could be achieved straightforwardly by a blue attack on its direct attacker (c), or its indirect attacker (a). To make it indefensible we could attack it, or its defender (b) with a blue argument. But we could also even the attacking chain by attacking (h), or (f) with a red argument. If we instead are interested in (e), we can make this indefensible by attacking it, or (a) or (c) with a blue argument. Equally we could make it objectively acceptable by attacking (b) or (d) with a blue argument, or (f) or (h) with a red argument.

With this information we can focus our attention on which arguments can be usefully attacked to produce the desired effect, and whether they need to be attacked by an argument of the same or different colour. These heuristics only consider the case where the *VAF* is extended by introducing a new argument attacking one other argument, which is probably the most usual way of continuing a dispute. Similar analysis could be used to provide heuristics if it is permitted to introduce a new attack between existing arguments, or to introduce several new arguments and attacks.

6. CONCLUDING REMARKS

In this paper I have introduced the notion of a Value-Based Argumentation Framework (*VAF*), an augmentation of standard Argumentation Frameworks, which represent an established way of exploring logics for defeasible reasoning. The key idea here is to allow for attacks to succeed or fail, depending on the relative worth of the values promoted by the competing arguments. *VAFs* have their basis in the important work of Perleman, who has persuasively argued that rationality need not be confined to demonstrable proofs, and that by recognising that arguments derive their force from promoting or protecting values shared by the audience of an argument, we may rationally choose amongst conflicting arguments, and that rational disagreement can be explained in terms of different priorities of the values associated with arguments. *VAFs* thus represent a way of adding this information to *AFs* so as to allow for this kind of explanation.

The main result of this paper is to show that in a *VAF* it may be possible to *force* rational acceptance of

particular arguments within the *VAF*, *irrespective of how the values are ranked.* The importance of this is that it demonstrates how some arguments are objectively compelling, even in the face of different subjective preferences.

These ideas have been illustrated by exploring the properties of *VAFs*, in particular for *VAFs* restricted to two values, which can represent an important class of moral and legal disputes. The analysis that is possible with a *VAF* has been illustrated by a discussion of a well known moral dispute. Finally I have shown how the properties of *VAFs* can be used to produce heuristics for extending a dispute in order to achieve some particular position.

References

Amgoud,L., and Cayrol, C., (1998). On the Acceptability of Arguments in Preference-Based Argumentation, in Cooper, G., and Moral, S., (eds), Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence.

Bench-Capon, T.J.M., and Dunne, P.E., (2002). *Value Based Argumentation Frameworks.* Research Report ULCS-02-001, Department of Computer Science, The University of Liverpool.

Bondarenko, A., Dung, P.M., Kowalski, R.A., and Toni, F., (1997). An abstract, argumentation-theoretic approach to default reasoning, *Artificial Intelligence*, 93, 63-101.

Christie, G.C., (2000). *The Notion of an Ideal Audience in Legal Argument*, Kluwer Academic Publishers.
Coleman, J., (1992). *Risks and Wrongs*. Cambridge University Press.

Dunne, P.E., and Bench-Capon, T.J.M., (2001). *Coherence in Finite Argument Systems.* Research Report ULCS-01-006, Department of Computer Science, The University of Liverpool.

Dung, P.H., (1995). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games, *Artificial Intelligence*, 77, 321-57.

Jacobovits, H., and Vermeir, D., (1999). Dialog Semantics for Argumentation Frameworks, Proceedings of the Seventh International Conference on AI and Law, ACM Press: New York, 53-62.

Perelman, Ch., (1980). *Justice, Law and Argument*, Reidel: Dordrecht.

Author Index

Amgoud, Leila, 399
Amir, Eyal, 41
Anger, Christian, 210
Anrig, Bernhard, 302
Antoniou, Grigoris, 84

Barcelo, Pablo, 160
Bench-Capon, Trevor, 444
Benferhat, Salem, 142
Bertossi, Leopoldo, 160
Berzati, Dritan, 302
Besnard, Philippe, 77
Bochman, Alexander, 135, 436
Bonatti, Piero A., 184
Brewka, Gerhard, 67
Broersen, Jan, 365

Cabalar, Pedro, 229
Calimeri, Francesco, 200
Castilho, Marcos A., 343
Cayrol, Claudette, 408
Chopra, Samir, 25, 117
Coletti, Giulianella, 275

Darwiche, Adnan, 6
Dastani, Mehdi, 33
Delgrande, James, 150
Denecker, Marc, 51, 183
Doutre, Sylvie, 408
Dubois, Didier, 103
Dunne, Paul E., 392

Faber, Wolfgang, 200
Fadel, Ronald, 356
Finzi, Alberto, 124
Fox, John, 293

Ghose, Aditya, 117
Gilis, David, 51
Giordano, Laura, 333
Gottlob, Georg, 4

Heidema, Johannes, 25
Herzig, Andreas, 343
Horty, John F., 391

Kakas, Antonis, 380
Kohlas, Jürg, 302
Konczak, Kathrin, 210
Konieczny, Sébastien, 171
Kyburg Jr., Henry E., 257

Lagasquie-Schiex, Marie-Christine, 408
Lagrue, Sylvain, 142
Leone, Nicola, 200
Lifschitz, Vladimir, 5
Linke, Thomas, 210
Lorenzo, David, 349
Lukasiewicz, Thomas, 265

Marek, Victor W., 219
Marquis, Pierre, 6
Maudet, Nicolas, 399
McBurney, Peter, 293
McIlraith, Sheila, 356
Mengin, Jérôme, 408
Mercer, Robert E., 77
Meyer, John-Jules, 365
Meyer, Thomas, 25, 117
Michael, Loizos, 380
Moinard, Yves, 15

Papini, Odile, 142
Pearce, David, 57
Pfeifer, Gerald, 200
Pino Pérez, Rámon, 171
Pirri, Fiora, 124
Prade, Henri, 103
Prakken, Henry, 91

Remmel, Jeffrey B., 219
Richter, Eva, 321

Sarsakov, Vladimir, 57
Satoh, Ken, 191
Schaub, Torsten, 57, 77, 238
Schroeder, Michael, 426
Schwarz, Sibylle, 247
Schweimeier, Ralf, 426
Schwind, Camilla, 333
Scozzafava, Romano, 275
Suderman, Matthew, 150

Teng, Choh Man, 257

Tompits, Hans, 57

van der Torre, Leendert, 33
Vantaggi, Barbara, 275
Varzinczak, Ivan J., 343
Verheij, Bart, 416

Wang, Kewen, 238
Weydert, Emil, 283
White, Graham, 372
Wieringa, Roel, 365
Williams, Mary-Anne, 3
Witte, René, 311
Woltran, Stefan, 57