Marrying Stable Models with Belief Update

Martin Slota[*] and **João Leite**[†]
CENTRIA & Departamento de Informática
Universidade Nova de Lisboa
Quinta da Torre
2829-516 Caparica, Portugal

Abstract

The need for integration of classical logic with nonmonotonic rules has been gaining importance in areas like the Semantic Web and Open Multi-Agent Systems. A number of researchers addressed this problem by proposing a unified semantics for knowledge bases composed of both first-order formulae and rules (also called *hybrid knowledge bases*). These semantics have matured over the years, but only provide solutions for the static case when knowledge does not need to evolve.

In this paper we take a first step towards addressing the dynamics of hybrid knowledge bases. Considering the results from the complementary areas of belief update and rule update, we focus on a scenario where rules can be used to describe the static background knowledge and reasoning of an agent, and the information about the current state of the world, encoded in classical logic, may evolve through time. We propose an update operator for this scenario and study its basic properties.

To the best of authors' knowledge, this is the first time results of both branches of research on updates are combined to design an update operator for a hybrid knowledge base.

1. Introduction

In this paper we define an update operator that is suitable for hybrid theories composed of both Logic Programming rules, used to represent the general knowledge or behaviour, and Classical (First-Order) Logic, used to represent evolving information about the state of a dynamic open world.

In open, distributed and heterogeneous environments, such as the one envisaged by many Semantic Web researchers, Classical Logic comes as a natural candidate for knowledge representation. It adopts the *open world assumption* (OWA), meaning that the knowledge base is, *by assumption*, potentially not complete, and, therefore, a proposition p is false only if the knowledge base is inconsistent with p. This suits well the open nature of such systems where complete knowledge about the environment cannot be assumed. Classical Logic has decidable as well as tractable fragments, such as the Description Logics (Baader et al. 2003), that are commonly used in practical applications.

Logic Programming, based on rules, provides a different set of features for knowledge representation, complementary to those offered by Classical Logic. It features formal, declarative and well-understood semantics, the stable model semantics (Gelfond and Lifschitz 1988) and its tractable approximation, the three-valued well-founded semantics (Gelder, Ross, and Schlipf 1991), being the most prominent and widely accepted. These semantics adopt the *closed world assumption* (CWA), meaning that the knowledge base *is assumed* to contain complete information. Consequently, a proposition p is considered false whenever it is not entailed to be true. This type of negation is usually dubbed *default negation* or *weak negation*, to distinguish it from the *classical negation* used in Classical Logic.

Incompleteness of information is also a cause for the need to reason with it under the CWA. In many situations it is desirable to be able to reason in the absence of a piece of information, rather than based on explicitly represented information only. Furthermore, even in an open environment, some parts of the knowledge base may be considered as completely known. Logic programs offer the necessary tools to reason using CWA and it is widely acknowledged that they can naturally express policies, preferences, norms and laws.

Since Classical Logic and Logic Programs both offer important features for knowledge representation in open systems, their unified use within a single knowledge base is of great interest. Their semantic integration, however, turned out to be a difficult task because OWA is largely incompatible with CWA. Many proposals for integrating the two formalisms have been proposed in the last decade (see (Hitzler and Parsia 2009) for a survey). One of the more mature formalisms are Hybrid MKNF Knowledge Bases (Motik and Rosati 2007) where decidability has been achieved through syntactic restrictions on the kinds of rules that are allowed. This semantics also has a tractable variant based on the well-founded semantics that allows for a top-down querying procedure (Alferes, Knorr, and Swift 2009), making the approach amenable to practical applications.

However, in many applications knowledge needs to evolve and the dynamics of hybrid knowledge bases cannot be addressed by the sole use of formalisms that only make it possible to represent and reason with static knowledge. The problems associated with knowledge evolution have been studied extensively by researchers in the field of

[*]Supported by FCT Scholarship SFRH/BD/38214/2007

[†]Partially supported by project DiFoS – LogICCC/0001/2007

belief change. The seminal work in this field is the paper by Alchourrón, Gärdenfors and Makinson (AGM) (Alchourrón, Gärdenfors, and Makinson 1985), which proposes a set of desirable properties for belief change operators, now called *AGM postulates*. Subsequently, in (Katsuno and Mendelzon 1991), *update* and *revision* have been distinguished as two very related but ultimately different belief change operations. While revision deals with incorporating new information about a static world, update deals with recording changes occurring in a dynamic world. The authors also formulated a separate set of postulates for updates.

More recently, when updates were investigated in the context of Logic Programming, it was shown that the above mentioned postulates and belief change operators are inappropriate for Logic Programming (Eiter et al. 2002), leading to the development of semantics for rule updates based on different principles and constructions, when compared to their classical counterparts. While earlier approaches based on literal inertia (Marek and Truszczynski 1998) proved not sufficiently expressive for dealing with rule updates, the introduction of the causal rejection principle (Leite 2003) lead to several approaches, the one exhibiting better properties being the one presented in (Alferes et al. 2005), not least because it does not allow any kind of cyclic updates to influence the semantics of the original rule base.

Combining both branches of research on updates is a very important, challenging, still unexplored problem.

In this paper, we take the first step in the development of update operators for hybrid knowledge bases. We choose a restricted scenario where rules represent static, general knowledge or behaviour, and first-order logic is used to represent evolving information about the open and dynamic environment. The following simple scenario illustrates the type of situations that our formalism is meant to address:

Example 1 *Suppose we have a search engine for finding ways of transportation. In order to choose from a large number of possibilities, the engine contains general rules of thumb, encoded in rules, about what kind of transportation users prefer in different situations. The system needs to work with evolving information about the currently available ways of transportation, represented in an evolving ontology, and as time goes, users may also personalise the default preferences encoded in the rules of the system.*

For this scenario, and others in which rules represent static behaviour and classical logic represents information about the evolving world, we develop an update operator and examine its basic properties, showing that it:

- adheres to the principle of primacy of new information (Dalal 1988), so every model resulting from the update of a program $\mathcal{P}$ by a theory $\mathcal{U}$ is a model of $\mathcal{U}$.
- yields the same set of models when updating with equivalent theories.
- generalises the stable model semantics (Gelfond and Lifschitz 1988).
- generalises, under certain conditions, the MKNF semantics for hybrid knowledge bases (Motik and Rosati 2007).
- generalises, under certain conditions, the *minimal change update operator* for first-order theories (Winslett 1990).

To the best of authors' knowledge, this is the first time that an update semantics for a hybrid knowledge base composed of nonmonotonic rules and a classical theory has been proposed in a single framework.

The remainder of this paper is structured as follows: In Sect. 2. we introduce the necessary notions that are needed throughout the rest of the paper. Section 3. contains the definition for our operator while in Sect. 4. we examine its properties. In Sect. 5. we discuss some further properties of the operator and related work. We also sketch a number of possible directions for future work.

2. Preliminaries

In this section we present the necessary preliminaries that we need to define the hybrid update operator, and discuss some of the choices we made. As the basis for the formal part of our investigation, we choose the same notation and notions as those used for Hybrid MKNF Knowledge Bases (Motik and Rosati 2007). This makes it possible to treat first-order formulae and nonmonotonic rules in a unified manner and easily compare our semantics to the one of Hybrid MKNF.

MKNF. The logic of Minimal Knowledge and Negation as Failure (MKNF) (Lifschitz 1991) forms the logical basis of Hybrid MKNF Knowledge Bases. It is an extension of first-order logic with two modal operators: $\mathbf{K}$ and $\mathbf{not}$. In the following, we follow the presentation of syntax and semantics of this logic as given in (Motik and Rosati 2007). We assume a function-free first-order syntax extended by the mentioned modal operators in a natural way. A *first-order atom* is a formula $P(t_1, t_2, \ldots, t_n)$ where P is a predicate symbol of arity n and t_i are terms. An MKNF formula ϕ is a *sentence* if it has no free variables; ϕ is *ground* if it does not contain variables; ϕ is *first-order* if it does not contain modal operators. By $\phi[t_1/x_1, t_2/x_2, \ldots, t_n/x_n]$ we denote the formula obtained by simultaneously replacing in ϕ all free occurrences of variable x_i by the term t_i. A set of first-order sentences is a *first-order theory*.

Similarly as in (Motik and Rosati 2007), we only consider Herbrand interpretations in our semantics. Apart from the constants used in formulae, we assume our signature to contain a countably infinite supply of constants not occurring in the formulae. The Herbrand Universe of such a signature is denoted by Δ. The set of all Herbrand first-order interpretations is denoted by $\mathcal{I}$. An *MKNF structure* is a triple $\langle I, M, N \rangle$ where I is a Herbrand first-order interpretation and M, N are nonempty sets of Herbrand first-order interpretations. The satisfiability of an MKNF sentence ϕ in $\langle I, M, N \rangle$ is defined as follows (where p is a ground first-order atom):

$$
\begin{array}{ll}
\langle I, M, N \rangle \models p & \text{iff } I \models p \\
\langle I, M, N \rangle \models \neg\phi & \text{iff } \langle I, M, N \rangle \not\models \phi \\
\langle I, M, N \rangle \models \phi_1 \wedge \phi_2 & \text{iff } \langle I, M, N \rangle \models \phi_1 \text{ and } \langle I, M, N \rangle \models \phi_2 \\
\langle I, M, N \rangle \models \exists x : \phi & \text{iff } \langle I, M, N \rangle \models \phi[c/x] \text{ for some } c \in \Delta \\
\langle I, M, N \rangle \models \mathbf{K}\,\phi & \text{iff } \langle J, M, N \rangle \models \phi \text{ for all } J \in M \\
\langle I, M, N \rangle \models \mathbf{not}\,\phi & \text{iff } \langle J, M, N \rangle \not\models \phi \text{ for some } J \in N
\end{array}
$$

The symbols $\vee$, $\forall$ and $\subset$ (material implication) are interpreted as usual. An *MKNF interpretation* M is a non-empty set of Herbrand first-order interpretations over Δ. Let $\mathcal{T}$ be a set of MKNF sentences and M an MKNF interpretation. M is an *S5 model* of $\mathcal{T}$, written $M \models \mathcal{T}$, if $\langle I, M, M \rangle \models \phi$ for every $\phi \in \mathcal{T}$ and all $I \in M$. If there exists the greatest S5 model M of $\mathcal{T}$, then it is denoted by $\mathsf{mod}(\mathcal{T})$. If $\mathcal{T}$ has no S5 model, then $\mathsf{mod}(\mathcal{T})$ denotes the empty set. For all other sets of formulae, $\mathsf{mod}(\mathcal{T})$ stays undefined. M is an *MKNF model* of $\mathcal{T}$ if M is an S5 model of $\mathcal{T}$ and for every MKNF interpretation $M' \supsetneq M$ there is some $I' \in M'$ such that $\langle I', M', M \rangle \not\models \phi$ for some $\phi \in \mathcal{T}$. For a sentence ϕ, the S5 models of ϕ, MKNF models of ϕ and $\mathsf{mod}(\phi)$ are defined as S5 models of $\{\,\phi\,\}$, MKNF models of $\{\,\phi\,\}$ and $\mathsf{mod}(\{\,\phi\,\})$.

Hybrid MKNF Knowledge Bases. We make use of the general MKNF framework to give a semantics to hybrid knowledge bases composed of a first-order theory and a normal logic program. We define a *rule* to be any formula of the following form

$$\mathbf{K}\,p \subset \mathbf{K}\,q_1 \wedge \mathbf{K}\,q_2 \wedge \cdots \wedge \mathbf{K}\,q_k \wedge \mathbf{not}\,s_1 \wedge \mathbf{not}\,s_2 \wedge \cdots \wedge \mathbf{not}\,s_l \quad (1)$$

where k, l are non-negative integers and p, q_i, s_j are first-order atoms for any $i \in \{\,1, 2, \ldots, k\,\}, j \in \{\,1, 2, \ldots, l\,\}$. Given a rule r of the form (1), the following notation is also defined: $H(r) = \mathbf{K}\,p$, $H^*(r) = p$, $B^+(r) = \{\,\mathbf{K}\,q_1, \mathbf{K}\,q_2, \ldots, \mathbf{K}\,q_k\,\}$, $B^-(r) = \{\,\mathbf{not}\,s_1, \mathbf{not}\,s_2, \ldots, \mathbf{not}\,s_l\,\}$ and $B(r) = B^+(r) \cup B^-(r)$. $H(r)$ is dubbed the *head of* r, $B^+(r)$ the *positive body of* r, $B^-(r)$ the *negative body of* r and $B(r)$ the *body of* r. An MKNF rule r is called *definite* if its negative body is empty; it is called a *fact* if its body is empty. A *program* is a set of rules and a *definite program* is a set of definite rules.

As was shown in (Lifschitz 1991), the MKNF semantics generalises the stable model semantics for logic programs. In particular, every logic programming rule of the form

$$p \leftarrow q_1, q_2, \ldots, q_k, not\,s_1, not\,s_2, \ldots, not\,s_l$$

can be translated into the MKNF formula (1) and the stable models of a sets of such rules (i.e. of normal logic programs) directly correspond to MKNF models of the set of translated rules.

We are now ready to define a hybrid knowledge base and its semantics.

Definition 2 *Let $\mathcal{T}$ be a first-order theory and $\mathcal{P}$ a program. The pair $\langle \mathcal{T}, \mathcal{P} \rangle$ is then called a* hybrid knowledge base.

For a rule r with the vector of free variables $\mathbf{x}$, a program $\mathcal{P}$, a first-order theory $\mathcal{T}$ and the hybrid knowledge base $\mathcal{K} = \langle \mathcal{T}, \mathcal{P} \rangle$, we define: $\pi(r) = (\forall \mathbf{x} : r)$, $\pi(\mathcal{P}) = \{\,\pi(r) \mid r \in \mathcal{P}\,\}$, $\pi(\mathcal{T}) = \{\,\mathbf{K}\,\phi \mid \phi \in \mathcal{T}\,\}$ and $\pi(\mathcal{K}) = \pi(\mathcal{T}) \cup \pi(\mathcal{P})$.

We say an MKNF interpretation M is an S5 model of $\mathcal{K}$ *if M is an S5 model of $\pi(\mathcal{K})$. We say M is an* MKNF model of $\mathcal{K}$ *if M is an MKNF model of $\pi(\mathcal{K})$.*

In this paper, we are not concerned with decidability of reasoning, so we refrain from introducing a safety condition on our rules as was done in (Motik and Rosati 2007).

Classical Updates. As a basis for our update operator, we adopt an update semantics called the Minimal Change Update Semantics (sometimes also called the Possible Models Approach (PMA)) as defined in (Winslett 1990) for updating first-order theories. There are a number of reasons for this choice. First, it satisfies all of Katsuno and Mendelzon's update postulates (Katsuno and Mendelzon 1991). This means, for instance, that unlike some other update semantics, such as the standard semantics (Winslett 1990), it is not sensitive to syntax of the original theory or of the update. Second, it is based on an intuitive idea, treating each classical model of the original theory as a possible world and modifying it as little as possible in order to become consistent with the new information. This idea has its roots in reasoning about action (Winslett 1988) and updates of relational theories (Winslett 1990). Third, the operator has already been successfully used to deal with ABox updates (Liu et al. 2006; Giacomo et al. 2007).

This semantics uses a notion of closeness of first-order interpretations w.r.t. a fixed first-order interpretation I. This notion is based on the set of ground first-order atoms that are interpreted differently than in I.

Definition 3 *Let P be a predicate symbol and I, J be first-order interpretations. The* difference in the interpretation of P between I and J, *written $\mathit{diff}(P, I, J)$, is a relation containing the set of tuples $(P^I \setminus P^J) \cup (P^J \setminus P^I)$.*

Given first-order interpretations I, J, J', we say that J is at least as close to I as J', denoted by $J \leq_I J'$, if for every predicate symbol P it holds that $\mathit{diff}(P, I, J)$ is a subset of $\mathit{diff}(P, I, J')$. We also say that J is closer to I than J', *denoted by $J <_I J'$, if $J \leq_I J'$ and $J' \not\leq_I J$.*

The minimal change update semantics then keeps those models of the updating theory that are the closest w.r.t. the relation $\leq_I$ to some model I of the original theory:

Definition 4 *Let $\mathcal{T}, \mathcal{U}$ be first-order theories, I a first-order interpretation and M a set of first-order interpretations. We define:*

$$\begin{aligned}
\mathsf{incorp}(\mathcal{U}, I) &= \{\, J \mid J \models \mathcal{U} \\
&\qquad \wedge (\nexists J' \in \mathcal{I})(J' \models \mathcal{U} \wedge J' <_I J)\,\} \\
\mathsf{incorp}(\mathcal{U}, M) &= \bigcup_{I \in M} \mathsf{incorp}(\mathcal{U}, I) \\
\mathsf{mod}(\mathcal{T} \oplus \mathcal{U}) &= \mathsf{incorp}(\mathcal{U}, \mathsf{mod}(\mathcal{T}))
\end{aligned}$$

If $\mathsf{mod}(\mathcal{T} \oplus \mathcal{U})$ is nonempty, we say it is the minimal change update model of $\mathcal{T} \oplus \mathcal{U}$.

The previous definition can be naturally generalised to allow for sequences of updates. Starting from the models of the original theory, for each update in the sequence we can transform the set of models according to the minimal change update semantics defined above. The resulting set of models then determines the updated theory. Formally:

Definition 5 *Let $\mathcal{T}$ be a first-order theory, $\mathcal{U}$ a sequence of n first-order theories $(\mathcal{U}_1, \mathcal{U}_2, \ldots, \mathcal{U}_n)$ and M a set of first-order interpretations. We inductively define:*

$$\begin{aligned}
\mathsf{incorp}(\mathcal{U}, M) &= \mathsf{incorp}((\mathcal{U}_2, \ldots, \mathcal{U}_n), \mathsf{incorp}(\mathcal{U}_1, M)) \\
\mathsf{mod}(\mathcal{T} \oplus \mathcal{U}) &= \mathsf{incorp}(\mathcal{U}, \mathsf{mod}(\mathcal{T}))
\end{aligned}$$

If $\mathsf{mod}(\mathcal{T} \oplus \mathcal{U})$ *is nonempty, we say it is the* minimal change update model of $\mathcal{T} \oplus \mathcal{U}$.

3. Hybrid Update Operator

Turning to the formal part of our proposal, our aim is to propose a semantics for a program $\mathcal{P}$ updated by a sequence of first-order theories $(\mathcal{U}_1, \mathcal{U}_2, \ldots, \mathcal{U}_n)$. We assume program $\mathcal{P}$ to be finite and ground, a common assumption when dealing with reasoning under the stable model semantics.

We follow a path similar to how the stable models of normal logic programs were originally defined (Gelfond and Lifschitz 1988), and start by defining how a definite program can be updated by a sequence of first-order theories, and only afterwards deal with programs with default negation.

As with the least model of a definite logic program, our resulting model is the least fixed point of an immediate consequence operator. Our operator is in a way similar to the usual immediate consequence operator $T_{\mathcal{P}}$ commonly used to draw consequences from a logic program. The crucial difference between $T_{\mathcal{P}}$ and our operator is that in the latter, the consequences are subsequently updated by the sequence of theories $\mathcal{U}$ using the classical update operator. Formally:

Definition 6 *Let $\mathcal{P}$ be a finite ground definite program and $\mathcal{U}$ a sequence of first-order theories. We define the operator $T_{\mathcal{P} \oplus \mathcal{U}}$ for any $M \subseteq \mathcal{I}$ as follows:*

$$T_{\mathcal{P} \oplus \mathcal{U}}(M) = \mathsf{mod}\left(\{\, H^*(r) \mid r \in \mathcal{P} \land M \models B(r) \,\} \oplus \mathcal{U}\right)$$

An important property of an immediate consequence operator is *continuity* because it guarantees the existence of a least fixed point and also provides a way of computing this least fixed point (using the Kleene Fixed Point Theorem). The $T_{\mathcal{P} \oplus \mathcal{U}}$ operator satisfies the condition of continuity:

Proposition 7 *Let $\mathcal{P}$ be a finite ground definite program and $\mathcal{U}$ a sequence of first-order theories. Then $T_{\mathcal{P} \oplus \mathcal{U}}$ is a continuous function on the complete partial order of all subsets of $\mathcal{I}$ with the least element $\mathcal{I}$.*

Now we can define a *minimal change dynamic stable model* of $\mathcal{P} \oplus \mathcal{U}$, where $\mathcal{P}$ is a definite program, as the least fixed point of $T_{\mathcal{P} \oplus \mathcal{U}}$:

Definition 8 *Let $\mathcal{P}$ be a finite ground definite program and $\mathcal{U}$ a sequence of first-order theories. We say an MKNF interpretation M is a* minimal change dynamic stable model of $\mathcal{P} \oplus \mathcal{U}$ *if it is the least fixed point of $T_{\mathcal{P} \oplus \mathcal{U}}$.*

Notice that for each definite program $\mathcal{P}$ and sequence of first-order theories $\mathcal{U}$, $\mathcal{P} \oplus \mathcal{U}$ has at most one minimal change dynamic stable model.

In order to deal with default negation in the bodies of rules, we use the Gelfond-Lifschitz transformation which was used to define the stable models of a normal logic program (Gelfond and Lifschitz 1988). We do this by defining the definite program $\mathcal{P}^M$ which is the result of performing the Gelfond-Lifschitz transformation on $\mathcal{P}$ – rules from $\mathcal{P}$ with a negative body that is in conflict with M are discarded, while for all the other rules, their negative bodies are discarded. Then $\mathcal{P}^M$ is updated by $\mathcal{U}$ using the above definition for definite logic programs and if the result is identical to M, then M is given the status of a minimal change dynamic stable model. Hence, the resulting operator can be used to update an arbitrary normal logic program by a first-order theory.

Definition 9 *Let $\mathcal{P}$ be a finite ground program, $\mathcal{U}$ a sequence of first-order theories and M an MKNF interpretation. We say M is a* minimal change dynamic stable model of $\mathcal{P} \oplus \mathcal{U}$ *if M is a minimal change dynamic stable model of $\mathcal{P}^M \oplus \mathcal{U}$ where*

$$\mathcal{P}^M = \left\{\, H(r) \subset B^+(r) \;\middle|\; r \in \mathcal{P} \land M \models B^-(r) \,\right\} \;.$$

The minimal change dynamic stable models can be used to define a consequence relation from $\mathcal{P} \oplus \mathcal{U}$ where $\mathcal{P}$ is a finite ground program and $\mathcal{U}$ a sequence of first-order theories. We offer a definition which adopts a skeptical approach to inference; credulous and other definitions may be obtained similarly.

Definition 10 *Let $\mathcal{P}$ be a finite ground program, $\mathcal{U}$ a sequence of first-order theories and ϕ an MKNF sentence. We say that $\mathcal{P} \oplus \mathcal{U}$* entails *$\phi$, written $\mathcal{P} \oplus \mathcal{U} \models \phi$, if and only if $M \models \phi$ for all minimal change dynamic stable models M of $\mathcal{P} \oplus \mathcal{U}$.*

We now demonstrate the defined update semantics on a simple example:

Example 11 *Using our update operator, we illustrate how to model one simple situation that may arise in the scenario described in Example 1. Suppose the system searches only among flights and trains and by default all flights are preferred to all trains. Further, exactly one preferred solution should appear in each minimal change dynamic stable model. This can be modelled in a logic program $\mathcal{P}$ using the following rules*[1]*:*

$$\mathit{trans}(X) \leftarrow \mathit{flight}(X). \tag{2}$$

$$\mathit{trans}(X) \leftarrow \mathit{train}(X). \tag{3}$$

$$\mathit{prefer}(X,Y) \leftarrow \mathit{flight}(X), \mathit{train}(Y). \tag{4}$$

$$1\,\{\, \mathit{take}(X) : \mathit{trans}(X) \,\}\, 1. \tag{5}$$

$$\leftarrow \mathit{trans}(X), \mathit{trans}(Y), \mathit{pref}(X,Y), \mathit{take}(Y). \tag{6}$$

Rules (2) *and* (3) *say that flights and trains are ways of transport. Rule* (4) *expresses the preference given to flights over trains and rule* (5) *makes sure that exactly one way of transport is present in each stable model of the program.*[2] *Finally, the constraint* (6) *discards solutions for which there exists a more preferred alternative.*

Now suppose that, initially, there are three flights (f_1, f_2 and f_3) and two trains (t_1 and t_2) available, i.e.

$$\mathcal{U}_1 = \{\, \mathit{flight}(f_1) \land \mathit{flight}(f_2) \land \mathit{flight}(f_3) \land \mathit{train}(t_1) \land \mathit{train}(t_2) \,\} \tag{7}$$

[1]For the sake of simplicity, we do not consider different destinations or multiple users of the system. The example is only meant to illustrate how the different situations may be encoded.

[2]This rule is written in Lparse notation which can be translated back into ordinary LP rules. Lparse notation is used by several solvers for computing the stable models of logic programs, such as Smodels and Clasp.

$\mathcal{P} \oplus \mathcal{U}_1$ has three minimal change dynamic stable models, namely M_1, M_2 and M_3, such that $M_1 \models take(f_1)$, $M_2 \models take(f_2)$ and $M_3 \models take(f_3)$.

Next, suppose that we hear on the news that one of the morning flights, f_1 and f_2, has been canceled. This is expressed by the following update:

$$\mathcal{U}_2 = \{ \neg flight(f_1) \vee \neg flight(f_2) \} \tag{8}$$

The only minimal change dynamic stable model of $\mathcal{P} \oplus (\mathcal{U}_1, \mathcal{U}_2)$ is M_4 with $M_4 \models take(f_3)$.

In the sequel, the user notices that train connections are also available and decides to modify her preferences to no longer prefer flight f_3 over train t_1 since this train is very comfortable and fast:

$$\mathcal{U}_3 = \{ \neg prefer(f_3, t_1) \} \tag{9}$$

After this update, there are two minimal change dynamic stable models M_5, M_6 of $\mathcal{P} \oplus (\mathcal{U}_1, \mathcal{U}_2, \mathcal{U}_3)$ such that $M_5 \models take(f_3)$ and $M_6 \models take(t_1)$.

Finally, the danger of a terrorist attack forces the local airport to cancel all flights:

$$\mathcal{U}_4 = \{ (\forall x)(\neg flight(x)) \} \tag{10}$$

Consequently, the two minimal change dynamic stable models of $\mathcal{P} \oplus (\mathcal{U}_1, \mathcal{U}_2, \mathcal{U}_3, \mathcal{U}_4)$ are M_7 and M_8 with $M_7 \models take(t_1)$ and $M_8 \models take(t_2)$.

4. Properties and Relations

In this section we investigate a number of formal properties of the defined operator. The first property guarantees that every minimal change dynamic stable model of $\mathcal{P} \oplus \mathcal{U}$ is a model of the update $\mathcal{U}$. This is known as the *principle of primacy of new information* (Dalal 1988).

Proposition 12 *Let $\mathcal{P}$ be a finite ground program, $\mathcal{U}$ a first-order theory and M a minimal change dynamic stable model of $\mathcal{P} \oplus \mathcal{U}$. Then $M \models \mathcal{U}$.*

The second property guarantees that our operator is syntax-independent w.r.t. the updates. This is a desirable property as it shows that updating by equivalent theories always produces the same result. It is inherited from the classical minimal change update operator.

Proposition 13 *Let $\mathcal{P}$ be a finite ground program, $\mathcal{U}, \mathcal{U}'$ two equivalent first-order theories and M an MKNF interpretation. Then M is a minimal change dynamic stable model of $\mathcal{P} \oplus \mathcal{U}$ if and only if M is a minimal change dynamic stable model of $\mathcal{P} \oplus \mathcal{U}'$.*

The following proposition relates the hybrid update operator to the static MKNF semantics of hybrid knowledge bases. It gives sufficient conditions for the static and dynamic semantics to coincide. In particular, the sufficient condition requires that for any set of consequences S of program $\mathcal{P}$ in the context of a model M, updating S by $\mathcal{U}$ has the same effect as making an intersection of the models of S with the models of $\mathcal{U}$.

Proposition 14 *Let $\mathcal{P}$ be a finite ground program, $\mathcal{U}$ a first-order theory and M an MKNF interpretation such that for every subset S of the set $\{ H^*(r) \mid r \in \mathcal{P} \wedge M \models B(r) \}$ the following condition is satisfied:*

$$\mathsf{mod}\,(S \oplus \mathcal{U}) = \mathsf{mod}\,(S \cup \mathcal{U}) \quad .$$

Then M is an MKNF model of $\langle \mathcal{U}, \mathcal{P} \rangle$ if and only if M is a minimal change dynamic stable model of $\mathcal{P} \oplus \mathcal{U}$.

The previous proposition has a number of consequences. In particular, when $\mathcal{U}$ is empty, the required condition is always satisfied. Hence, the minimal change dynamic stable models of $\mathcal{P} \oplus \emptyset$ are exactly the MKNF models of $\mathcal{P}$ and since the MKNF semantics generalises the stable model semantics (Lifschitz 1991), they also coincide with the stable models of $\mathcal{P}$.

Corollary 15 *Let $\mathcal{P}$ be a finite ground program. Then M is an MKNF model of $\mathcal{P}$ if and only if M is a minimal change dynamic stable model of $\mathcal{P} \oplus \emptyset$.*

Corollary 16 *Let $\mathcal{P}$ be a finite ground program. Then M is a stable model of $\mathcal{P}$ if and only if M is a minimal change dynamic stable model of $\mathcal{P} \oplus \emptyset$.*

Turning to relations with the minimal change update operator, we show that updating a logic program containing only facts has the same effect as updating a first-order theory with these facts. Hence, our update operator generalises the classical minimal change update operator.

Proposition 17 *Let $\mathcal{P}$ be a finite ground program containing only facts, $\mathcal{U}$ a sequence of first-order theories and M an MKNF interpretation. Then M is a minimal change dynamic stable model of $\mathcal{P} \oplus \mathcal{U}$ if and only if M is a minimal change update model of $\mathcal{T}_\mathcal{P} \oplus \mathcal{U}$ where $\mathcal{T}_\mathcal{P} = \{ p \mid \mathbf{K}\, p \in \mathcal{P} \}$.*

Another property that our operator inherits from the classical minimal change update operator is that empty theories in the updating sequence do not influence the resulting models. Similarly, updating an empty program simply yields the set of all first-order models of the update. These last two properties ensure that empty program and updates cannot influence the resulting models under our update operator.

Proposition 18 *Let $\mathcal{P}$ be a finite ground program, $\mathcal{U} = (\mathcal{U}_1, \mathcal{U}_2, \ldots, \mathcal{U}_n)$ a sequence of first order theories (where $n \geq 1$) and let $\mathcal{U}' = (\mathcal{U}_1, \mathcal{U}_2, \ldots, \mathcal{U}_{i-1}, \mathcal{U}_i, \emptyset, \mathcal{U}_{i+1}, \ldots, \mathcal{U}_n)$ for some $i \in \{ 0, 1, 2, \ldots, n \}$. Then an MKNF interpretation M is a minimal change dynamic stable model of $\mathcal{P} \oplus \mathcal{U}$ if and only if M is a minimal change dynamic stable model of $\mathcal{P} \oplus \mathcal{U}'$.*

Proposition 19 *Let $\mathcal{U}$ be a first-order theory and M be an MKNF interpretation. Then M is a minimal change dynamic stable model of $\emptyset \oplus \mathcal{U}$ if and only if $M = \mathsf{mod}(\mathcal{U})$.*

5. Discussion and Future Work

As seen, our operator properly generalises the two main ingredients that it is motivated by – the stable model semantics of normal logic programs (Corollary 16) and the classical minimal change update operator (Proposition 17). We now briefly discuss its relation to Katsuno and Mendelzon's postulates for updates of finite propositional knowledge bases formulated in (Katsuno and Mendelzon 1991). Each such knowledge base can be represented by a single propositional formula and the result of the update can also be represented as a propositional formula. The eight desirable properties of an update operator $\diamond$ are as follows:

KM 1: $\phi \diamond \psi$ implies ψ.
KM 2: If ϕ implies ψ, then $\phi \diamond \psi$ is equivalent to ϕ.
KM 3: If both ϕ and ψ are satisfiable, then $\phi \diamond \psi$ is satisfiable.
KM 4: If ϕ_1 is equivalent to ϕ_2 and ψ_1 is equivalent to ψ_2, then $\phi_1 \diamond \psi_1$ is equivalent to $\phi_2 \diamond \psi_2$.
KM 5: $(\phi \diamond \psi) \wedge \chi$ implies $\phi \diamond (\psi \wedge \chi)$.
KM 6: If $\phi \diamond \psi_1$ implies ψ_2 and $\phi \diamond \psi_2$ implies ψ_1, then $\phi \diamond \psi_1$ is equivalent to $\phi \diamond \psi_2$.
KM 7: If for each atom p either ϕ implies p or ϕ implies $\neg p$, then $(\phi \diamond \psi_1) \wedge (\phi \diamond \psi_2)$ implies $\phi \diamond (\psi_1 \vee \psi_2)$.
KM 8: $(\phi_1 \vee \phi_2) \diamond \psi$ is equivalent to $(\phi_1 \diamond \psi) \vee (\phi_2 \diamond \psi)$.

In order to examine these postulates in our setting, we restrict our attention to a finite propositional language. We also need to define the meaning of a number of notions used in the postulates. Let $\mathcal{P}, \mathcal{P}_1, \mathcal{P}_2$ be logic programs and μ, μ_1, μ_2 be propositional formulae. We need to discuss and define, at least:

1. When does $\mathcal{P} \oplus \mu_1$ imply μ_2? (used in KM 1 and KM 6)
2. When does $\mathcal{P}$ imply μ? (used in KM 2 and KM 7)
3. When is $\mathcal{P}_1 \oplus \mu$ equivalent to $\mathcal{P}_2$? (used in KM 2)
4. When is $\mathcal{P}$ satisfiable? (used in KM 3)
5. When is $\mathcal{P} \oplus \mu$ satisfiable? (used in KM 3)
6. When is $\mathcal{P}_1$ equivalent to $\mathcal{P}_2$? (used in KM 4)
7. When is $\mathcal{P}_1 \oplus \mu_1$ equivalent to $\mathcal{P}_2 \oplus \mu_2$? (used in KM 4 and KM 6)
8. What is the semantics of $(\mathcal{P} \oplus \mu_1) \wedge \mu_2$? (used in KM 5)
9. What is the semantics of $(\mathcal{P} \oplus \mu_1) \wedge (\mathcal{P} \oplus \mu_2)$? (used in KM 7)
10. What is the semantics of $\mathcal{P}_1 \vee \mathcal{P}_2$? (used in KM 8)

Most of these questions can be answered in multiple different ways while some of them are hard to provide answers to at all. In the following, we suggest ways of answering most of these questions and then analyse whether our operator satisfies the corresponding postulates.

Question 1. can be answered using the consequence relation from Def. 10. A similar consequence relation can be defined using stable models to answer question 2. A simple answer to question 3 is to say that $\mathcal{P}_1 \oplus \mu$ is equivalent to $\mathcal{P}_2$ if the set of minimal change dynamic stable models of $\mathcal{P}_1 \oplus \mu$ is equal to the set of stable models of $\mathcal{P}_2$. Regarding questions 4. and 5., we can say that $\mathcal{P}$ is satisfiable if it has at least one stable model and, similarly, $\mathcal{P} \oplus \mu$ is satisfiable if it has at least one minimal change dynamic stable model. Question 6. can be answered similarly as question 3. by comparing the sets of minimal change dynamic stable models of $\mathcal{P} \oplus \mu_1$ and $\mathcal{P} \oplus \mu_2$. Finally, question 7. can be answered by comparing the sets of stable models of $\mathcal{P}_1$ and $\mathcal{P}_2$ or by using strong equivalence (Lifschitz, Pearce, and Valverde 2001). Providing reasonable answers to the remaining questions requires more investigation, so, for now, we do not further examine postulates KM 5, KM 7 and KM 8.

Turning to the rest of the postulates, we note that our operator adheres to KM 1, which was proved in Proposition 12. The same is not the case with postulate KM 2, as shown by the following counterexample. Consider the program

$$\begin{aligned} \mathcal{P}: \quad & \mathbf{K}\, p \subset \mathbf{not}\, q \\ & \mathbf{K}\, q \subset \mathbf{not}\, q \\ & \mathbf{K}\, r \subset \mathbf{not}\, r, \mathbf{K}\, q \\ & \mathbf{K}\, r \subset \mathbf{K}\, p \end{aligned} \tag{11}$$

and the update $\mu = r$. The only stable model of $\mathcal{P}$ is the maximal S5 model M of $\{\, p, r \,\}$. Clearly, $M \models \mu$. But $\mathcal{P} \oplus \mu$ has another minimal change dynamic stable M', which is the maximal S5 model of $\{\, q, r \,\}$. In fact, this behaviour is inherited from the stable semantics for logic programs which does not satisfy the very similar property of *cumulativity* (Dix 1995). Hence, it is expectable that KM 2 is never satisfied by any update semantics that properly generalises the stable model semantics.

A similar situation arises with postulate KM 3, because the stable model semantics allows to express integrity constraints, and these may easily be broken by an update. For example, the program $\mathcal{P} = \{\, \mathbf{K}\, p \subset \mathbf{not}\, p, \mathbf{K}\, q \,\}$, updated by $\mu = q$, of which both are satisfiable, does not allow for any minimal change dynamic stable model. It is not clear how an integrity constraint should be updated because, once it is a part of the knowledge base, which is assumed to be a correct representation of the world, it should not be violated, and no new information should have the power to override it. Or should it? That is another open research question worth investigating.

Postulate KM 4 is partially formulated in Proposition 13, which shows that updating by equivalent theories produces the same result. In order to formulate the other half, we would need to prove that updating equivalent logic programs by the same theory also produces the same result. For the two notions of program equivalence that we proposed this postulate does not hold. As a counterexample take $\mathcal{P}_1 = \{\, \mathbf{K}\, p, \mathbf{K}\, q \,\}$ and $\mathcal{P}_2 = \{\, \mathbf{K}\, p, \mathbf{K}\, q \subset \mathbf{K}\, p \,\}$ which have the same answer sets and are also strongly equivalent. An update by $\mu = \neg p$, produces two different results, which we believe is in accordance with intuitions regarding these two programs. It may be the case that for different notions of program equivalence that better suit our scenario, such as the update equivalence of logic programs proposed in (Leite 2003), this property holds. Further investigation is needed to answer this question.

Finally, postulate KM 6 is also not satisfied by the operator. As a counterexample we can take the program $\mathcal{P}$ defined in (11), $\mu_1 = r$ and $\mu_2 = p \vee q$.

However, the failure of our operator to satisfy many of Katsuno and Mendelzon's postulates is not surprising. A wide range of classical update and revision postulates was already studied in the context of rule updates, only to find that many of them were inappropriate for characterising plausible rule update operators (Eiter et al. 2002). The search for desirable properties of hybrid update operators is an interesting future research area.

To conclude, in this paper, to the best of our knowledge, we proposed the first update operator for hybrid knowledge bases. We deal with a scenario in which the rules represent static knowledge, behaviour or norms of the domain or agent, and the classical theory is used to represent the current state of the open and dynamic environment which may evolve with time. We proved a number of properties of our operator, among which its relations with the theories it was based on, such as the stable model semantics (Gelfond and Lifschitz 1988), the MKNF semantics for hybrid knowledge bases (Motik and Rosati 2007) and the minimal change update operator for first-order theories (Winslett 1990).

This is only the first step to address updates of hybrid knowledge bases. There are many ways in which the current operator can be generalised, and many properties still to be examined, among them decidability as well as complexity of reasoning. Since we cannot expect the operator to perform any better than the stable model semantics and the classical update operator it is based on, its tractable approximations need to be defined and examined. The well-founded semantics for logic programs (Gelder, Ross, and Schlipf 1991) and its version for hybrid MKNF knowledge bases (Alferes, Knorr, and Swift 2009) constitute crucial starting points. The recent research on ontology evolution in general (see (Flouris et al. 2008) for a survey), and updates of description logic ABoxes (Liu et al. 2006; Giacomo et al. 2007) in particular, can help design tractable update operators for hybrid knowledge bases.

The rule part was assumed to be static in this paper, but in truly dynamic environments, rules should also be allowed to be updated. The large body of work on rule updates (Leite 2003; Alferes et al. 2005) needs to be exploited in the attempts to define an update operator that can deal with the evolution of both rules and the classical part.

While incorporating new knowledge in a knowledge base is important, the complementary task of removing a certain piece of information is also important. Hence, hybrid erasure operators should be studied and related to hybrid update operators. Again, the classical work on erasure operators as well as on erasure in description logics (Giacomo et al. 2007) should be the starting point of this research.

We believe that this new area of research brings exciting new problems to solve and will bridge a number of existing research areas. It will certainly find many applications and perhaps even provide further philosophical insights into how human knowledge evolves.

References

Alchourrón, C. E.; Gärdenfors, P.; and Makinson, D. 1985. On the logic of theory change: Partial meet contraction and revision functions. *Journal of Symbolic Logic* 50(2):510–530.

Alferes, J. J.; Banti, F.; Brogi, A.; and Leite, J. A. 2005. The refined extension principle for semantics of dynamic logic programming. *Studia Logica* 79(1):7–32.

Alferes, J. J.; Knorr, M.; and Swift, T. 2009. Queries to hybrid MKNF knowledge bases through oracular tabling. In *Procs. of ISWC 2009*, 1–16.

Baader, F.; Calvanese, D.; McGuinness, D. L.; Nardi, D.; and Patel-Schneider, P. F., eds. 2003. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press.

Dalal, M. 1988. Investigations into a theory of knowledge base revision. In *Procs. of AAAI'88*. AAAI Press / The MIT Press.

Dix, J. 1995. A classification theory of semantics of normal logic programs: I. strong properties. *Fundam. Inform.* 22(3):227–255.

Eiter, T.; Fink, M.; Sabbatini, G.; and Tompits, H. 2002. On properties of update sequences based on causal rejection. *Theory and Practice of Logic Programming (TPLP)* 2(6):721–777.

Flouris, G.; Makanatas, D.; Kondylakis, H.; Plexousakis, D.; and Antoniou, G. 2008. Ontology change: classification and survey. *The Knowledge Engineering Review* 23(2):117–152.

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 *Procs. ICLP'88*, 1070–1080. MIT Press.

Giacomo, G. D.; Lenzerini, M.; Poggi, A.; and Rosati, R. 2007. On the approximation of instance level update and erasure in description logics. In *Procs. of AAAI'07*. AAAI Press.

Hitzler, P., and Parsia, B. 2009. Ontologies and rules. In *Handbook on Ontologies*. Springer. 111–132.

Katsuno, H., and Mendelzon, A. O. 1991. On the difference between updating a knowledge base and revising it. In *Procs. of KR'91*. Morgan Kaufmann Publishers.

Leite, J. A. 2003. *Evolving Knowledge Bases*, volume 81 of *Frontiers of Artificial Intelligence and Applications*. IOS Press.

Lifschitz, V.; Pearce, D.; and Valverde, A. 2001. Strongly equivalent logic programs. *ACM Transactions on Computational Logic (TOCL)* 2(4):526–541.

Lifschitz, V. 1991. Nonmonotonic databases and epistemic queries. In *Procs. of IJCAI'91*.

Liu, H.; Lutz, C.; Miličić, M.; and Wolter, F. 2006. Updating description logic ABoxes. In *Procs. of KR'06*. AAAI Press.

Marek, V., and Truszczynski, M. 1998. Revision programming. *Theoretical Computer Science* 190(2):241–277.

Motik, B., and Rosati, R. 2007. A faithful integration of description logics with logic programming. In *Procs. of IJCAI'07*.

Winslett, M. 1988. Reasoning about action using a possible models approach. In *Procs. of AAAI'88*. AAAI Press / The MIT Press.

Winslett, M. 1990. *Updating Logical Databases*. Cambridge University Press.