Operator decision in naval action's simulations

Isabelle Toulgoat
DCNS Ingénierie, Le Mourillon, BP 1306, 83 076 Toulon Cedex, France,isabelle.toulgoat@dcnsgroup.com
Pierre Siegel
Université de Provence,France, siegel@cmi.univ-mrs.fr
Yves Lacroix
Systèmes Navals Complexes, Avenue Georges Pompidou, 83160, La Valette du Var, France,yves.lacroix@univ-tln.fr
Julien Botto
DCNS Ingénierie,Le Mourillon, BP 1306, 83 076 Toulon Cedex, France,julien.botto@dcnsgroup.com

Abstract

Naval action's simulations estimate the operational performance of warships or submarines for a given scenario. In common models, the operator's reactions are predefined. This is not realistic: the operator's decision can produce unexpected reactions.
This article presents a method to model operator decision in simulations. This method allows to reason about incomplete, revisable and uncertain information: an operator has only partial information about his environment and must revise his decisions. Our method uses a non-monotonic logic: the rules of behavior are formalized with the default logic, to which we added a consideration of time. Our method uses preferences to manage choice between different rules, with simple probability techniques.
This method has been implemented in Prolog, interfaced to DCNS simulator framework and applied to a scenario involving two adverse submarines.

1. Introduction

Naval action's simulations estimate the operational performance of warships or submarines for a given scenario. At DCNS, the simulator framework ATANOR models complex scenarios involving several platforms with combat system and equipment (Toulgoat et al. 2009).
In this simulator framework, the behavior is modelled with Petri nets (Petri 1962), composed of places, which model the equipment states and transitions between these places. These transitions are activated by internal and external events. Only one place is activated at the same time, which forbids the simultaneous actions.
The modelling of behavior rules with Petri nets provides automatic reactions of the combat system to a tactical situation. This is not realistic: in a tactical situation the decision of an operator is a key aspect, which can provide unexpected reactions. Moreover, a disadvantage of the modelling with Petri nets is to have to revise its implementation for any new behavior (Ferber 1995).

The purpose of this work is to develop a system allowing to model the behavior of an operator in the performance simulations. We worked on a study case involving two adverse submarines. This system has to answer several requirements:

- to be able to model the behavior rules of the operator.
- to be able to reason with incomplete, revisable and uncertain information. Indeed, an operator only has a partial sight of his environment. This environnement is always changing: the submarine can lose the detection, it hasn't the exact position of its adversary, it is just an estimation .... Therefore he must reason with uncertain and incomplete information. His decisions must be revised with the arrival of new information (Sombé 1989) (Cordier and Siegel 1992).
- to choose between different proposals when the system proposes several actions for a same situation.
- to allow the addition of new behavior rules, without having to modify the knowledge representation and without calling into question the previous rules (unlike the Petri nets, in which the modifications are complicated).
- to be able to reason with general rules, without having to compile in a very precise way all the information. It is not necessary for the user to describe all the possibilities.

This work is financed by the company DCNS for military applications: we need a simple and robust program. Therefore, we used the most known nonmonotonic logic: the default logic. We added a consideration of time: we have submarine's data at the time t, and the extensions calculus gives all the possible extensions at the next time $t+1$. Each extension is a proposal for the action of the submarine. We calculate a weight function for each extension, thanks to preferences on defaults. Then, we use simple probability techniques to choose between these extensions. This work has been implemented using Prolog, and interfaced with DCNS simulator.

In the following article we will first present the study case and some behavior rules. Then, we will present the limits of the classical logic and why we need the non-monotonic logic. We will explain the formalization of the behavior rules with the default logic. We use only normal defaults and Horn clauses in order to symplify the program, though we could extend this work to other case studies, with more complicated rules. Next we explain the choice between the extensions thanks to preferences with simple probability techniques. Finally, some results are presented.

2. Case study: submarine detection and tracking

In a scenario including two submarines, we model the decision of an on watch officer in the submarine according to the events perceived on the tactical situation. In this purpose, we questioned submariners about this study case, and we inferred behavior rules.

Here are some examples of these rules:

- Rule 1: As long as the submarine has no detection, it continues a random research trajectory in its patrol area. During that process, the submarine makes successive straight sections: it goes straight ahead and sometimes it changes his course. The submarine is deaf in its rear (behind the submarine, the sonar's reception is decreased for several reasons), this manoeuvre allows the submarine to check that it isn't tracked. With this manoeuvre, the submarine covers the entire patrol zone, in order to increase its chances to detect a intruder.

  Remark: it is a rule of minimal change (Ginsberg and Smith 1987) (Winslett 1988) (Cordier and Siegel 1992). This rule is applied as long as the submarine has no new information.
- Rule 2: If the submarine detects another submarine, the officer engages the following actions:
  - Collision avoidance manoeuvre.
  - Elaboration of the solution manoeuvre: he manoeuvres in order to confirm his information about the distance, the course and the speed of the enemy.
  - Bypassing of the enemy manoeuvre: when the officer is sure not to be detected, he gets closer to the enemy's rear, position in which he won't be detected.
  - Tracking manoeuvre: when the submarine is in the enemy's rear, it begins the tracking: it makes straight sections in the enemy's rear, avoiding to be detected and keeping good information about the enemy's kinematics.
- Rule 3: If the submarine is detected when it makes one of the following manoeuvres: elaboration of the solution, bypassing of the enemy or tracking, it must escape: the officer manoeuvres in order to go away from the enemy, aiming the loss of contact.
- Rule 4: If the submarine is a diesel submarine and more than a few hours passed since the last battery charge, it must rise to the surface and use the snorkel to take air from the surface and evacuate exhaust gas.
- Rule 5: If the submarine loses the contact during the tracking, the officer rallies the last position of the the adversary and searches for it.
  If he finds it, he resumes the tracking actions (Rule 2).
  If after one hour he hasn't found him, he resumes the random research trajectory (Rule 1).
  During this research hour, the submarine can't rise and use the snorkel (Rule 4).
- Rule 6: With the sonar called MOAS (Mine and Obstacle Avoidance System), the submarine can detect mines, big rocks, cliffs. If the submarine detects a big rock, it changes its course in order to place the rock to one side.

These rules can be in competition: at a same time, it is possible that the submarine needs to do two actions. For example, it needs to rise the surface and use the snorkel and it needs to continue the tracking. The system must be able to manage these alternative choices.

3. Classical logic and its limits

The classical logics, as the mathematical or the propositional logics, are monotonic : if we add information or a formula E' to a formula E, everything which was deducted from E will be deducted from E $\cup$ E '.This monotonicity will generate problems to reason with incomplete, uncertain and revisable information. Indeed, in this case, it can happen that previously established conclusions turn invalid due to new information arrival or information change.

- The classical logic doesn't allow to reason about incomplete information. Let us take the rule: "Generally, a submarine with no detection makes a random research trajectory". At first sight, we can express this type of information with the first order logic:
  Rule 1: $\forall x, \neg detection(x) \rightarrow random\_trajectory(x)$
  This formulation is coherent if the only known information is "The submarine has no detection".
  But if we had the rule: "If more than four hours passed since the last battery charge, the submarine must rise to the surface and use the snorkel to take air.", we express it with the first order logic:
  Rule 2: $\forall x, Tlc(x) \geq 4 \rightarrow snorkel(x)$, where Tlc denotes the time since the last charge and $snorkel$ the action of rising to the surface and using the snorkel.
  With these rules, it is difficult to manage general rules containing an important number of exceptions (Sombé 1989).
- The classical logic doesn't allow to review the information: it doesn't plan to review the previously established deductions. Let us take again the rules 1 and 2.
  Knowing that the submarine has no detection, we deduct that it must make a random trajectory. But, if we know that more than four hours passed since the last battery charge, we conclude that the submarine must use the snorkel.
  We obtain two conclusions which are not consistent: the submarine can't make at the same time these two actions. It illustrates how classical logics don't allow reviewing the reasoning and the conclusions. This kind of reasoning is common in artificial intelligence, as well as in the daily life.

In the case of a submarine, blind in submersion, the only information comes from the passive sonar system, this information is uncertain and incomplete (Prouty 2007). The officer must be able to revise the decisions with the arrival of new information. We need a logic which allows to reason about incomplete, uncertain and revisable information.

4. Non-monotonic logic and default logic

A non-monotonic logic allows to eliminate the monotony property of the classical logic: if a reasoning gives some conclusions using some given knowledge, these conclusions could be reviewed with the addition of new knowledge.
A non-monotonic logic allows to take the incomplete, revisable, uncertain information into account. This logic has a natural similarity with the human reasoning: due to the lack of information or lack of time, one can reason with partial knowledge and revise the conclusions when one has more information.
The default logic, introduced by Ray Reiter (Reiter 1980), is the most used logic. It formalizes the default reasoning: sense conclusions can be made, in the absence of opposite proof. A default logic is defined by $\Delta = (D, W)$, W is a set of facts (formulae from the propositional logic or the first order logic), and with D, a set of defaults, (inference rules with specific content, which handle the uncertainty).

Let us remind the definitions of defaults and extensions:

Definition: Default A default is an expression of the form: $\frac{A(X) : B(X)}{C(X)}$, where A(X), B(X) and C(X) are formulae and X is a set of variables. A(X) is the prerequisite, B(X) is the justification and C(X) is the consequent. Intuitively, the default $\frac{A(X) : B(X)}{C(X)}$ means: if A(X) is true, if it is possible that B(X) is true (B(X) is consistent), then C(X) is true.
If $B(X) = C(X)$, the default is normal. The normal default means: " Normally, the As are Bs".

Definition: Extension The use of defaults allows to deduct more formulae from a knowledge base W. To generate the deducted formulae, we calculate extensions, which are defined as follows:
E is an extension of Δ if and only if $E = \cup_{i=0,\infty} E_i$, with

$$E_0 = W \text{ and for } i \geq O,$$
$$E_{i+1} = Th(E_i) \cup \{C/(\frac{A : B}{C}) \in D, A \in E_i, \neg B \notin E\}$$

where $Th(E_i)$ is the set of theorems obtained in a monotonic way from E_i.

It is important to notice that E appears in the definition of E_{i+1}. So, we need to know E to find E_i, it is not possible to obtain the extensions with an incremental algorithm.

If we work with normal defaults, the definition of an extension is changed: we need to verify that $\neg B \notin E_i$:

$$E_0 = W \text{ and for } i \geq O,$$
$$E_{i+1} = Th(E_i) \cup \{B/(\frac{A : B}{B}) \in D, A \in E_i, \neg B \notin E_i\}$$

where $Th(E_i)$ is the set of theorem obtained in a monotonic way from E_i.

For our study case, we only use normal defaults, but we could extend our work to general defaults.

5. Rules formalization with the default logic

5.1 Time's consideration

To formalize the rules of behavior, we used the default logic, to which we added a consideration of time. Indeed, we have submarines data at the time t, and we have to deduct the submarines instructions at the next time t+1, taking into account the state of the submarine and updated information. These instructions will generate the submarine's updated data for time t+1. To introduce the time, we used previous work by Cordier and Siegel (Cordier and Siegel 1992).
We need the time's consideration in the definitions of the facts W and the defaults D of the default logic $\Delta = (D, W)$.

5.2 Facts definition with time's consideration

The set of facts W is defined with formulae from the propositional logic or the first order logic. We use only Horn clauses. They allow us to write two types of rules:

- the Horn clauses with a positive literal, written as follows: $(g(t) \vee \neg f_1(t) \vee \ldots \vee \neg f_k(t))$, where the $f_i(t)$ and $g(t)$ are positive literal at time t. This formula can also be written with an implication: $(f_1(t) \wedge \ldots \wedge f_k(t)) \rightarrow g(t)$. This type of rules allows to define rules which are always true, these are classic rules of expert systems.
  Example: we formalise a rule such as "If the submarine has a random research trajectory, it turns by an angle between α and β", as follows: $random\_trajectory(X_t) \rightarrow turn(X_t, (\alpha, \beta))$
- the Horn clauses with no positive literal, written as: $(\neg f_1(t) \vee \ldots \vee \neg f_k(t))$, ie $\neg(f_1(t) \wedge \ldots \wedge f_k(t))$. We use these rules to define mutual exclusions in pairs, these are the predicates which cant be executed at the same time: $(\neg f_1(t) \vee \neg f_2(t))$, equivalent to $\neg(f_1(t) \wedge f_2(t))$.
  Example : we can define a rule such as "The submarine can't make at the same time a random research trajectory and rise to use the snorkel" as follows: $\neg(random\_trajectory(X_t) \wedge snorkel(X_t))$.

5.3 Default definition with time's consideration

The defaults D are inference rules with specific content, they allow to manage uncertainty. They express the fact that, if there is no contradiction to execute an action, the submarine can do it. We use here only normal defaults. They allow us to formalize rules such as "If the submarine has no detection, then it makes a random research trajectory" in the following way: $\frac{\neg detection(X_t) : random\_trajectory(X_{t+1})}{random\_trajectory(X_{t+1})}$ This default means: "If the submarine has no detection at time t and if it can make a random research trajectory at time t+1, it makes a random research trajectory at time t+1".

The defaults allow us to define general rules on the behavior of the submarine (rise to the surface to use snorkel, collision avoidance, tracking ...). Then, the set of facts allows to specify, for each behavior, the action to realize (change course, speed, submersion) and the mutual exclusions between the behaviors.

5.4 Extension calculus

We use extensions calculus to study all the defaults and to retain the defaults which answer to the problem in a coherent way. Each extension is a possible solution to the problem: according to the submarine state at the time t, an extension gives a possible solution of action for the submarine at the next time t+1. The normal defaults grant the existence of at least one extension. Generally, we will have several extensions for the same knowledge base.
We could use the answer set programming (Nicolas, Garcia, and Stephan 2005) to calculate the extensions, which are equivalent. In order to have a simple system, we rather implement our own extension calculus. The normal defaults and the Horn clauses allow to implement easily the extensions calculus with the language Prolog. We called our programm NoMROD for Non-Monotonic Reasoning for Operator Decision.

6. Extensions selection with preferences

The aim of this part is to simulate the officer decision. In a tactical situation, the decision of an operator is a key aspect. At each time, the officer has to choose between the actions. We define a method to choose between the different extensions proposed. This method allows to simulate different types of behavior, depending on the officer's character.
We distinguish two stages in this method. The first stage consists in defining general principles for the extension selection, and a weight function for the extension with a multi-criteria decision aid method. This weight function is a general formula, which can be used for other study cases. This function handles the officer's behavior at two levels:

- the importance of each default according to different criteria, with preference's coefficients $C_1, \ldots, C_n$, which allow to define preferences on defaults.These coefficients describe the importance of each action.
- the officer character with the character coefficients $\beta_1, \ldots, \beta_n$: which criteria will he favor? Thanks to these character coefficients, we can model different officers: careful, bold ...

Next, the second stage is more specific to the study case. We define a method to select an extension, thanks to the weight function.

6.1 Extension selection principles

First, we study some principles and requirements to which the extension selection must answer:

1. to choose the interesting extension and let the others aside.
   Example: NoMROD proposes two extensions: to make a random research trajectory or to rise to the surface and use the snorkel to take air.
   The random research trajectory is a rule of minimal change: it is applied while the submarine has no new information. The officer will choose to rise to the surface, which is a more important behavior.
2. to choose the extensions which are obligatory for the crew survival.
   For example, the officer can postpone the rising to use the snorkel, if he is doing another important action (for example the tracking). When this rule is verified, the officer has approximately thirty minutes of battery. When this reserve will be practically empty, the officer will be obliged to rise.
3. to manage the choice between several extensions.
   Example:NoMROD proposes two extensions: avoid the collision with a submarine and avoid the collision with a big rock. These two behaviors are very important for the submarine safety. NoMROD must be able to choose between extensions which have the same importance.
4. to respect the minimal change: while the submarine has no new information, it stays in the same state, it doesn't change its behavior. We must give more chance to an action already engaged.
   With this rule, the officer will persist in its choices, he won't oscillate between several behaviors. However, NoMROD must be able to stop an action if another becomes obligatory.
   Example: NoMROD proposes two extensions: to track the enemy and to rise to the surface and use the snorkel to take air. We suppose that the system chooses the tracking. These two extensions will be proposed again while the rising won't be executed. The best choice for the officer is to carry on the tracking, and when the rising becomes obligatory (the submarine has just enough battery to rise to the surface), he must execute this action.
   When the submarine begins an action, it seems important to carry on this action during a certain time. The officer can't change his opinion too often.
5. the enemy submarine doesn't have to guess which action our officer will select.

These principles are some principles of common sense. Aside from the last principle, the other can be applied generally to other study cases.

6.2 Extensions weight function

We define a weight function to give weights to the extensions.These weights quantify the extensions importance. In order to define this weight function, we use a method of multi-criteria decision aid (MCDA). MCDA aims at modelling the preferences of a decision maker. It allows the decision maker to solve complex problems, where several criteria must be handled in the choice (Ben Mena 2000),(Vincke 1989). There are several categories of methods in MCDA. We use the multi-attribute utility theory. This theory is based on the following axiom:
Every decision maker tries unconsciously to maximize a function $U = U(g_1, \ldots, g_n)$, which aggregates all the points of view to be handled (with g_i the criteria).
In order to define a general method to define the extensions weight functions, we use a simple aggregation function: the weighted sum.

Each extension uses defaults. First, we attribute preference's coefficients on defaults. Next, we calculate the

utility function for each extension, and finally we calculate the weight functions for each extension using the weighted sums.

Preference's coefficients on defaults Each default is a general behavior. In order to specify the importance of behaviors, we attribute preference's coefficients to the defaults. In a similar way, the preferences are used within a system of nonmonotonic reasoning, allowing finding an appropriate compromise solution. For example, Brewka (Brewka 1994) defined a prioritized default logic, with a definition of order in which defaults must be applied.
We define different preference's coefficients $C_1, \ldots, C_n$ to specify the importance of the defaults according to different criteria.
For example, we can specify the default importance for the submarine safety, the efficiency on the submarine mission, the order obedience, ecologist criterion ...
For each default D_j, we attribute values to these coefficients $C_{1j} \ldots C_{kj}, \ldots, C_{nj}$. We fixed arbitrarily these coefficients between 0 and 1000.

Utility function Each extension E_i uses defaults: $E_i = \{D_1 \ldots, D_m\}$.
For each extension E_i, we calculate the scores $\{Score_1(E_i), \ldots, Score_n(E_i)\}$, which correspond respectively with the coefficients $C_1, \ldots, C_n$.
A score $Score_k(E_i)$ is the sum of the coefficients C_k of each default used in the extension E_i: $Score_k(E_i) = \sum_{j=1}^{m} C_{kj}$.

We suppose that NoMROD proposes p extensions. For each $Score_k(E_i)$ for an extension E_i, we calculate an utility function $\mu_k(E_i)$. The sum of the utility functions μ_k must be egal to 1: $\sum_{j=1}^{p} \mu_k(E_j) = 1$. The score $Score_k(E_i)$ is divided by the sum of all the $Score_k(E_j)$ of the p extensions proposed by the system:
$\mu_k(E_i) = \frac{Score_k(E_i)}{\sum_{j=1}^{p} Score_k(E_j)}$. $\mu_k(E_i)$ is the evaluation of the extension E_i, according to the criterion C_i.

Officer's character We want to model the officer's character. According to his character, the officer will use different tactics.
For example, a careful officer will give more importance to the behaviors which ensure the submarine's safety. A bold officer will favor the efficient behavior for the submarine mission. To model these characters, we define character coefficients $\beta_1, \ldots, \beta_n$ in the weight function, such as the sum of these coefficients is equal to one $\sum_{k=1}^{n} \beta_k = 1$.

Weight function Finally, we use a simple function to aggregate criteria: the weighted sum. For each extension E_i, we have:
$P(E_i) = \sum_{k=1}^{n} \beta_k \mu_k(E_i)$, with $\sum_{k=1}^{n} \beta_k = 1$. The sum of the extension weight functions has the following property: $\sum_{i=1}^{p} P(E_i) = 1$. We obtain a general formula, which could be easily reused with other study case.

The use of the weighted sum as aggregation function implies some limitations (Grabisch and Labreuch 2005). More specially, this function can favor the extreme extensions (for example, an extension with an utility function very small for a criterion and with an utility function very important for an other criterion) to the detriment of an other extension with more well-balanced utility functions. Other aggregation functions, such as the Choquet intergal (Grabisch 2006), allow to solve this problem by handling the interaction between criteria. We will test this aggregation function.

Example of weight function calculus We define four defaults: $\{D_1, D_2, D_3, D_4\}$, and two coefficents: the submarine safety C_{safety}, and the efficiency on the submarine mission $C_{efficiency}$.
We attribute values to each defaults:

Defaults	C_{safety}	$C_{efficiency}$
D_1	500	600
D_2	10	800
D_3	1000	900
D_4	50	60

We suppose we have to choose between two extensions: $E_1 = \{D_1, D_4\}$ and $E_2 = \{D_2, D_3, D_4\}$.
We calculate the extensions scores:

Scores	E_1	E_2
$Score_{safety}$	550	1060
$Score_{efficiency}$	660	1760

We calculate the utility functions:

μ	E_1	E_2
μ_{safety}	$\frac{550}{1610}$	$\frac{1060}{1610}$
$\mu_{efficiency}$	$\frac{660}{2420}$	$\frac{1760}{2420}$

Finally, we have the weight functions:
$P(E_1) = \beta_{safety} * \mu_{safety}(E_1) + \beta_{efficiency} * \mu_{efficiency}(E_1)$
$P(E_2) = \beta_{safety} * \mu_{safety}(E_2) + \beta_{efficiency} * \mu_{efficiency}(E_2)$

If the officer prefers to favor the safety of his submarine rather than the mission efficiency, we can define: $\beta_{safety} = 0.6$ and $\beta_{efficiency} = 0.4$. With this example, we obtain: $P(E_1) = 0.31$ and $P(E_2) = 0.69$.

6.3 Random extension choice

In a tactical situation, the decision of an operator is a key aspect, which can provide unexpected reactions. In our study case, the choice hasn't to be always determinist. Moreover, with deterministic reactions, it is easy for the opposing submarine to guess these reactions (principle 5). We need a choice method, which handles these unexpected reactions.
For these reasons, we don't choose always the extension

with the maximum weight function. We prefer to introduce an additional part of uncertainty with a random choice. However, this random choice must be coherent with the principles which guide the officers decision, and with the extension selection principles defined previously.

Random choice The random choice is based on a random sampling: we have p extensions: $E_1, \ldots, E_p$, and the respective weight functions: $P(E_1), \ldots, P(E_p)$, such as $\sum_{i=1}^{p} P(E_i) = 1$. Each weight function $P(E_i)$ is the probability for the extension to be chosen.
Example 1: We have to choose between two extensions:

- E_1 with the weight function $P(E_1) = 0.1$.
- E_2 with the weight function $P(E_2) = 0.9$.

With this random choice, we manage the choice between several extensions (principle 3).

Correction on the extension weight function With this random sampling, we have problems to solve. The random choice is realist if we have to choose between extensions with close weight.
However, if we have an extension with a very important weight, we want to choose this one (example 1).
We have the same problem in the following example (example 2). The system proposes six extensions: five extensions with the same weight function 0.1 and one with the weight function equal to 0.5. The random sampling gives as much chances to be chosen to the five extensions with the weight function equal to 0.1: $5 * 0.1 = 0.5$, as to the extension with the weight function equal to 0.5. In a such case, it seems more natural to choose the extension with the weight function equal to 0.5.
We have to modify the weight function, in order to give a more important weight to the important extension, and less important weights to the others. In this purpose, we apply a correction to the weight function: the power function $f(x) = x^k$, with $k > 1$.
The correction is applied as follows:

- We have to choose between p extensions: $E_1, \ldots, E_p$, and the respective weight functions: $P(E_1), \ldots, P(E_p)$, such as $\sum_{i=1}^{p} P(E_i) = 1$.
- We apply the power function $f(x) = x^k$, with $k > 1$: $P(E_1)^k, \ldots, P(E_p)^k$. The more k will be important, the more the extensions with small weights will be minimized.
- The sum of the weight functions must be equal to one: we divide with the sum of the extensions weights $\frac{P(E_j)^k}{\sum_{i=1}^{p} P(E_i)^k}$.

This correction gives more importance to the extensions with high weight function, and less importance to the others. For the moment, we don't fix the value of the power k, we want to test different values.
Let us apply this correction on the example 2. We take the power function $f(x) = x^2$. The weight function 0.1 becomes $0.1^2 = 0.01$ and the weight function 0.5 becomes $0.5^2 = 0.25$.
We want the sum of the weight functions equal to 1. The sum of the weight functions with correction is $\sum_{i=1}^{6} P(E_i) = 0.3$.
We obtain:

- Five extensions with the weight function $\frac{0.1^2}{0.3} = 0.04$.
- One extension with the weight function $\frac{0.5^2}{0.3} = 0.8$.

With the correction, we give more chances to the extension with the more important weight to be chosen.

Filtering of the extensions with small weight functions To be sure to choose the most interesting extension and let the others aside (principle 1), we eliminate the extension with very small weight functions. We fix a threshold: the extensions with a weight function smaller than this threshold are removed.

6.4 Respect for minimal change

To respect the minimal change (principle 4), we define a general rule of minimal change. In this purpose, we remember the submarine behavior (random research trajectory, collision avoidance, ...) at the time t. We call this behavior $Behavior(t)$, and the condition to be in this behavior: $Prerequisite$, which corresponds to the default prerequisite for this behavior. The general rule of minimal change is a default:

$$D_{min\_ch} = \frac{Behavior(t) \wedge Prerequisite(t) : Behavior(t+1)}{Behavior(t+1)}$$

This rule means: " If the prerequisite of the previous behavior are always true at the time t, and if it is possible to stay in this behavior at time $t + 1$, the submarine can stay in this behavior at time $t + 1$".
The preference's coefficients $C_1 \ldots C_n$ can't be too important, in order to allow new behaviors. This rule gives more chances to an action already beginning and allows also persistency (principle 4).

7. Interface with the simulator framework and results

An interface has been realized between NoMROD and the DCNS simulator framework ATANOR. ATANOR sends to NoMROD the information about the submarine at which we apply the behavior rules (course, speed, submersion, position) and about the enemy submarine (detection, position, speed).

NoMROD compiles the behavior rules, selects an extension and sends back the instruction of course, speed and submersion to the simulator framework. On the figure 1, we have an example of a run of NoMROD, interfaced with the simulator ATANOR.

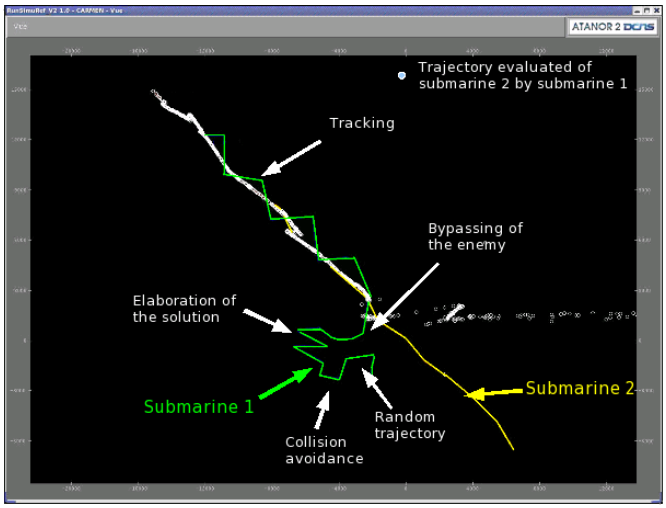

Figure 1: Detection and tracking of a submarine

We call Submarine 1 the submarine to which we apply the behavior rules of NoMROD. And we note Submarine 2 the enemy submarine, whose behavior is defined by ATANOR. The goal for the enemy submarine is to cross the patrol area without being detected. This transit is modeled with straight sections, which course is selected to match an average course.
In this scenario, the submarine 1 makes random research trajectory, because it has no detection. When it detects the enemy, the officer makes the sequence of actions: collision avoidance, elaboration of the solution, bypassing the enemy and tracking. The trajectory evaluated at the beginning is far from the real trajectory of the submarine 2. The manoeuvre of elaboration of the solution allows to obtain a better estimation of this trajectory.

8. Conclusion

The default logic allowed to formalize the behaviour rules of an officer in a submarine, by handling the incomplete, uncertain and revisable information on the environment. The formalization in default logic allows to write general rules, the defaults, without having to care about rules previously established. We have just to define the mutual exclusions between the different behaviours.
Then the obtained system compiles the available information and gives all the possibilities of actions with the extensions. To simulate the officer's choice, we defined a method to choose an extension:

- we defined weight functions for extensions, with preference's coefficients on the defaults and a weighted sum;
- we defined a method to choose an extension thanks to these weight functions with a probability technique: a random choice with corrections (power function and threshold) to be coherent with the principles which guide the officer's decisions.

We would like now to test another aggregation function: the Choquet integral, which allows to handle the interaction between the criteria. We must also work on a general method to attribute the value of the preference's coefficients on the defaults and the character coefficients: we could use learning to attribute the best values.

References

Ben Mena, S. 2000. Introduction aux méthodes multicritères d'aide la décision. *Biotchnol.Agron.Soc.envion.* 83–93.

Brewka, G. 1994. Adding priorities and specificity to default logic. In L.Pereira, and D.Pearce., eds., *Lecture Notes in Artificial Intelligence*, 247–260. European Workshop on Logics in Artificial Intelligence (JELIA'94).

Cordier, M., and Siegel, P. 1992. A temporal revision model for reasoning about world change. In *Second International Conference on Principles of Knowledge Representation and Reasoning*, 732–739.

Delgrande, J.; Shaub, T.; ; and Tompits, H. 2004. A classification and survey of preference handling approaches in non-monotonic reasoning. *Computational Intelligence* 20(2):308–334.

Ferber. 1995. *Les systèmes multi-agents. Vers une intelligence collective.* InterEditions.

Ginsberg, M., and Smith, D. 1987. Reasoning about action 1: a possible worlds approach. *Readings in Non Monotonic Reasoning*.

Grabisch, M., and Labreuch, C. 2005. Fuzzy measures and integrals in mcda. In J.Figueira; S.Greco; and Ehrgott, M., eds., *Multiple Criteria Decision Analysis*, 563–608. Springer Verlag.

Grabisch, M. 2006. L'utilisation de l'intégrale de choquet en aide multicritère la décision. *Newsletter of the European Working Group: Multicriteria Aid for Decisions*.

Nicolas, P.; Garcia, L.; and Stephan, I. 2005. Possibilistic stable models. In *International Joint Conferences on Artificial Intelligence*.

Petri, C. 1962. Fundamentals of a theory of asynchronous information flow. In *1st IFIP World Computer Congress*.

Prouty, J. 2007. *Displaying uncertainty: a comparison between submarine subject matter experts*. Ph.D. Dissertation, Naval postgraduate school, Monterey, California.

Reiter, R. 1980. A logic for default reasoning. *Artificial intelligence*.

Sombé, L. 1989. *Raisonnement sur des informations incomplètes en intelligence artificielle*. Teknea.

Toulgoat, I.; Botto, J.; De lassus, Y.; and Audoly, C. 2009. Modeling operator decision in underwater warfare performance simulations. In *Conference UDT, Cannes.*

Vincke, P. 1989. *L'aide multicritère la décision*. Ellipses.

Winslett, M. 1988. Reasoning about actions using a possible model approach. In *Proceedings of the 7th National Conference of AI*, 89–93.