Updating of a Possibilistic Knowledge Base by Crisp or Fuzzy Transition Rules

Boris Mailhé *

*ENS Cachan,
antenne de Bretagne,
Rennes, France
bmailhe@bretagne.ens-cachan.fr

Henri Prade **

** IRIT
118 route de Narbonne,
31062 Toulouse cedex, France
prade@irit.fr

Abstract

In this paper, partial knowledge about the possible transitions which can take place in a dynamical environment is represented by a set of pairs of propositional formulae, with the following intended meaning: If the first one is true then the second will be true at the next step. More generally, a certainty level representing the lower bound of a necessity measure can be associated with the second formula for modelling uncertain transitions. A possibilistic transition graph relating possible states can be deduced from these pairs and used for updating an uncertain knowledge base encoded in possibilistic logic and semantically associated to a possibility distribution. The updating of the base can be directly performed at the syntactic level from the transition pairs and the possibilistic logic formulas, in agreement with the semantics of the base and of the transitions. As there are many sets of pairs that represent the same transition graph, it is convenient to find a representative in this class that gives the easiest syntactic computations. Besides, a second type of weighted pairs of formulas is introduced for refining the representation of the available information about transitions. While the first type of pairs encodes the transitions that are not non-impossible (using necessity measures), the second type of pairs provides the transitions whose possibility is guaranteed. This constitutes a bipolar description of the possible transitions.

Key words. updating, transition, possibistic logic, bipolar knowledge, possibility theory.

Copyright © 2004, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

Introduction

A possibilistic propositional logic base is a convenient representation format for expressing a concise description of an uncertain state of knowledge, by means of classical logic formulas, which are put in different layers according to their level of certainty (viewed as a lower bound of a necessity measure). Such a basis is semantically associated with a possibility distribution that rank-orders the interpretations according to their level of plausibility (Dubois, Lang, and Prade 1994).

Using this representation framework, different information processing operations can be defined and performed equivalently, either at the syntactic level, or at the semantic level. Examples of such operations are the revision of a knowledge base when receiving an input information (possibly uncertain) (Dubois and Prade 1997), or the fusion of knowledge bases provided by different sources (Benferhat et al. 2002b), or a logical counterpart of Kalman filtering, i.e. an update operation in a dynamical environment followed by a revision (Benferhat, Dubois, and Prade 2000). It also includes the calculation of optimal decisions when a possibilistic logic base represents the available uncertain knowledge and the agent's preferences are also represented as another possibilistic logic base modelling prioritized goals (Dubois et al. 1999).

When the evolution is perfectly known, as in (Benferhat, Dubois, and Prade 2000), the transition function can be defined on all the possible states of the system. In this paper, we consider the case where the available information is imprecise, which leads to a non-functional relation on the states. The states are here the models of a logical language, so the transition function is partially specified at the syntactic level by means of conditional pairs of propositions. Obviously, computations should be performed as far as possible at the syntactic level in order to work with a more compact representation, but still in agreement with the semantic level.

After a short background on possibility theory, crisp, or fuzzy, transition functions are defined at the syntactic level, and a relation between the states is associated with this syntactic specification. The relation will then allow us to update a knowledge base at the semantic level, and then a way is provided to get the same result directly from the formulae. Then we look for the smallest possible syntactic form for expressing the transition function without loss of information and making the computation simpler. Finally, we do the same work with a finer description of the incomplete knowledge about the transition function, which distinguishes between states that can be reached for sure, from states for which nothing forbids to think that they are reachable. This is encoded by means of a bipolar knowledge base that is made of both a possibilistic knowledge base expressed in terms of necessity, and of another knowledge base stated in terms of guaranteed possibility, in order to describe the system.

Background

In possibility theory (Zadeh 1978), the basic representation tool is the notion of a possibility distribution. A possibility distribution π is a mapping from a set of interpretations Ω to a numerical scale such as the real interval [0,1], or to a, maybe discrete, linearly ordered symbolic scale. In the paper, for notational simplicity, we use [0,1] as a scale, but only in an ordinal way, which would be compatible with a re-encoding of the possibility degrees on a linearly ordered symbolic scale. Such a possibility distribution rank-orders interpretations according to their plausibility level. In particular, $\pi(\omega) = 0$ means that the interpretation is impossible; π is said normalized if $\exists\omega$ such as $\pi(\omega) = 1$; there may exist several distinct interpretations with a possibility degree equal to 1; they correspond to the most plausible interpretations. Total ignorance is represented by the possibility distribution $\pi_{\top}$ equal to 1 everywhere: $\forall\omega \in \Omega$, $\pi_{\top}(\omega) = 1$.

Given a possibility distribution π and a formula F, three measures are defined over the set of models of F, denoted by $[F] \subseteq \Omega$ (Dubois and Prade 1988).

$$\Pi(F) = \max_{\omega\in[F]} \pi(\omega),$$

is called the possibility of F. It measures how unsurprising the formula F is for the agent. $\Pi(F) = 0$ means that F is bound to be false.

$$N(F) = 1 - \Pi(\neg F) = 1 - \max_{\omega\notin[F]} \pi(\omega),$$

is called the necessity of F. Thus the necessity if F corresponds to the impossibility of 'not F'. It expresses how expected the formula F is. N(F) = 1 means that F is bound to be true. Note that the function $1 - (\cdot)$ is just a reversing map of the scale.

$$\Delta(F) = \min_{\omega\in[F]} \pi(\omega),$$

is called the guaranteed possibility of F. It expresses how much any model of the formula F is possible. Note that Δ is decreasing w. r. t. set inclusion, while Π and N are increasing (in the wide sense).

In this paper, we are interested in updating uncertain information represented by a possibilistic knowledge base

$$K = \{(F_i, \alpha_i), i = 1,n\}$$

where F_i is a propositional formula, and its certainty level α_i is such that $N(F_i) \geq \alpha_i$, where N is a necessity measure (or equivalently $\Pi(\neg F_i) \leq 1 - \alpha_i$). In possibilistic logic, the models of a knowledge base form a fuzzy set whose distribution is π. This possibility distribution π_K is the greatest solution of the set of constraints $N(F_i) \geq \alpha_i$, i = 1,n (where N is the necessity measure associated with the possibility distribution). It has been shown that (Dubois, Lang, Prade 1994):

$$\forall\omega \in \Omega,\ \pi_K(\omega) = 1 - \max_{i=1,n\ /\ \omega\notin[F_i]} \alpha_i$$
$$= \min_{i=1,n} \max(\mu_{[F_i]}(\omega), 1 - \alpha_i).$$

where $\mu_{[F_i]}$ is the characteristic function of the set $[F_i]$. Thus, $\pi_K(\omega)$ is all the smaller as ω falsifies a formula in K with a high certainty level. The distribution π_K is normalized if and only if the classical propositional base $\{F_i, i = 1,n\}$ is logically consistent.

A necessity measure can also be associated to the transition rules, associating a level of confidence to each rule if the agent has an uncertain knowledge of the evolution.

Representation of a transition function by a set of transition rules

We consider a set of propositional variables A; $\Omega \equiv P(A)$ is the universe of possible states, and L the set of formulae on A.

Case of a crisp transition function

Definition. An imprecise representation f of a transition function (from Ω to Ω) , on A is a set of pairs (H; C) called rules, where H and C are formulae on A.

Intuitively, it is to be understood as: "if H is true at a time t, then C has to be true at the time t + 1". The idea of describing pieces of evolution under the form of pairs, possibly associated with complementary terms related to uncertainty or nonmonotonicity, can be found in many authors, e.g. (Cordier and Siegel 1992, 1995). This representation is imprecise because even if the present state is completely known (H is represented by a singleton in Ω), there may remain a choice in C between several future possible states.

Given an imprecise representation f of a transition function, one can define a corresponding relation τ on Ω. Then, $\tau(\omega)$ gives all the possible states at the next time assuming that the initial state is ω, based on the available information encoded by f. As the knowledge of a set of possible states $A \subseteq \Omega$ is interpreted as an imprecise piece of knowledge, each state of A can be the real state, so each successor of each state of A is a possible state at the next time. τ is extended to sets of states using upper images, posing

$$\tau(A) = \cup_{\omega \in A} \tau(\omega).$$

Moreover, to remain consistent with our interpretation of the rules, we want to have

$$\forall\ (H; C) \in f,\ \tau([H]) \subseteq [C].$$

This imposes that

$$\forall \omega \in \Omega,\ \forall\ (H; C) \in f,\ \omega \in [H] \Rightarrow \tau(\omega) \subseteq [C].$$

So the most general possible form for τ is given by

$$\tau(\omega) = \cap_{(H;C) \in f / \omega \in [H]} [C],$$

abbreviated into $\tau(\omega) = \cap_{\omega \in [H]} [C]$.

One can then compute the inverse relation

$$\tau^{-1}(\omega) = \cap_{(H;C) \in f / \omega \in [C]^c} [H]^c.$$

where the exponent denotes set complementation. One can see that τ and τ^{-1} have similar forms, switching H and C and complementing them. Thus we can pose

$$f^{-1} = \{(\neg C; \neg H) \mid (H;C) \in f\},$$

which corresponds to the contrapositive of the rule.

Example
Let us imagine a simple coffee machine which can be working (W) or be broken, have got enough money from the user (M), have a goblet under the tap (G), or be delivering coffee (C). Then we can roughly describe some of its possible transitions by

$$f = \{(W \wedge M \wedge G; C \wedge \neg M); (W \wedge M \wedge \neg G; W \wedge M \wedge G); (\neg G; \neg C); (\neg M; \neg C)\}.$$

For instance, the first rule means that if the machine is working, has money in it, and a goblet ready, then in the next state coffee is delivered and money is spent.

Case of a fuzzy transition function

Now we consider f as a set of triples of the form (H; C; α), where α is the certainty level of the rule, i.e. "if H is true at time t, C will have a necessity degree of at least α at time t+1".Namely, f = $\{(H_i; C_i; \alpha), i=1,n\}$. By extending the previous approach, we can now build a fuzzy relation τ from this imprecise and uncertain description of a transition function:

$$\mu_\tau(\omega, \omega') = 1 - \max_{(H, C, \alpha) \in f / \omega \in [H]; \omega' \notin [C]} \alpha,$$

with the convention that 'max' taken over an empty set yields 0. Note thatthe above expression is formally similar to the one defining providing $\pi_\Sigma(\omega)$ from Σ.

Then, we use the notion of fuzzy upper image (Dubois and Prade 1992) to extend the relation:

$$\mu_{\tau(A)}(\omega') = \max_{\omega \in A} \mu_\tau(\omega, \omega').$$

This is the extension principle of fuzzy set theory (see, e.g. (Benferhat, Dubois, Prade 2000)). We can check that if a state ω' does not satisfy C, where (H; C; α) $\in$ f, then $\mu_{\tau([H])}(\omega') \le 1 - \alpha$, which is what is expected.

Fuzzy transition functions can be used to represent the uncertain effects of actions taken by an agent.

Example (continued)
For the coffee machine, we could imagine simple actions such as putting money in the machine $\{(\neg M; M; 1)\}$, taking the goblet $\{(G; \neg G; 1)\}$, or hitting the machine when it does not work correctly $\{(W; \neg W; 0.8); (\neg W; W; 0.1)\}$. The uncertainty helps representing the fact that the agent is not sure of the effects of the action he can take.

Application to updating

Now we consider a necessity-based (N-based for short) possibilistic knowledge base $\Sigma = \{(F_i, \alpha_i), i = 1,n\}$ on Ω and its induced possibility distribution π_Σ (or π for short), where (F_i, α_i) is understood as $N_\pi(F_i) \ge \alpha_i$ and N_π is associated with π_Σ. We want to build the N-based possibilistic base Σ' and its associated possibility distribution π' such that if a state is made possible by π, its successors by τ are possible in π'. This is obtained by computing the upper image of π by τ, but it can also be done at the syntactic level, drawing Σ' directly from Σ.

Crisp transition graph

Semantically, we define π' by

$$\pi'(\omega') = \max_{\omega \in \tau^{-1}(\omega')} \pi(\omega),$$

as in (Benferhat, Dubois, and Prade 2000), following the extension principle that computes $\pi(\tau^{-1}(\omega'))$. It expresses the fact that if a state ω is possible at time t, its successors ω' by τ are at least as possible at time t + 1. We can then deduce Σ' from this distribution.

Lemma 1.
Let (H_0; C_0) be a rule of f. Let □ = $N_□(H_0)$. Then $N_□'(C_0)$ ≥ □.

Proof.
$N_□(H_0)$ = □ so □□ □ $[H_0]^c$, □(□) □ 1 - □
since $N_□(H_0)$ = 1 □ max□□$[H_0]$ □(□).
Let □'□ □, such that □' □ $[C_0]$.
Then $□^{□1}$(□') □ $[H_0]^c$
since $□^{□1}$(□) = ⋂ □□$[C]^c$ $[H]^c$.
So □'(□')□ 1 □ □ because of the first inequality.
Thus $N_□'(C_0)$ ≥ □
since $N_□'(C_0)$ = 1 □ max □□$[C_0]$ □'(□').

Q.E.D.

This lemma gives a syntactic way to directly build □'. It shows that the syntactic computation given by

□' = {(C; □) | (□ H, (H; C) □ f) □ □ = $N_□$(H)}

is consistent, i.e. each formula of □' is in the belief set associated with the distribution □' (made of the formulas F such that □ □ > 0,$N_□'$(F) ≥ □ where $N_□'$ is defined from □').

However this computation is not complete as we can lose information in the process. For example, if we have f = {(A, B); (C, D)} and □ = {(A C, 1)}, then the syntactic computation gives an empty knowledge base (which corresponds to a constant distribution equal to 1 everywhere), whereas the upper image computation validates the formula (B D; 1) in □', which is what we would like to achieve at the syntactic level.

If we take the closure of f, denoted f^* and given by

f^* = {(A; B)| □([A]) □ [B]}

for each formula A, then f and f^* give the same graph. Moreover, as f^* contains all the rules a system must obey if it obeys the rules of f, it gives the same syntactic and semantic computations.

Fuzzy transition graph

Now, we compute the fuzzy upper image (Dubois and Prade 1992) of □ by □ to get □', because all the successors of possible states are possible at the next step:

□'(□') = max□□□ min{□(□), $□_□$(□, □')}.

Lemma 2. Let (H_0; C_0; □) be a rule of f^*. Let □ = $N_□(H_0)$. Then $N_□'(C_0)$ ≥ min{□, □}.

Proof.
$N_□(H_0)$ = □ so □□□$[H_0]$, □(□) □ 1 □ □ since $N_□(H_0)$ = 1 □ max □□$[H_0]$ □(□).
Let □'□ □, such that □' □ $[C_0]$.
Then □□ □ $[H_0]$, $□_□$(□; □') □ 1 □ □
since $□_□$(□, □') = 1 – max □□ [H]; □'□[C] □.
So □'(□') = max□□□ min{□(□), $□_□$(□, □')}
□ max{1 □ □, 1 □ □} = 1 □ min{□, □}
because if □ □ $[H_0]$, $□_□$(□, □')□ 1 □ □
and if □ □ $[H_0]$, □(□) □ 1 □ □.
Thus $N_□'(C_0)$ ≥ min{□, □}.

Q.E.D.

The proof is very similar to the one of lemma 1, and it allows to define the same syntactic transformation, namely:

□' = {(C, □) | □ H s.t. (H; C; □) □ f^*)
□ □ = min{□, $N_□$(C)}}.

The problem of information loss is the same as for the crisp transition function and we can solve it the same way, using the closure of f denoted f^*, given by the following formula:

f^* =
{(A; B; □) | □ = 1 □ max□□[A]; □'□ [B] $□_□$(□, □')}.

Reduction of f*

The results above give a theoretically correct framework, but the computation of the closure f^* of f is not reasonable in practice: for n propositional variables, you can have up to 2^(2^{n+1}) rules in f. Moreover, the use of f^* introduces too many information in □' which has to be simplified after that. So we have to eliminate the information that is not useful from f^*, without losing anything important. For this purpose, we have to first introduce an informativeness relation between rules.

Definition. Let (H_1; C_1) and (H_2; C_2) be two transition rules.
(H_1; C_1) is more informative than (H_2; C_2) iff
$[H_2]$ □ $[H_1]$ and $[C_1]$ □ $[C_2]$.

It means that a more informative rule only needs a weaker hypothesis to get a more precise conclusion, so it is more useful than the other, as this lemma shows:

Lemma 3. We assume that an imprecise representation f of a transition function f contains the rules (H_1; C_1) and (H_2; C_2) with (H_1; C_1) more informative than (H_2; C_2). Then f \ (H_2; C_2) and f give the same syntactic updating computation.

Proof.
Let □ □ □, such that □□ $[H_2]$. Then □□ $[H_1]$. So □(□) = □ □□[H] [C] = $[C_1]$ □ $[C_2]$ □ [C'], where C'

is given by the other hypotheses. As $[C_1]$ □ $[C_2]$, $[C_1]$ □ $[C_2] = [C_1]$. So □(□) = $[C_1]$ □ [C']. This is directly the image of □ by the relation of f \ (H_2; C_2). Moreover, the suppression of (H_2; C_2) has no influence on the states that do not satisfy H_2. So f and f \ (H_2; C_2) induce the same relation.

Q.E.D.

So we only have to keep the more informative rules of f^*. To do this, we can proceed in two steps:

- For each hypothesis, only keep the more precise conclusion. As f^*= {(H; C)| □([H]) □ [C]}, we obtain f_1 = {(H; C) | □([H]) = [C]}, where □ is the relation associated with f. At this point, one can observe that all interesting conclusions look like
  [C] = □([H]) = □ □□[H]□ (H_i;C_i) □ f/ □□[Hi] [C_i],
  which shows that all useful C can be obtained from a combination of the C_i that are conclusions of the rules of f.
- For each conclusion of f_1, only keep the less precise hypothesis to obtain
  f_2 = {((H; C) □ f1 H; C) | □ H, (H; C) □ f_1}. f_2 only contains the most informative rules since their condition part is as general as possible for a given conclusion, but f_2 is not easy to compute, especially at the syntactic level.

That is why we try to use a function close to f_2 but syntactically computable. We pose f_3 = {(I (□J H_i); I (□J C_i)) | for all the (H_i; C_i) in f}, where I and J are any arbitrary and independent sets of indices of rules in f . First, all the rules of f_3 are in f^* because if (H_1; C_1) and (H_2; C_2) are rules of f^*, (H_1 □ H_2; C_1 □ C_2) and (H_1 H_2; C_1 C_2) are also rules of f^*. Secondly, all the conclusions of f_2 can be deduced in f_3, thanks to the remark made in the first point above.

Lemma 4. Let (H'; C') be a rule of f_2. Then
(□□[H'] (□□□[Hi] H_i); □□[H'] (□□□[Hi] C_i)) is an equivalent rule, i.e.
[H'] = [□□[H'] (□□□[Hi] H_i)] and
[C'] = [□□[H'] (□□□[Hi] C_i)].

Proof.
Since C' is a conclusion of f_2, there are sets I and J_I such that C' = I □J_I C_i. The lemma is trivially verified if H' = I □J_I H_i. Let us assume it is not the case. Then, since H' is the least precise hypothesis, [H'] [I □J_I H_i]. So there is a state □ which does not belong to [I □J_I H_i] but belongs to [H']. For this state, □(□) □ [I □J_I C_i], so [I □J_I C_i] = [(I □J_I C_i) (□□□[Hi] H_i)]. The same idea applied to all the states of H' \ [I □J_I H_i] gives that [C'] = [□□[H'] (□□□[Hi] C_i)]. Moreover we trivially have that [H'] □ [□□[H'] (□□□[Hi] H_i)]. If they were not equal, there would be two states □and □' satisfying the same hypotheses, thus having the same image by □, with □ □ [H'] and □' □ [H']. Then [H'] □ {□'} would be enough to get the conclusion C' in f^*, which cannot be since H' is the least precise hypothesis that enables the conclusion C' to be true at the next time.
So [H'] = [□□[H'] (□□□[Hi] H_i)].

Q.E.D.

So we only need f_3 to keep all the information. f_3 = {(I (□J_I H_i); I (□J_I C_i))| all the (H_i; C_i) are in f}. f_3 can also be simplified because if in a knowledge base □ we know (A; □) and (B; □), we can deduce (A □ B; min{□, □}). So if we develop the rules of f_3 to put them in the form (□I' (J'I' H_i); □I' (J'I' C_i)), in fact we only need to know the elementary disjunctions. Moreover, once J'I' H_i is proved, there is no need to try to use less precise hypotheses, since they will only bring less precise conclusions. This leads to the following algorithm:

Algorithm.
Parameters: a set of transition rules f and a knowledge base □.
Variables: three sets of transition rules g, h and k, an integer i, a float □ and a Boolean b.
Result: a knowledge base □'

```
□' := empty base;
g := f;
h := empty set;
for i := 0 to (card (f) -1) do
   for each rule (H1, C1) of g do
      □ := N□(H1);
      if □ > 0 then do
         □' := □' □ {(C1, □)};
         h := h □ {(H1, C1)};
      done;
  done;
  k := empty set;
  for each rule (H, C) of f do
      for each rule (H1, C1) of g do
         if (H1,C1) is less informative than (H, C) then
do
               continue;
         done;
         b := true;
            for each rule (H2, C2) of h do
               if (H     H1, C     C1) is less informative
than (H2, C2) then do
                     b := false;
```

```
                break;
            done;
        done;
        if b then do
            k := k □ {( H    H1, C    C1)};
        done;
    done;
  done;
  g:= k;
done;
for each rule (H1, C1) of g do
   □ := N□(H1);
   if □ > 0 then do
      □' := □' □ {(C1, □)};
   done;
done;
```

At the beginning of each iteration g lists the disjunctions of i distinct rules of f that could bring new information if the hypothesis was provable from □. h lists the rules that have enabled us to add information to □', so there is no point in working with less precise rules. The test to see if a rule is more informative than another is approximated by a syntactic test (which may leave some redundancy in f3 and in □'): since all the rules we consider have the form ($_I$ H$_i$; $_I$ C$_i$), we will only test for two rules ($_I$ H$_i$; $_I$ C$_i$) and ($_J$ H$_j$; $_J$ C$_j$) if I □ J, which is weaker (from time to time we can keep a useless rule), but much easier and cheaper to do. The worst case is when no disjunction of condition part of rule in f can be proved from □. Then we have to test $2^{\text{card (f)}}$ rules. But each time we prove a disjunction of k rules of f, it enables us not to test up to ($2^{\text{card (f)} - \text{k}}$ □ 1) less precise rules.

Example (continued)
Let us consider our coffee machine and its transition rules given by f = {(W□M□G; C□□M); (W□M□□G; W□M□G); (□G; □C); (□M; □C)}. Now we assume that what we initially know is □ = {(W; 0.7); (M; 1)}. With this knowledge base we can prove no hypothesis of f, so at the end of the first iteration, f2 and □' are empty. Then we try with the disjunctions of 2 rules and this time we can prove ((W□M□G)   (W□M□□G); 0.7). So at the end of the second iteration we have
f2 = {((W□M□G)   (W□M□□G); (C□□M)   (W□M□G))} and
□' = {((C□□M)   (W□M□G), 0.7)}.
Now we try the disjunctions of 3 distinct rules, but not those which would be less precise than the one we already used. This lets only 2 combinations instead of 4. It gives nothing. As □' is not empty at this time, the disjunction of the four rules cannot be useful since it is less precise than all the rules that were already found. So we can deduce □' = {((C□□M)   (W□M□G); 0.7)}, and with the same method, repeating the application of the transition rules, □'' = {((□C)   (C□□M); 0.7)}. One transition function is enough to describe the evolution of a Markovian stationary system. If the system is not stationary, we need as many functions as there are steps in the evolution.

The function can also be used in a diagnosis problem. If we know that at the time t we have $\square_t$ = {(□C, 1)}, then we deduce $\square_{t\square 1}$ = {(□W   □M   □G,1)}. The disjunction indeed embodies the different causes that can lead to the same effects.

A less straightforward, but abstract, example illustrating the use of the algorithm is provided in the Appendix.

Bipolar representation of a knowledge base

In some situations, it is useful to distinguish between positive and negative information (Dubois, Hajek, and Prade 2000). For example, an agent may have some goals he would like to reach, some situations he would like to avoid, and between these two extremes, there are situations with respect to which he is indifferent. An N-based possibilistic knowledge base represents negative information, since it specifies an upper bound by inducing the set of interpretations that are more or less acceptable since they are not (completely impossible). Indeed N(F$_i$) ≥ $\square_i$ is equivalent to □(□F$_i$) ≤ 1 □ $\square_i$ and having a N-based possibilistic logic base □ = {(F$_i$, $\square_i$), i = 1,n} amounts to state that the interpretations that are models of the □F$_i$'s are somewhat impossible, or at least have a bounded possibility.

One can use a second knowledge base

□ ={(G$_j$, $\square_j$), j=1,m}

where formulae are associated with lower bounds of guaranteed possibility degrees □ in order to represent positive information (Benferhat et al. 2002a), i.e.,

□j = 1,m, □(G$_j$) ≥ $\square_j$.

These set of inequalities induces a lower bound on the possibility degree of each interpretation which is given by

$\square_\square$(□) = $\max_{j=1,m}$min($\square_{[G_j]}$(□), $\square_j$).

Thus, each interpretation □ will have an upper bound $\square_\square$(□), and a lower bound $\square_\square$(□) that assesses to what extent □ is guaranteed to be possible, while $\square_\square$(□) expresses to what extent □ is not known as impossible. Of course it does only make sense when the consistency condition holds

□□ □ □, $\square_\square$(□) ≥ $\square_\square$(□).

Then we need a second transition function to work draw the □-base at each step. An easy way to ensure the consistence condition is to take the same hypotheses for the two functions and to check that for each hypothesis, the conclusion for the N-base is entailed by the conclusion of the □-base.

A crisp transition rule is now interpreted as: "if H is a true at time t, then C is guaranteed to be possible at time t+1". Note that neither H nor C have to be true at any time, they only have to be possible, which will be expressed with a □measure. Another relation □ can be deduced from rules interpreted like this. Now □(□) gives all the states which are guaranteed to be possible successors of □. The knowledge of a set of possible states A is still an imprecise piece of knowledge (corresponding to the constraint N(A) = 1), but now we deal with the □–part of the transition function, i.e., a state has to be a successor of all the states of A to be surely accepted. So we can extend □ using lower images (Dubois and Prade 1992), posing

$$□(□) = □_{□□A}\, □(□).$$

With this interpretation the transition rules have the property

$$□(H; C) □ f,\ □([H]) \supseteq [C]$$

$$\text{so } □□,\ □H,\ □ □ [H] □\ □(□) \supseteq [C].$$

This time the most general possible form is given by

$$□(□) = □_{□□[H]}\, [C].$$

It is easy to see that with this interpretation, $□^{□1}(□) = □_{□□[C]}\, [H]$. So this time we can pose $f^{□1}$ = {(C; H) | (H; C) □ f}. Now the closure of f is given by $f_□$ = {(A; B)| □([A]) ⊇ [B]}, and then we have: if $(H_1; C_1)$ and $(H_2; C_2)$ belong to $f_□$, then $(H_1 □ H_2; C_1 \quad C_2)$ and $(H_1 \quad H_2; C_1 □ C_2)$ also belong to $f_□$.

A fuzzy transition function will lead to a relation □ based on lower images and □ will be regarded as a guaranteed possibility degree for C. The image of the states is given by

$$□_□(□, □') = \max_{□□[H];\, □'□[C]} □$$

and its extension to fuzzy sets of □ by

$$□_□(A)(□') = \min_{□□A} □_□(□, □').$$

This time we can see that if a state □' satisfies C, then $□_□([H])(□') \geq □$, which gives □(C) ≥ □if H was true at the precedent time. One can also compute the closure of f:

$$f_□ = \{(A; B; □) \mid □ = \min_{□□[A];\, □'□[B]} □_□(□, □')\}$$

We can use the relation □to draw a future □-knowledge base from the present N-possibilistic knowledge base. To do this we compute the lower image □' of □ by □ , which is given by

$$□'(□') = \min_{□□□^{□1}(□')} (1 □ □(□)).$$

Then we have the lemma:

Lemma 5. Let $(H_0; C_0)$ be a rule of $f_□$. If $□_□(H_0) = □$, then $□'(C_0) \geq □$.

Proof.

$□_□(H_0) = □$ so $□□□ [H_0], □(□) □ 1 □ □$ since

$□_□(H_0) = 1 □ \max_{□□[H_0]} □(□)$.

Let □'□ □, such that $□' □ [C_0]$.

Then $□^{□1}(□') \supseteq [H_0]$ since $□^{□1}(□) = □_{□□[C]}\, [H]$.

So $□□ □ □^{□1}(□'), □(□) □ 1 □ □$ because of the first inequality. 1 □ □(□) ≥ □.

□'(□') ≥ □ because

$□'(□') = \min_{□□□^{□1}(□')} (1 □ □(□))$

Thus $□_□'(C_0) \geq □$ since $□_□'(C_0) = \min_{□'□[C_0]} □'(□')$.

Q.E.D.

As for the other interpretation, we now have a syntactic way to compute $□_□'$, which is given by

$$□_□' = \{(C; □) \mid (□H, (H; C) □ f_□) □ □ = □_□(H)\}.$$

These results can be extended when using a fuzzy transition function $f_□$ to work on a □-knowledge base. The semantic computation of □' is deduced from a fuzzy lower image (Dubois and Prade 1992), using Dienes implication:

$$□'(□') = \min_{□□□} \max\{1 □ □(□), □_□(□; □')\}.$$

We finally get a counterpart of lemma 2:

Lemma 6. Let $(H_0; C_0; □_0)$ be a rule of $f_□$. Let $□ = □_□(H_0)$. Then $□_□'(C_0) \geq \min\{□, □\}$.

Proof.

$□_□(H_0) = □$ so $□□ □ [H_0], □(□) □ 1 □ □$ since

$□_□(H_0) = 1 □ \max_{□□[H_0]} □(□)$.

Let □'□ □, such that $□' □ [C_0]$.

Then $□□ □ [H_0], □_□(□; □') \geq □_0$ since

$□_□(□; □') = \max_{(H;\, C)\, /\, □□[H];\, □'□[C]} □$.

So $□'(□') = \min_{□□□} \max\{1 □ □(□), □_□(□; □')\}$

$\geq \min\{□_0, □\}$

because if $□ □ [H_0], 1 □ □(□) \geq □_0$

and if $□ □ [H_0]$, then $□_□(□; □') \geq □$

Thus $□_□'(C_0) \geq \min\{□_0; □\}$.

Q.E.D.

So we are again allowed to compute □' without working on □ and just posing the formula

$□'$ =

$\{(C; □) \mid (□ H,(H; C; □) □ f_□) □ □ = \min\{□, □_□(H)\}\}$.

Related works

Providing high level and compact descriptions of dynamical systems is an important research trends nowadays in artificial intelligence, e.g. (Boutilier, Dean, and Hanks 1999). Indeed, agents often have to react or refer to evolving environments. The paper can be related to this concern. Different problems are related to this issue such as the updating of databases, the description of future or previous states of the system or environment. Katsuno and Mendelzon (Katsuno and Mendelzon 1992) have proposed postulates that the updating of a knowledge base by a new piece of information should obey. The underlying semantics of these postulates can be understood in terms of preferred transitions in given states (Dubois, Dupin de Saint-Cyr, and Prade 1995). Here we have assumed that this transition function is partially known.

The proposed approach can be compared to the genereralized update procedure introduced by Boutilier (Boutilier 1995), and further developed in (Lang, Marquis, Williams 2001) in a more compact way by taking account logical independence relations. However several differences are worth pointing out. (Boutilier 1995) works at the semantical level only, uses a conditioning operator in the representational framework of Spohn "kappa" functions (a framework which can be related to possibility theory (Dubois and Prade 1991)), and privileges the most plausible transitions. In this paper, we work in a purely qualitative framework, both at the semantic and at the syntactic level, and we take into account all the more or less certain information pertaining to the transitions. Moreover, a refined bipolar possibilistic logic description of transitions is provided.

Concluding remarks

This paper has shown how partial knowledge about transition functions pervaded with uncertainty can be represented in a possibilistic logic style, and how the prediction and postdiction problems can be handled with such uncertain transition information from a possibilistic knowledge base describing the available information about the current state of the world. It has been shown how computation can be made directly at the syntactic level, in agreement with the possibilistic semantics, using a compact representation. Moreover these problems have also been discussed in the case of a richer, bipolar representation.

This work can be developed in various directions including extension to first order calculus, relationships with logic of actions on the one hand, and possibilistic automata on the other hand.

References

Benferhat, S.; Dubois, D.; and Prade, H. 2000. Kalman-like filtering and updating in a possibilistic setting. In *Proceedings of the 14th European Conf. on Artificial Intelligence (ECAI '2000),* (W. Horn, ed.), Berlin, 8-12.

Benferhat, S.; Dubois, D.; Kaci, S.; and Prade, H. 2002a. Bipolar representation and fusion of preferences in the possibilistic logic framework. In *Proceedings of the 8th International Conference, Principles of Knowledge Representation and Reasoning (KR 02),* Toulouse, France, 421-432. Morgan Kaufmann Publishers.

Benferhat, S.; Dubois, D.; Prade, H.; and Williams M.-A. 2002b. A practical approach to revising prioritized knowledge bases. *Studia Logica* 70:105-130

Boutilier, C. 1995. Generalized update□Belief change in dynamic settings. In *Proceedings of the 14th Inter. Joint Conf. on Artificial Intelligence (IJCAI'95)*, 1550-1556.

Boutilier, C.; Dean, T.; and Hanks, S. 1999. Decision-theoretic planning□ structural assumptions and computational leverage. *J. of Artificial Intelligence Research* 11:1-94.

Cordier, M.-O., and Siegel, P. 1992. A temporal revision model for reasoning about world change. In *Proceedings of the 3rd Int. Conf. Principles of Knowledge Representation and Reasoning* (KR'92), 732-739. Morgan Kaufmann.

Cordier, M.-O., and Siegel, P. 1995. Prioritized transitions for updates. In *Proceedings of the 3rd European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty* (ECSQARU), (C. Froidevaux, J. Kohlas, eds.), 142-151. LNCS 946, Springer Verlag.

Dubois, D.; Dupin de Saint-Cyr, F.; and Prade, H. 1995. Update postulates without inertia. In *Proceedings of the Europ. Conf. of Symbolic and Quantitative Approaches to Reasoning and Uncertainty (ECSQARU '95),* (C. Froidevaux, J. Kohlas, eds.), 162-170.

Dubois, D.; Hajek, P.; and Prade, H. 2000. Knowledge-driven versus data-driven logics. *J. of Logic, Language, and Information* 9:65-89.

Dubois, D.; Le Berre, D.; Prade, H.; and Sabbadin, R. 1999. Using possibilistic logic for modeling qualitative decision: ATMS-based algorithms. *Fundamenta Informaticae* 37:1-30.

Dubois, D.; Lang, J.; and Prade, H. 1994. Possibilistic logic. *Handbook of Logic in Artificial Intelligence and Logic Programming, Vol. 3*, (Gabbay, D.M., Hogger, C.J., Robinson, J.A et Nute, D., eds.), 439-513. Oxford University Press.

Dubois, D., and Prade, H. 1988. *Possibility Theory.* Plenum Press.

Dubois, D., and Prade, H. 1991. Epistemic entrenchment and possibilistic logic. *Artificial Intelligence J.* 50:223-239.

Dubois, D., and Prade, H. 1992. Upper and lower images of a fuzzy set induced by a fuzzy relation: application to fuzzy inference and diagnosis. *Information Sciences* 64*:* 203-232.

Dubois, D., and Prade, H. 1997. A synthetic view of belief revision with uncertain inputs in the framework of possibility theory. *Int. J. of Approximate Reasoning* 17:295-324.

Katsuno H., and Mendelzon, A.O. 1992. On the difference between updating a knowledge base and revising it. In *Proceedings of the 2nd Inter. Conf. on Principles of Knowledge Representation and Reasoning (KR'91),* (J. Allen, R. Fikes, E. Sandewall, eds.)1991, 387-394. Revised version in *Belief Revision,* (P. Gärdenfors, ed.), 1992, 301-311. Cambridge Univ. Press.

Lang, J.; Marquis, P.; and Williams M.-A. 2001. Updating epistemic states. *Proceedings of the 14th Australian Joint Conf. on Artif. Intelligence, AI 2001/ Advances in Artificial Intelligence* (M. Stumptner, D. Corbett, M. Brooks, eds.), 297-308. LNAI 2256, Springer Verlag.

Zadeh, L.A. 1978. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 1: 3-28.

Appendix - Example

Notation :

for two rules r1 = (H_1, C_1) and r2 = (H_2, C_2), we will note r1 r2 the rule (H_1 H_2, C_1 C_2). We see that r1 r2 = r2 r1.

Let us choose
f = {r1 = (B, C E), r2 = (E, F), r3 = (B C, C D), r4 = (C, D), r5 = (F, G), r6 = (G, B)}

and

□ = {(B C, 0.7), (E F G, 1), (B E G, 0.8)}.

We compute the updated base S' which is initially empty. First we try to see if any of the rules of f1 = {(B, C E), (E, F), (B C, C D), (C, D), (F, G), (G, A B)} can be applied. Only the rule r3 can be applied so at the end of this step we have S' = {(C D, 0.7)} (this is the application of the rule to update the base) and f2 = {r3}.

Then we compute the disjunctions of 2 rules we will have to deal with at the next step. The rule (r1 r1) is rejected by the first test because we want disjunctions of distinct rules. The rule (r1 r2) is accepted. The rule (r1 r3) is rejected by the second test because r3 is in f2 so it can no longer be used to build other rules. The rule (r1 r4) is accepted although it is semantically less informative than the rule r3. This is a consequence of the approximated test of information. The process goes on and we finally find:

f1 = {r1 r2 = (B E, C E F),
r1 r4 = (B C, C D E),
r1 r5 = (B F, C E G),
r1 r6 = (B G, B C E),
r2 r4 = (E C, D F),
r2 r5 = (E F, F G),
r2 r6 = (E G, B F),
r4 r5 = (C F, D G),
r4 r6 = (C G, B D),
r5 r6 = (F G, B G)}.

Thanks to the use of r3 at the first step, there are only 10 rules in f1 instead of 15. At that time we can of course prove the condition of the rule (r1 r4) (since we have already proved a formula which entails this one, but the algorithm does not know it), and there is no other rule to apply. So at the end of this step we have S' = {(C D, 0.7), (C D E, 0.7)} and f2 = {r3, r1 r4}. We see that S' has redundant formulae due to the acceptance of a useless rule in f1. Then we compute the disjunctions of 3 distinct rules that could still bring information, removing those which contain r3 or (r1 r4).

f1 = {r1 r2 r5 = (B E F, C E F G),
r1 r2 r6 = (B E G, B C E),
r1 r5 r6 = (B F G, B C E G),
r2 r4 r5 = (E C F, D F G),
r2 r4 r6 = (E F G, B F G),
r4 r5 r6 = (C F G, B D G)}

Now we only have 6 possibly useful rules whereas the total number of rules we could have at this step is 20: the more formulae we deduce, the less work we will have to work in the future. At this step there are 2 different rules that can be used so after that we get

□' = {(C D, 0.7), (C D E, 0.7), (B F G, 1), (B C E, O.8)} and f2 = {r3, r1 r4, r2 r5 r6, r1 r2 r6}.

When we try to compute the rules to try at the next step, we find that f1 is empty, which means that the process is finished (once f1 is empty, it will remain empty until the end so the system cannot evolve anymore).

On the whole process, we have tried to apply 22 rules, and the 4 rules we managed to apply enabled us to skip the 42 other tests. So this algorithm can have a cost which is much cheaper than its worst case complexity.