# Reasoning about actions under uncertainty: A possibilistic approach

Juan Carlos Nieves [a,1], Mauricio Osorio [b], Ulises Cortés [a], Francisco Caballero [c] and Antonio López-Navidad [c]

[a] *Software Department (LSI), Universitat Politècnica de Catalunya, Spain*
[b] *Fundación Universidad de las Américas - Puebla, México*
[c] *Hospital de la Santa Creu y Sant Pau, Spain*

**Abstract.** In this paper, we present an action language which is called $\mathcal{A}^{Poss}$ in order to perform reasoning about actions under uncertainty. This language is based on a possibilistc logic programming semantics which is able to deal with reasoning under uncertain information. In order to illustrate the language, we use a medical scenario.

**Keywords.** Decision making, Uncertainty, Planning, Non-Monotonic Reasoning

## 1. Introduction

It is well-known that decision making is the physician's major activity. Many researches about how doctors make decisions have been growing steadily during at least the last 60 years. The psychologist Paul Meehl pointed out that the use of mathematical methods could make better clinical decisions than unaided judgment [1]. Meehl's argument for adoption of mathematical methods in decision making was enormously controversial and still is, despite of its considerable influence. But his claim that people frequently make systematics errors in their decision making is no longer questioned [1]. Now the question is, which mathematical methods are suitable for coping medical decision making?

Let us introduce a medical scenario in the human organ transplanting context. In this scenario, there is a donor *D* which is infected by the hepatitis B virus (HBV+) and there is a recipient *R* which needs a heart transplantation and is not infected by HBV (HBV-) but his clinical situation is 0-urgency. This means that he needs a heart transplantation in the next 24 hours, otherwise he could die.

According to [2], the common criterion for transplanting an organ from a donor HBV+ suggests that any organ from a donor HBV+ *could be* considered for transplanting if and only if the recipient is HBV+. However, it is known that organs from donors HBV+ *do not always* transmit HBV infection to recipients HBV- [2]. Moreover, *some times* when the HBV infection occurs, the recipient can spontaneously clear the virus and turn

[1]Correspondence to: Software Department (LSI), Universitat Politècnica de Catalunya. C/Jordi Girona 1-3, E08034, Barcelona, Spain. E-mail: (J.C. Nieves: jcnieves@lsi.upc.edu), (M. Osorio: osoriomauri@googlemail.com), (U. Cortés: ia@lsi.upc.edu)

to HBV-. Hence, the medical question about the scenario is: is the donor's heart viable to be transplanted onto the recipient *R*? It is quite obvious that this crucial decision is vital for the recipient's survival. The main complexity of the question is that the decision is taken under uncertainty. This means that the environment in which the decision takes place is *incomplete*, *imprecise* or *uncertain*.

One of the first problems we must confront in order to develop a decision support system that could perform reasoning under uncertainty is to identify a specification language able to handle uncertain information. Moreover, it is important that our specification language must be able to capture *actions* in order to overcome a problem which is involved with uncertain information. For instance, in human organ transplanting all the recipients need a post-transplant treatment, then it is important to consider a plan of actions which will be considered after the allocation of an organ to a recipient. It is worth mentioning that each plan of actions is particular to each medical situation.

It is well-known that the most common methods for uncertain reasoning are based on probabilities. However, since medical decision making is susceptible to the evidence of the information, it is not always natural to quantify the medical knowledge in a numerical way. For instance in [3], it is pointed out that the chief disadvantages of the decision theory approach are the difficulties of obtaining reasonable estimates of probabilities and utilities for a particular analysis.

Tversky and Kahneman observed in [4] that many decisions which are made in our common life are based on beliefs concerning the likelihood of uncertain events. For instance, it was pointed out that it is common to use statements such as "I think that . . .", "chances are . . .", "it is *probable* that . . .", "it is *plausible* that . . .", *etc.*, for supporting our decisions. Usually, in this kind of statements we are appealing to our experience or commonsense for solving a problem. In the medical domain, it is not rare that a doctor supports his decisions based on his commonsense when he has uncertain information.

In [5], it was presented a specification language which is able to cope with uncertain reasoning based on a possibilistic approach. This approach permits to explicitly use labels like *possible*, *probable*, *plausible*, *etc.*, for capturing the incomplete state of a belief. However, this specification language does not provide a syntax for dealing with actions.

Since we are interested on modeling medical decision making where not only we want to capture uncertain information but also to generate plans of actions in order to support medical decision making. In this paper, we extend the specification language presented in [5] into an action language which is called $\mathcal{A}^{Poss}$. Moreover, we present a mapping from $\mathcal{A}^{Poss}$'s syntax into possibilistic programs in order to formulate planning using possibilistic answer sets.

The rest of the papers is divided as follows: In §2, it is defined the $\mathcal{A}$'s syntax. In §3, it is presented a mapping from $\mathcal{A}^{Poss}$'s syntax into possibilistic programs. In the last section, our conclusions are presented and the future work is outlined.

## 2. The language $\mathcal{A}^{Poss}$

Our language is based on the syntax of the language $\mathcal{A}$, proposed by Gelfond and Lifschits in [6]. This language has a simple English-like syntax to express the effects of actions on a world. As Baral pointed out in [7], $\mathcal{A}$ is remarkable for its simplicity and has been extended in several directions to incorporate additional features of dynamic worlds

and to facilitate elaboration tolerance. We now present an extension of $\mathcal{A}$, which we call $\mathcal{A}^{Poss}$, in order to capture incomplete states of a belief/fluent.

As in $\mathcal{A}$'s alphabet, $\mathcal{A}^{Poss}$'s alphabet consists of two nonempty disjoint sets of symbols **F** and **A**, where **F** is a set of *possibilistic fluents* and **A** is a set of actions. A possibilistic fluent $(f\ \alpha)$ is a possibilistic literal[1] where intuitively $f$ expresses a property of an item in a world and $\alpha$ expresses $f$ is certain at least to the level $\alpha$. A state $\sigma$ is a collection of possibilistic fluents. We say that a fluent $(f\ \alpha)$ holds in a state $\sigma$ if $(f\ \alpha) \in \sigma$ and a fluent literal $(\neg f\ \alpha)$ holds if $(\neg f\ \alpha) \notin \sigma$.

For example, let us go back to the medical scenario described in the introduction. In this example, let us assume that any belief about the medical situation could be quantified by *certain*, *likely*, *maybe*, *unlikely* and *false* (see Figure 1). For simplicity, we will only consider the following fluents: Clinical situation (CS), organ function (O), and infection (Inf). Then some possibilistic fluents are: $(CS(stable)\ certain)$, meaning that it is *certain* that the clinical situation of the recipient is stable, $(inf(positive)\ maybe)$, meaning that it is *probable* that the recipient could have a positive infection, $(O(delayed\_graft\_function)\ certain)$, meaning that the organ has a delay in its functions after it has been transplanted. Some actions about the medical scenario are: *transplant*, meaning that a transplantation is done, *wait*, meaning that nothing is done *w.r.t.* the clinical situation of the recipient, and *post-transplant treatment*, meaning that a post-transplant treatment is applied to the recipient. Notice that there is not a quantification of the certainty of the actions. This is because the actions are only conditioned by the status of the possibilistic fluents.

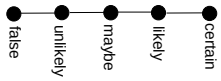


**Figure 1.** A single lattice where the following relations hold: $false \prec unlikely$, $unlikely \prec maybe$, $maybe \prec likely$, $likely \prec certain$.

Situations are representations of the history of action execution. The initial situation will be the situation where no action has been executed. It is represented by $[]$. The situation $[a_n, \ldots, a_1]$ corresponds to the history where action $a_1$ is executed in the initial situation, followed by $a_2$ and so on until $a_n$.

Now we will describe the three sub-languages of $\mathcal{A}^{Poss}$.

**Domain description language** Similar to $\mathcal{A}$, the domain description language is used to succinctly express the transition between states due to actions. This language consists of possibilistic effect propositions of the following form:

$$a \textbf{ cause } (f\ \alpha) \textbf{ if } p_1, \ldots, p_n, \neg q_{n+1}, \ldots, \neg q_r \tag{1}$$

where $a$ is an action, $(f\ \alpha)$ is a possibilistic fluent, and $p_1, \ldots, p_n, q_{n+1}, \ldots, q_r$ are atoms such that there exist the possibilistic fluents $(p_1\ \alpha_1), \ldots, (p_n\ \alpha_n)$, $(\neg q_1\ \alpha_{n+1}), \ldots, (\neg q_r\ \alpha_r)$. If $r$ is equal to 0 then the possibilistic effect proposition is written as follows :

[1] A *possibilistic literal* is a pair $l = (a, q) \in L \times Q$, where $L$ is a finite set of literals and $(Q, \leq)$ is a finite lattice. We apply the projection $*$ over $l$ as follows: $l^* = a$.

$$a \textbf{ cause } (f\ \alpha) \tag{2}$$

Also, if there is a set of possibilictic effect propositions of the form $\{a$ **cause** $(f_1\ \alpha_1), \ldots, a$ **cause** $(f_m\ \alpha_m)\}$, then this set could be written as follows:

$$a \textbf{ cause } (f_1\ \alpha_1), \ldots, (f_m\ \alpha_m) \tag{3}$$

In order to illustrate the possibilisitic effect propositions, let us consider again our medical scenario. Then a possible possibilistic effect proposition is :

$$\text{wait } \textbf{cause } (\text{CS(unstable) maybe}) \quad \textbf{if } \text{CS(stable), O(terminal\_insufficient\_function).} \tag{4}$$

The intended meaning is that if the recipient has a stable clinical condition, but he has an organ whose functions are terminal insufficient[2] and the doctor's action is *wait*, then it is *probable* that the clinical condition of the recipient could be *unstable*. The Figure 2 presents a diagram which expresses the transitions between different states where each arrow could be expressed by a possibilistic effect proposition.

**Observation language** A set of observations $Obs$ consists of possibilistic value propositions which are of the form:

$$(f\ \alpha) \textbf{ after } a_1, \ldots, a_m \tag{5}$$

where $(f\ \alpha)$ is a possibilistic fluent and $a_1, \ldots, a_m$ are actions. The intuitive reading is that if $a_1, \ldots, a_m$ would be executed in the initial situation then in the state corresponding to the situation $[a_1, \ldots, a_m]$, $(f\ \alpha)$ would hold. When $m$ is equal to 0, the possibilistic value propositions are written as follows:

$$\textbf{initially } (f\ \alpha) \tag{6}$$

In this case the intuitive meaning is that $(f\ \alpha)$ holds in the state corresponding to the initial situation. For instance, let us go back again to our medical scenario. We already know that the recipient's clinical situation is 0-urgency. Then this initial value is represented as follows:

$$\textbf{initially } (CS(0\_urgency)\ certain) \tag{7}$$

Moreover he is not infected by HBV

$$\textbf{initially } (inf(negative)\ certain) \tag{8}$$

He has an organ whose functions are terminal insufficient.

$$\textbf{initially } (O(terminal\_insufficient\_function)\ certain) \tag{9}$$

[2] We say that an organ has *terminal insufficient* functions when there exists no clinical treatment for improving the organ's functions.

Since the donor was infected by the hepatitis B virus, then it is probable that the recipient could be infected by HBV in case the heart transplantation is done. This information is represented by the following possibilistic value proposition:

$$(inf(positive)\ maybe)\ \textbf{after}\ transplant \tag{10}$$

Also we know that if the heart transplantation is not done, then the recipient could die.

$$(CS(dead)\ maybe)\ \textbf{after}\ wait \tag{11}$$

**Query Language** Queries consist of possibilistic value propositions of the form (5).

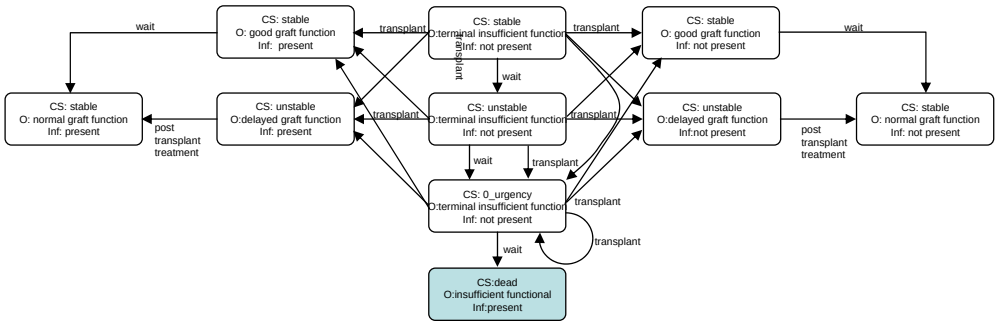


**Figure 2.** An automata of states and actions for considering infections in organ transplanting.

Executability conditions are not part of the standard $\mathcal{A}$'s syntax; however they are useful in real applications as is discussed in [7]. Then we will also introduce possibilistic executability conditions which are of the following form:

$$\textbf{executable}\ a\ \textbf{if}\ p_1, \ldots, p_n, \neg q_{n+1}, \ldots, \neg q_r \tag{12}$$

where $a$ is an action and $p_1, \ldots, p_n, q_{n+1}, \ldots, q_r$ are atoms such that there exist the possibilistic fluents $(p_1\ \alpha_1), \ldots, (p_n\ \alpha_n)$, $(\neg q_1\ \alpha_{n+1})$. In order to illustrate the possibilistic executability conditions, let us consider the following possibilistic executability condition which is in the context of our medical scenario:

$$\textbf{executable}\ tranplant\ \textbf{if}\ \neg CS(dead) \tag{13}$$

This executability conditions suggests an obvious condition which is that an organ transplant is not possible if the recipient is dead.

## 3. Reasoning about actions

In this section, we present a mapping from $\mathcal{A}^{Poss}$'s syntax into possibilistic programs in order to formulate planning using possibilistic answer sets [5]. Like in Answer Set Planning [7], we divide our encoding into two parts: the domain dependent part and the domain independent part.

We point out that we use the predicate $neg$ for denoting the negation of the fluents and we use *not* for denoting the common negation by failure used in logic programming.

### *3.1. Encoding the domain dependent part*

The domain dependent part consists of five parts, defining the domain, the executability conditions, the dynamic causal laws, the initial state, and the goal conditions.

1. The domain part defines the objects in the world, the fluents and the actions. For instance, by considering our medical scenario, a small part of the domain is:
   $certain : patient\_states(stable).$
   $certain : patient\_states(unstable).$
   $certain : patient\_states(zero\_urgency).$
   $certain : patient\_states(dead).$
   $certain : organ\_status(terminal\_insufficient\_function).$
   $certain : organ\_status(delayed\_graft\_function).$
   $certain : organ\_status(good\_graft\_function)., \ldots$
   $certain : infection\_status(present).$
   $certain : infection\_status(not\_present).$
   $certain : fluent(CS(X)) \leftarrow patient\_states(X).$
   $likely : fluent(CS(X)) \leftarrow patient\_states(X).$
   $maybe : fluent(CS(X)) \leftarrow patient\_states(X).$
   $unlikely : fluent(CS(X)) \leftarrow patient\_states(X).$
   $false : fluent(CS(X)) \leftarrow patient\_states(X).$
   $certain : fluent(O(X), Y) \leftarrow organ\_status(X)., \ldots,$
   $certain : action(transplant).$
   $certain : action(wait).$
   $\ldots$
2. The executability conditions state the executability conditions of each action. For instance, the possibilistic executability condition (13) is encoded as follows:
   $certain : exec(tranplant, neg(CS(dead))).$
   Notice that we are using the predicate $neg$ for denoting the negation of the fluents.
3. The dynamic causal laws state the effects of the actions. For instance, the possibilictic effect proposition (4) is mapped to:
   $$maybe : cause(wait, CS(unstable), TT) \leftarrow holds(CS(stable), T), holds(O(terminal\_insufficient\_function), T), time(T), time(TT), TT = T + 1.$$
4. The initial state defines the initial state by explicitly listing which fluents are true in the initial state. In our medical scenario some declaration *w.r.t.* the initial state are:

   $certain : initially(CS(0\_urgency)).$
   $certain : initially(inf(not\_present)).$
   $certain : initially(O(terminal\_insufficient\_function)).$

   It is assumed that the fluents not explicitly listed to be true are false in all the states. This means that the knowledge about all the states are assumed to be com-

plete. Therefore, in order to have our knowledge base complete in our medical scenario, we add the following possibilistic clauses:

$$certain : holds(neg(CS(Y)), T) \leftarrow holds(CS(X), T), patient\_states(X), patient\_states(Y), time(T), not\ eq(X, Y).$$
$$ceratin : holds(neg(O(Y)), T) \leftarrow holds(o(X), T), organ\_status(X), organ\_status(Y), time(T), not\ eq(X, Y).$$
$$ceratin : holds(neg(inf(Y)), T) \leftarrow holds(inf(X), T), infection\_status(X), infection\_status(Y), time(T), not\ eq(X, Y).$$

5. The goal conditions list the fluents literals that must hold in a goal state. For instance:

$$certain : finally(CS(stable)).$$
$$certain : finally(O(normal\_graft\_function)).$$

This goal suggests that we are expecting that the recipient's clinical situation must be stable and his organ graft must have normal functions.

### *3.2. Encoding the domain independent part*

This part is independent of the instance of a particular domain. Since this part defines the general rules for generating plans of actions, all of the clauses are quantified by the top of the lattice that was considered in the domain dependent part. In our medical example, it will be *certain*.

First, it is defined an exact bound or at least an upper bound of the plan lengths that we want to consider. This length makes that each possiblisitc answer set will be finite. Using this length, it is defined the predicate *time* which specifies the time points of our interest:

$$certain : time(1 \ldots length).$$

The following possibilistic clauses define when all the goal conditions are satisfied

$$certain : not\_goal(T) \leftarrow finally(X),\ not\ holds(X, T), time(T).$$
$$certain : goal(T) \leftarrow not\ not\_goal(T), time(T).$$

The following constraint eliminates possible possibilisitc answer sets where the goal is not satisfied in the last time point of interest.

$$certain : \leftarrow not\ goal(length).$$

The following possibilistic facts define when a possibilistc fluent literal is the negation of the other.

$$certain : contrary(F, neg(F)).$$
$$certain : contrary(neg(F), F).$$

The following two possibilistic clauses use the executability conditions to define when an action $A$ is executed in a time $T$.

$$certain : not\_executable(A, T) \leftarrow exec(A, F),\ not\ holds(F, T).$$
$$certain : executable(A, T) \leftarrow T < length,\ not\ not\_executable(A, T).$$

The following possibilistic clause states the fluents that are held at time point 1.

$$certain : holds(F, 1) \leftarrow initially(T).$$

The following possibilistic clause describes the change in fluent values due to the execution of an action.

$$certain : holds(F, TT) \leftarrow T < length, executable(A, T), occurs(A, T), causes(A, F, TT), TT = T + 1, time(T), time(TT).$$

The following possibilistic clause describes the fluents which do not change their values after an action is executed (frame action).

$$certain : holds(F, TT) \leftarrow contrary(F, G), T < length, holds(F, T),\ not\ holds(G, TT), \\ TT = T + 1, time(T), time(TT).$$

The following possibilistic clauses enumerate the action occurrences. They encode that in each possibilistic answer set at each time point only one of the executable actions occurred. Also, for each action that is executed in a possibilistic answer set at a time point, there is a possibilistic answer set where this action occurs at that time point.

$$certain : occurs(A, T) \leftarrow action(A),\ not\ goal(T),\ not\ not\_occurs(A, T), time(T).$$
$$certain : not\_occurs(A, T) \leftarrow action(A), action(AA), time(T), occurs(AA, T), A \neq AA.$$
$$certain :\ \leftarrow action(A), time(T), occurs(A, T),\ not\ executable(A, T).$$

### *3.3. Possibilistic answer set semantics*

In the previous subsections, we have presented a mapping from $\mathcal{A}^{Poss}$'s syntax into possibilistic programs. The possibilistic programs' semantics which permits to explicitly use labels like *possible*, *probable*, *plausible*, *etc.*, was defined in [5]. This semantics is based on the operator $\mathcal{T}$ which is inspired in partial evaluation [8] and an inference rule of possibilistic logic [9]. Due to lack of space, we do not present a formal presentation of this semantics, however we will present the general steps in order to decide if a set of possibilistic literals is a possibilistic answer set. Let $M$ be a set of possibilistic literals and $P$ be a possibilistic logic program[3].

1. The first step is to verify that $M$ must satisfy that $M^*$ is an answer set of $P^*$. $*$ is a projection which removes all the possibilistic quantifications of any set of possibilistic literals and possibilistic clauses. For instance, $\{(a, certain), (b, maybe)\}^* = \{a, b\}$ and $\{(certain : contrary(F, neg(F))), (certain : contrary(neg(F), F))\}^* = \{contrary(F, neg(F)), contrary(neg(F), F))\}$. Hence, if $M$ is a set of possibilistic literals and $P$ a possibilistic programs, then $M^*$ is a standard set of literals and $P^*$ is a standard logic program.
2. The second step is to reduce the program $P$ by $M$. The reduced program $P^M$ will be a possibilistic positive logic program, this means that $P^M$ does not have possibilistic clauses which have negative literals in their bodies[4].
3. The third and last step is to apply the operator $\mathcal{T}$ to $P^M$ in order to compute the fix point $\Pi(P^M)$. $\Pi(P^M)$ will suggest a set of possibilistic literals $M'$. If $M'$ is equal to $M$, we will say that $M$ is a possibilistic answer set of $P$.

As we can see the possibilistic answer sets have a strong relation with the common answer sets. In fact, we can observe in the point one, that $M^*$ is an answer set of $P^*$. By the moment, we do not have a possibilisitc answer set solver. However, we have used common answer set solvers for our experiments. For instance, the reader can find a small implementation of our medical scenario in (http://www.lsi.upc.edu/∼jcnieves/). This implementation is based on the answer set solver SMODELS[10]. In this implementation, we have a concrete implementation of the domain independent part of our mapping and present some mappings from effect propositions to logic program.

Due to lack of space, we can not discuss extensively the generation of plans of actions from our possibilistic action language. However let us consider again our medical scenario in order to illustrate a single plan of action.

---

[3] For more details of the possibilistic answer set semantics' definition see [5].

[4] Let $l$ be a literal. $not\ l$ is called negative literal.

First let $\Phi$ be a specification in $\mathcal{A}^{Poss}$ which describes our medical scenario. Therefore let $\Pi$ be the possibilistic logic program which we get from $\Phi$ by mapping it to possibilistic logic programs. Hence, in order to infer a possible plan of actions for our medical scenario we have to compute the answer sets of $\Pi^*$.

In order to get a plant of actions in two steps, we fix the constant *length* to 2. Remember that *length* is a constant in the encoding of the domain independent part. For the conditions that were described in this paper, the program $\Pi^*$ will have just one answer set which is:

$$\begin{aligned}\text{ASP}(\Pi^*) = \{&holds(cs(zero\_urgency), 1), holds(inf(not\_present), 1),\\ &holds(o(terminal\_insufficient_function), 1), holds(cs(stable), 2),\\ &holds(o(good\_graft\_function), 2), holds(inf(present), 2),\\ &cause(transplant, o(good\_graft_function), 2), occurs(transplant, 1)\}\end{aligned}$$ [5]

First of all, we have to remember that our initial states of our fluents are:

$$certain : initially(CS(0\_urgency)).$$
$$certain : initially(inf(not\_present)).$$
$$certain : initially(O(terminal\_insufficient\_function)).$$

This situation is reflected in ASP($\Pi^*$) by the atoms: $holds(cs(zero\_urgency), 1)$, $holds(inf(not\_present), 1)$, $holds(o(terminal\_insufficient\_function), 1)$.
Now our goal conditions are:

$$certain : finally(CS(stable)).$$
$$certain : finally(O(normal\_graft\_function)).$$

These conditions are reflected in ASP($\Pi^*$) by the atoms: $holds(o(good\_graft\_function), 2)$, $cause(wait, cs(dead), 2)$. It is clear that the goal conditions are satisfied, but which actions are applied for satisfying the goal conditions? These actions are suggested by the predicate $occurs$. Notice that $occurs(transplant, 1)$ belongs to ASP($\Pi^*$). Therefore the action *organ transplantation* is applied for satisfying the goal conditions.

We accept that this example is really simple, however it is enough for illustrating that the plans of actions are inferred directly by the the possibilistic answer sets of the program $\Pi$.

## 4. Conclusions

We are interested in developing a decision support system in the medical domain. We have observed that one of the main challenges is to identify a specification language in order to capture uncertain and incomplete information. The literature suggests that probability theory is not always a good option for supporting medical decision making [3].

In this paper, we introduce a possibilistic action language $\mathcal{A}^{Poss}$ in order to capture natural specifications from medical specialists. In order to be friendly with medical experts, $\mathcal{A}^{Poss}$ has a simple English-like syntax to express the effects of actions on a world. Moreover, we present a mapping from $\mathcal{A}^{Poss}$'s syntax into possibilistic programs in order to formulate planning using possibilistic answer sets.

[5] We are listing the atoms which are relevant for our example.

**Acknowledgements**

We are grateful to anonymous referees for their useful comments. J.C. Nieves wants to thank CONACyT for his PhD Grant. J.C. Nieves and U. Cortés were partially supported by the grant FP6-IST-002307 (ASPIC). The views expressed in this paper are not necessarily those of ASPIC consortium.

## References

[1] John Fox and Subrata Das. *Safe and Sound: Artificial Intelligence in Hazardous Applications*. AAAI Press/ The MIT Press, 2000.

[2] Antonio López-Navidad and Francisco Caballero. Extended criteria for organ acceptance: Strategies for achieving organ safety and for increasing organ pool. *Clinical Transplantation, Blackwell Munksgaard*, 17:308–324, 2003.

[3] Peter Szolovits. *Artificial Intelligence and Medicine*. Westview Press, Boulder, Colorado, 1982.

[4] Amos Tversky and Daniel Kahneman. *Judgment under uncertainty:Heuristics and biases*, chapter Judgment under uncertainty:Heuristics and biases, pages 3–20. Cambridge Univertisy Press, 1982.

[5] Juan Carlos Nieves, Mauricio Osorio, and Ulises Cortés. Semantics for possibilistic disjunctive programs. In Gerhard Brewka Chitta Baral and John Schlipf, editors, *Ninth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR-07)*, number 4483 in LNAI, pages 315–320. Springer-Verlag, 2007.

[6] Michael Gelfond and Vladimir Lifschitz. Representing Action and Change by logic programs. *Journal of Logic Programming*, 17(2,3,4):301, 323 1993.

[7] Chitta Baral. *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge University Press, Cambridge, 2003.

[8] Stefan Brass and Jürgen Dix. Semantics of (Disjunctive) Logic Programs Based on Partial Evaluation. *Journal of Logic Programming*, 38(3):167–213, 1999.

[9] Didier Dubois, Jérôme Lang, and Henri Prade. Possibilistic logic. In Dov Gabbay, Christopher J. Hogger, and J. A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming, Volume 3: Nonmonotonic Reasoning and Uncertain Reasoning*, pages 439–513. Oxford University Press, Oxford, 1994.

[10] System SMODELS. Helsinki University of Technology. http://www.tcs.hut.fi/Software/smodels/, 1995.