Possibilistic-based Argumentation: An Answer Set Programming Approach

Juan Carlos Nieves and Ulises Cortés
Universitat Politècnica de Catalunya
Software Department
c/Jordi Girona 1-3, E08034, Barcelona, Spain
Email: {jcnieves,ia}@lsi.upc.edu

Mauricio Osorio
Universidad de las Américas - Puebla
CENTIA, Sta. Catarina Mártir
Cholula, Puebla, 72820 México
Email: osoriomauri@googlemail.com

Abstract

In many fields of automated information processing it becomes crucial to consider together imprecise, uncertain or inconsistent information. Argumentation theory is a suitable framework for practical and uncertain reasoning, where arguments could support conclusions. We present a possibilistic-based argumentation approach which is based on a possibilistic disjunctive language. This specification language is able to capture incomplete information and incomplete states of a knowledge base at the same time.

1 Introduction

Many decisions that we make in our common life are based on beliefs concerning the likelihood of uncertain events. In fact, we commonly use arguments such as "I think that . . . ", "chances are . . . ", "it is *probable* that . . . ", "it is *plausible* that . . . ", *etc.*, for supporting our decisions. In this kind of statements usually we have appealed to our experience or our commonsense. It is not surprising to think that a reasoning based on these kinds of statements could reach *biased conclusions*. However, these conclusions could reflect the experience or commonsense of an expert. Pelletier and Elio pointed out in [41] that people simply have tendencies to ignore certain information because of the (evolutionary) necessity to make decisions quickly. This gives rise to "biases" in judgments concerning what they "really" want to do. A deep study of the importance of the biasses and heuristics in judgment under uncertainty is presented in the book [25].

In view of the fact that we know that a reasoning based on arguments which are quantified by relative likelihoods could capture our experience or our commonsense, the question is: how could these statements be captured by real application systems like Multi Agent Systems? For those steeped in probability, Halpern has remarked in [24] that probability has its problems. For one thing, the numbers are not always available. For another, the commitment to numbers means that any two events must be comparable in terms of their probabilities: either one event is more probable than the other, or they have equal probability. In fact, in [31], McCarthy and Hayes pointed out that attaching probabilities to a statement has some objections. For instance,

> The information necessary to assign numerical probabilities is not ordinarily available. Therefore, a formalism that required numerical probabilities would be epistemologically inadequate [31].

Now, the question is why not to use explicit labels like *possible*, *probable*, *plausible*, *etc.*, for capturing the incomplete state of a belief in a knowledge base when the numerical representations are not available or difficult to get. In fact, by considering partially ordered sets, it is possible to capture the confidence of a claim by using qualifiers like the Toulmin's famous "qualifiers"[47]. For instance, in [20] Fox and Modgil discuss the expressiveness of these qualifiers for capturing the uncertainty of medical claims.

As far as we know, there are few attempts to formalize argument-based decision making under uncertainty. Some of the most representative approaches are

- Bonet and Geffner approach [11],

- works based on Logic of Argumentation [21, 28] and

- more recently works based on the Possibilistic Logic [6, 1, 2].

We have been working in the decision making process for deciding if a human organ is viable or not for being transplanted [46, 37, 45]. Our experience suggests that in our medical domain, we require a *qualitative* theory of *default reasoning* for modeling incomplete information and a *quantitative* theory like possibilistic logic for modeling uncertain events.

The use of logic specification languages is a successful approach for encoding knowledge. In the last two decades, one of the most successful logic programming approach has been Answer Set Programming (ASP) [9]. ASP is the realization of much theoretical work on Non-monotonic Reasoning and Artificial Intelligence applications. It represents a new paradigm for logic programming that allows, using the concept of *negation as failure*, to handle problems with default knowledge and produce non-monotonic reasoning. The efficiency of the answer set solvers have allowed to increase the list of ASP's practical applications *e.g.,* planning, logical agents and Artificial Intelligence [14, 43].

In [34], a possibilistic framework for reasoning under uncertainty was proposed. This framework is a combination between ASP and possibilistic logic [16]. Possibilistic Logic is based on possibilistic theory where at the mathematical level, degrees of possibility and necessity are closely related to fuzzy sets [16]. Thanks to the natural properties of possibilistic logic and ASP, Nicolas *et al.*'s approach allows to deal with reasoning that is at the same time *non-monotonic* and *uncertain*. Nicolas *et al.*'s approach is based on the concept of *possibilistic stable model* which defines a semantics for *possibilistic normal logic programs*.

Following Nicolas *et al.*'s approach, in [39, 38, 40], an possibilistic disjunctive logic programming approach was introduced. This approach is able to deal with reasoning under *uncertainty* and *incomplete* information. It permits to use explicit labels like *certain*, *probable*, *plausible*, *etc.*, for capturing the incomplete state of a belief in a disjunctive logic program.

In this paper, we will define a possibilistic-based argumentation approach which is based on the possibilistic disjunctive language introduced in [39, 38, 40] and the possibilistic disjunctive semantics introduced in [38]. We define the concept of *possibilistic argument* which is based on possibilistic answer sets and is quantified by the levels of certainty. Based on the levels of certainty, we also show how to lead with conflicts between possibilistic arguments.

The rest of the paper is divided as follows: In §2, some basic concepts of possibilistic logic and possibilistic answer set programming are presented. In §3, our argumentation-based inference procedure is described. This means that we describe how to build *possibilistic arguments* and how to manage *conflict between possibilistic arguments*. In §4, we formalize a couple of properties *w.r.t.* our possibilistic based-argumentation approach. In §5, we discuss a little the existence of possibilistic arguments in an inconsistent possibilistic knowledge base. In §6, a small discussion of related work is presented. In the last section, we present our conclusions.

2   Background

In this section, we define some basic concepts of Possibilistic Logic and Possibilistic Answer Set Programming. We assume familiarity with basic concepts in classic logic and in the semantics of logic programs *e.g.,* interpretations, models, *etc*. A good introductory treatment of these concepts can be found in [9, 32].

2.1   Possibilistic Logic

A necessity-valued formula is a pair $(\varphi\ \alpha)$ where φ is a classical logic formula and $\alpha \in (0, 1]$ is a positive number. The pair $(\varphi\ \alpha)$ expresses that the formula φ is certain at least to the level α, *i.e.* $N(\varphi) \geq \alpha$, where N is a necessity measure modeling our possibly incomplete state knowledge [16]. α is not a probability (like it is in probability theory) but it induces a certainty (or confidence) scale. This value is determined by the expert providing the knowledge base. A necessity-valued knowledge base is then defined as a finite set (*i.e.* a conjunction) of necessity-valued formulae.

Dubois *et al.*, [16] introduced a formal system for necessity-valued logic which is based in the following axioms schemata (propositional case):

(A1) $(\varphi \rightarrow (\psi \rightarrow \varphi)\ 1)$

(A2) $((\varphi \rightarrow (\psi \rightarrow \xi)) \rightarrow ((\varphi \rightarrow \psi) \rightarrow (\varphi \rightarrow \xi))\ 1)$

(A3) $((\neg\varphi \rightarrow \neg\psi) \rightarrow ((\neg\varphi \rightarrow \psi) \rightarrow \varphi)\ 1)$

Inference rules:

(GMP) $(\varphi\ \alpha), (\varphi \rightarrow \psi\ \beta) \vdash (\psi\ min\{\alpha, \beta\})$

(S) $(\varphi\ \alpha) \vdash (\varphi\ \beta)$ if $\beta \leq \alpha$

According to Dubois *et al.*, basically we need a complete lattice in order to express the levels of uncertainty in Possibilistic Logic. Dubois *et al.*, extended the axioms schemata and the inference rules for considering partially ordered sets. We shall denote by $\vdash_{PL}$ the inference under Possibilistic Logic without paying attention if the necessity-valued formulae are using either a totally ordered set or a partially ordered set for expressing the levels of uncertainty.

The problem of inferring automatically the necessity-value of a classical formula from a possibilistic base was solved by an extended version of *resolution* for possibilistic logic (see [16] for details).

2.2   Possibilistic Answer Set Programming

In this subsection, we first introduce the standard syntax and semantics of answer set programs and after that we introduce the syntax and semantics of possibilistic answer set programs.

2.2.1 Non-Possibilistic Syntaxis

The language of a propositional logic has an alphabet consisting of

(i) proposition symbols: $p_0, p_1, ...$

(ii) connectives : $\vee, \wedge, \leftarrow, \neg, \; not, \bot$

(iii) auxiliary symbols : (,).

where $\vee, \wedge, \leftarrow$ are 2-place connectives, $\neg$, not are 1-place connective and $\bot$ is 0-place connective. The proposition symbols, $\bot$, and propositional symbols of the form $\neg p_i$ $(i \geq 0)$ stand for the indecomposable propositions, which we call *atoms*, or *atomic propositions*. Notice that there are complementary atoms such that the complement of an atom a is defined as $\widetilde{a} = \neg a$ and $\widetilde{\neg a} = a$. The negation sign $\neg$ is regarded as the so called *strong negation* by the ASP's literature and the negation not as the *negation as failure*. A literal is an atom, a, or the negation of an atom $not \; a$. Given a set of atoms $\{a_1, ..., a_n\}$, we write $not \; \{a_1, ..., a_n\}$ to denote the set of literals $\{not \; a_1, ..., not \; a_n\}$.

An extended disjunctive clause, C, is denoted:

$$a_1 \vee \ldots \vee a_m \leftarrow a_1, \ldots, a_j, not \; a_{j+1}, \ldots, not \; a_n$$

where $m \geq 0$, $n \geq 0$, each a_i is an atom. When $n = 0$ and $m > 0$ the clause is an abbreviation of $a_1 \vee \ldots \vee a_m$. When $m = 0$ the clause is an abbreviation of $\bot \leftarrow a_1, \ldots, a_n$ such that $\bot$ is the proposition symbol that always evaluates to false. Clauses of this form are called constraints (the rest, non-constraint clauses). An extended disjunctive program P is a finite set of extended disjunctive clauses. By $\mathcal{L}_P$, we denote the set of atoms in the language of P.

We denote an extended disjunctive clause C by $\mathcal{A} \leftarrow \mathcal{B}^+, \; not \; \mathcal{B}^-$, where $\mathcal{A}$ contains all the head atoms, $\mathcal{B}^+$ contains all the positive body atoms and $\mathcal{B}^-$ contains all the negative body atoms. When $\mathcal{B}^- = \emptyset$, the clause is called positive disjunctive clause. A set of positive disjunctive clauses is called a positive disjunctive logic program. When $\mathcal{A}$ is a singleton set, the clause can be regarded as a normal clause. A normal logic program is a finite set of normal clauses. Finally, when $\mathcal{A}$ is a singleton set and $\mathcal{B}^- = \emptyset$, the clause can be also regarded as a definite clause. A finite set of definite clauses is called a definite logic program.

We will manage the strong negation ($\neg$), in our logic programs, as it is done in ASP [9]. Basically, each negative atom $\neg a$ is replaced by a new atom symbol a' which does not appear in the language of the program. For instance, let P be the normal program:

$$
\begin{aligned}
a & \leftarrow q. \\
\neg q & \leftarrow r. \\
q. & \\
r. &
\end{aligned}
$$

Then replacing each negative atom by a new atom symbol, we will have:

$$
\begin{aligned}
a & \leftarrow q. \\
q' & \leftarrow r. \\
q. & \\
r. &
\end{aligned}
$$

In order not to allow inconsistent models of the non-possibilistic logic programs, usually it is added a constraint of the form $\leftarrow q, q'$. We will omit this constraint in order to allow complementary atoms in a possibilistic answer set. However the user could add this constraint without losing generality.

Given a set of proposition symbols S and a theory (a set of well-formed formulae) Γ in a logic X. $\Gamma \vdash_X S$ if and only if $\forall s \in S \; \Gamma \vdash_X s$.

2.2.2 Non-Possibilistic Semantics

The answer set semantics was first defined in terms of the so called *Gelfond-Lifschitz reduction* [22] and it is usually studied in the context of syntax dependent transformations on programs. The following definition of an answer set for general programs generalizes the definition presented in [22] and it was presented in [23]: Let P be any extended disjunctive program. For any set $S \subseteq \mathcal{L}_P$, let P^S be the positive program obtained from P by deleting

(i) each rule that has a formula $not \; a$ in its body with $a \in S$, and then

(ii) all formulae of the form $not \; a$ in the bodies of the remaining rules.

Clearly P^S does not contain not (this means that P^S is either a positive disjunctive logic program or a definite logic program), hence S is an answer set of P if and only if S is a minimal model of P^S.

In the answer set definition, we are omitting the restriction that if S has a pair of complementary atoms then $S := \mathcal{L}_P$. This means that we are allowing that an answer set could have a pair of complementary atoms. For instance, let us consider the program P:

$$a. \qquad \neg a. \qquad b.$$

then, the only answer set of this program is : $\{a, \neg a, b\}$.

It is worth mentioning that in the literature there are several forms for handling an inconsistency program. For instance, by applying the original definition [23] to P, the only stable model is: $\{a, \neg a, b, \neg b\}$. The DLV system [14] returns no models if the program is inconsistent. The Smodels system [43] has an option which permits to compute inconsistent answer sets.

2.2.3 Possibilistic Syntaxis

First of all, we start defining some relevant concepts. In all the paper, we will consider finite lattices. This convention was taken based on the assumption that in real applications we will rarely have infinite lattices for expressing the incomplete state of a knowledge base.

A *possibilistic atom* is a pair $p = (a, q) \in \mathcal{A} \times Q$, where $\mathcal{A}$ is a finite set of atoms and $(Q, \leq)$ is a lattice (since the lattice is finite then it is complete). We apply the projection $*$ as follows: $p^* = a$. Given a set of possibilistic atoms S, we define the generalization of $*$ over S as follows: $S^* = \{p^* | p \in S\}$. Given a lattice $(Q, \leq)$ and $S \subseteq Q$, $LUB(S)$ denotes the least upper bound of S and $GLB(S)$ denotes the greatest lower bound of S.

Definition 1 *Let $\mathcal{A}$ be a finite set of atoms and $(Q, \leq)$ be a lattice. Consider $\mathcal{PS} = 2^{\mathcal{A} \times Q}$ the finite set of all the possibilistic atoms sets induced by $\mathcal{A}$ and Q. $\forall A, B \in \mathcal{PS}$, we define.*

$$A \sqcap B = \{(x, GLB\{q_1, q_2\}) | (x, q_1) \in A \wedge$$
$$(x, q_2) \in B\}$$
$$A \sqcup B = \{(x, q) | (x, q) \in A \text{ and } x \notin B^*\} \cup$$
$$\{(x, q) | x \notin A^* \text{ and } (x, q) \in B\} \cup$$
$$\{(x, LUB\{q_1, q_2\}) | (x, q_1) \in A \text{ and}$$
$$(x, q_2) \in B\}.$$
$$A \sqsubseteq B \iff A^* \subseteq B^*, \text{ and } \forall x, q_1, q_2,$$
$$(x, q_1) \in A \wedge (x, q_2) \in B \text{ then } q_1 \leq q_2.$$

Now, we define the syntax of a valid possibilistic logic program. Let $(Q, \leq)$ be a lattice. A possibilistic disjunctive clause is of the form:

$$r = (\alpha : \mathcal{A} \leftarrow \mathcal{B}^+, \text{ not } \mathcal{B}^-)$$

where $\alpha \in Q$. The projection $*$ for a possibilistic clause is $r^* = \mathcal{A} \leftarrow \mathcal{B}^+, \text{ not } \mathcal{B}^-$. $n(r) = \alpha$ is a necessity degree representing the certainty level of the information described by r. A possibilistic constraint is of the form:

$$c = (TOP_Q : \leftarrow \mathcal{B}^+, \text{ not } \mathcal{B}^-)$$

where TOP_Q is the top of the lattice $(Q, \leq)$. As in possibilistic clauses, the projection $*$ for a possibilistic constraint is : $c^* = \leftarrow \mathcal{B}^+, \text{ not } \mathcal{B}^-$. A possibilistic disjunctive logic program P is a tuple of the form $\langle (Q, \leq), N \rangle$, where N is a finite set of possibilistic disjunctive clauses and possibilistic constraints. The generalization of $*$ over P is as follows: $P^* = \{r^* | r \in N\}$. Notice that P^* is an extended disjunctive program. When P^* is a normal program, P is called a possibilistic normal program. Also when P^* is a positive disjunctive program, P is called a possibilistic positive logic program. A given set of possibilistic disjunctive clauses $\{\gamma, \ldots, \gamma\}$ is also represented as $\{\gamma; \ldots; \gamma\}$ to avoid ambiguities with the use of comma in the body of the clauses.

Given a possibilistic disjunctive logic program $P = \langle (\mathcal{Q}, \leq), N \rangle$, we define the α-*cut* and the *strict* α-*cut* of P, denoted respectively by P_α and $P_{\overline{\alpha}}$, by

$P_\alpha = \langle (\mathcal{Q}, \leq), N_\alpha \rangle$ such that $N_\alpha = \{c | c \in N \text{ and } n(c) \geq \alpha\}$
$P_{\overline{\alpha}} = \langle (\mathcal{Q}, \leq), N_{\overline{\alpha}} \rangle$ such that $N_{\overline{\alpha}} = \{c | c \in N \text{ and } n(c) > \alpha\}$

In order to illustrate a possibilistic program, let us consider the following example.

Example 1 *Let S be an ordered set such that $S = \{certain, likely, maybe, unlikely, false\}$ and the following relations hold: $false \preceq unlikely$, $unlikely \preceq maybe$, $maybe \preceq likely$, $likely \preceq certain$. Also, let us consider the following predicates with their respective intended meanings:*

- elec(X,O) *: The recipient X is eligible for transplanting organ* O;

- compatible(X,O,L)*: The recipient X is histocompatible with organ O in a level L;*

- urgency(X,E) *: The recipient X has an urgency E in order to be transplanted[1];*

- temperature(X,T) *: The recipient X has a temperature* T.

Now, let us suppose that there are two possible recipients (r_1 and r_2) and we want to assign a heart. Then, let us consider the following grounded possibilistic program P:

The heart will be assigned to one of the recipients.

certain*: $elec(r_1, heart) \vee elec(r_2, heart)$.*

It is possible *that if the receptor r_1 has a high histocompatibility with the heart, then r_1 will be eligible for transplanting.*

maybe*: $elec(r_1, heart) \leftarrow compatible(r_1, heart, high)$.*

It is true *that r_1 has a high histocompatibility with the heart.*

certain*: $compatible(r_1, heart, high)$.*

It is very likely *that if the receptor r_1 is in 0-urgency, then r_1 will be eligible for transplanting.*

---

[1]In Spain, there are tree levels of urgency to consider the recipient's risk. We will suppose that 0-urgency is the highest urgency-level. This means that recipient's life is at risk.

likely: $elec(r_1, heart) \leftarrow urgency(r_1, 0\text{-}urgency)$.

It is very unlikely *that r_1 could be in 0-urgency.*

unlikely:$urgency(r_1, 0\text{-}urgency)$.

It is likely *that if r_1 has high temperature, then r_1 will not be eligible for transplanting.*

likely: $\neg elec(r_1, heart) \leftarrow temperature(r_1, high)$.

It is true *that the recipient r_1 has high temperature.*

certain: $temperature(r_1, high)$.

Let us suppose that the recipient r_2 could be eligible for transplanting if the recipient r_1 is not explicitly eligible.

maybe: $elec(r_2) \leftarrow not \neg elec(r_1, heart)$.

It is unlikely *that the recipient could be eligible if he has low histocompatibility.*

unlikely: $elec(r_2, heart) \leftarrow compatible(r2, heart, low)$.

It is certain that the recipient has low histocompatibility.

certain: $compatible(r2, heart, low)$.

By considering the program P, we want to know: who is eligible for transplanting (recipient r_1 or recipient r_2)? At the end of Section 3, we will answer this question.

2.2.4 Possibilistic Semantics

The semantics of the possibilistic disjunctive logic programs is defined in terms of a syntactic reduction which is defined as follows:

Definition 2 (Reduction P^M) *Let $P = \langle (Q, \leq), N \rangle$ be a possibilistic disjunctive logic program, M be a set of atoms. P reduced by M is the positive possibilistic disjunctive program:*

$$P^M := \{(n(r) : \mathcal{A} \cap M \leftarrow \mathcal{B}^+) | r \in N, \mathcal{A} \cap M \neq \emptyset, \\ \mathcal{B}^- \cap M = \emptyset, \mathcal{B}^+ \subseteq M \}$$

where r^* is of the form $\mathcal{A} \leftarrow \mathcal{B}^+, not\ \mathcal{B}^-$.

Notice that $(P^*)^M$ is not exactly the *Gelfond-Lifschitz reduction*. In fact, our reduction is stronger than Gelfond-Lifschitz reduction when P^* is a disjunctive program. The reduced program $(P^{M^*})^*$ always is either a positive disjunctive logic program or a definite logic program.

Example 2 *Let P be the possibilistic program of Example 1 and let*

$$S := \{(compatible(r_2, heart, low), certain), \\ (compatible(r_1, heart, high), certain), \\ (temperature(r_1, high), certain), \\ (urgency(r_1, 0\text{-}urgency), unlikely), \\ (elec(r_1, heart), maybe), \\ (\neg elec(r_1, heart), likely), \\ (elec(r_2, heart), unlikely)\}.$$

We can see that P^{S^*} is:

certain:	$elec(r_1, heart) \vee elec(r_2, heart)$.
maybe:	$elec(r_1, heart) \leftarrow compatible(r_1, heart, high)$.
certain:	$compatible(r_1, heart, high)$.
likely:	$elec(r_1, heart) \leftarrow urgency(r_1, 0\text{-}urgency)$
unlikely:	$urgency(r_1, 0\text{-}urgency)$.
likely:	$\neg elec(r_1, heart) \leftarrow temperature(r_1, high)$.
certain:	$temperature(r_1, high)$.
unlikely:	$elec(r_2, heart) \leftarrow compatible(r2, heart, low)$.
certain:	$compatible(r2, heart, low)$.

Once a possibilistic logic program P has been reduced by a set of possibilistic atoms M, it is possible to test whether M is a possibilistic answer set of the program P by considering the following definition.

Definition 3 (Possibilistic answer set) *Let $P = \langle (Q, \leq), N \rangle$ be a possibilistic disjunctive logic program and M be a set of possibilistic atoms such that M^* is an answer set of P^*. M is a possibilistic answer set of P if and only if $P^{M^*} \vdash_{PL} M$ and $\nexists M' \in \mathcal{PS}$ such that $M' \neq M$, $P^{(M')^*} \vdash_{PL} M'$ and $M \sqsubseteq M'$.*

Let us consider the following example in order to illustrate the definition.

Example 3 *Let P be again the possibilistic program of Example 1 and S be the possibilistic set of atoms introduced in Example 2. First of all, we can see that S^* is an answer set of the extended disjunctive program P^*. Hence, in order to prove that S is a possibilistic answer set of P, we have to verify that $P^{S^*} \vdash_{PL} S$. This means that for each possibilistic atom $p \in S$, $P^{S^*} \vdash_{PL} p$.*

We can see that it is straightforward that

$$P^{S^*} \vdash_{PL} \{(compatible(r_2, heart, low), certain), \\ (compatible(r_1, heart, high), certain), \\ (temperature(r_1, high), certain), \\ (urgency(r_1, 0\text{-}urgency), unlikely)\}.$$

Now let us prove $(elec(r_1, heart), maybe)$ from P^{S^*}.

Premise from P^{S^*}
1. $compatible(r_1, heart, high) \rightarrow elec(r_1, heart)$ maybe
Premise from P^{S^*}
2. $compatible(r_1, heart, high)$certain

From 1 and 2 by GMP
3. $elec(r_1, heart)$ *maybe*

The proofs of the possibilistic atoms $(\neg elec(r_1, heart), likely)$ *and* $(elec(r_2, heart), unlikely)\}$ *are similar to the proof of the possibilistic atom* $(elec(r_1, heart), maybe)$. *Therefore, we can say that* $P^{S^*} \vdash_{PL} S$ *is true.*

It is not difficult to see that there does not exist a possibilistic set S' *such that* $P^{(S')^*} \vdash_{PL} S'$ *and* $S' \sqsubseteq S$. *Hence, we can infer that* S *is a possibilistic answer set of* P.

3   Argumentation based inference

The argumentation-based inference procedure consists of two steps: *constructing possibilistic arguments* and *managing conflict between possibilistic arguments*. Thus, we shall start by defining how to build arguments from a possibilistic program.

3.1   Building arguments

A possibilistic argument is based on possibilistic answer sets and is defined as follows:

Definition 4 (Possibilistic argument) *Let* $P = \langle (Q, \leq), N \rangle$ *be a possibilistic logic program. A possibilistic argument* Arg *w.r.t.* P *is a tuple of the form* $Arg = \langle Claim, Support, q \rangle$, *such that there is a possibilistic answer set* M *of* P *such that* $(Claim, q) \in M$ *and the following conditions hold:*

1. *$Support \subseteq P$;*

2. *$Support^M \vdash_{PL} (Claim, q)$; and*

3. *$Support$ is minimal w.r.t. set inclusion.*

$\mathcal{ARG}_P$ *gathers all the possibilistic arguments which can be constructed from* P.

Notice that $(Claim, q)$ is a possibilistic atom; hence, *Claim* is an atom and $q \in Q$. We want to point out that q is regarded as a *possibilistic qualifier* which has the objective of quantifying the level of *certainty* of the argument. $Support$ is the minimal subset of P such that $Support^M \vdash_{PL}$ $(Claim, q)$.

In order to illustrate the definition, let us consider the following example:

Example 4 *Let us consider again our possibilistic program* P *of Example 1. In Example 3, we have already seen that* P *has a possibilistic answer set which is:*

$$S := \{(compatible(r_2, heart, low), certain),$$
$$(compatible(r_1, heart, high), certain),$$
$$(temperature(r_1, high), certain),$$
$$(urgency(r_1, 0\text{-}urgency), unlikely),$$
$$(elec(r_1, heart), maybe),$$
$$(\neg elec(r_1, heart), likely),$$
$$(elec(r_2, heart), unlikely)\}.$$

In Example 3, we saw that there is a proof for the possibilistic atom $(elec(r_1, heart), maybe)$, *hence we can build a possibilistic arguments which suggests that the recipient* r_1 *is eligible for transplanting.*

$$Arg_1 =$$

$$\langle elec(r_1, heart),$$
$$\{Maybe : elec(r_1, heart) \leftarrow compatible(r_1, heart, high);$$
$$Certain : compatible(r_1, heart, high)\},$$
$$maybe \rangle$$

The intuitive reading of the argument Arg_1 *is that it is* probable *that the recipient could be eligible for transplanting because it has a high level of histocompatible with the heart.*

3.2   Managing conflict between possibilistic arguments

In the case that a rational agent's knowledge base is inconsistent, there is a possibilistic answer set M such that $\{(a, q_1), (\neg a, q_2)\} \subseteq M$. For instance, let S be the answer set of Example 3. Notice that S is an inconsistent possibilistic answer set because $(elec(r_1, heart), maybe), (\neg elec(r_1, heart), likely) \in S$. When there is an inconsistent possibilistic answer set, one can construct two possibilistic arguments of the form: $Arg_1 = \langle a, Support_1, q_1 \rangle$ and $Arg_2 = \langle \neg a, Support_2, q_2 \rangle$. This means that these arguments attack each other, then there is a conflict between them. The conflicts between possibilistic arguments are formalized by the following definitions.

Definition 5 *Let* $Arg_1, Arg_2 \in \mathcal{ARG}_P$ *such that* $Arg_1 = \langle Claim_1, Support_1, q_1 \rangle$ *and* $Arg_2 = \langle Claim_2, Support_2, q_2 \rangle$. Arg_1 *attacks* Arg_2, *if* $Claim_1 = l$ *and* $Claim_2 = \tilde{l}$.

Definition 6 *Let* $Arg_1, Arg_2 \in \mathcal{ARG}_P$. $Arg_1 = \langle Claim_1, Support_1, q_1 \rangle$ *undercuts* $Arg_2 = \langle Claim_2, Support_2, q_2 \rangle$ *if and only if* $\exists (q : l \leftarrow \mathcal{B}^+, not\ \mathcal{B}^-) \in Support_2$ *such that* $Claim_1 \in \mathcal{B}^-$.

Notice that, the concept of undercut is just over literals negated by negation as failure, this means that if Arg_1 undercuts Arg_2, then Arg_1 is attacking Arg_2's assumptions. Two arguments are compared by considering their certainty levels as follows:

Definition 7 *Let $Arg_1, Arg_2 \in \mathcal{ARG}_P$ such that $Arg_1 = \langle Claim_1, Support_1, q_1 \rangle$ and $Arg_2 = \langle Claim_2, Support_2, q_2 \rangle$. Arg_1 is preferred to Arg_2 if and only if $q_1 \geq q_2$.*

Once is identified a conflict between arguments, it is important to identify which arguments win a discussion. Then, the concept of *defeat* is defined as follows:

Definition 8 *Let $Arg_1, Arg_2 \in \mathcal{ARG}_P$ such that $Arg_1 = \langle Claim_1, Support_1, q_1 \rangle$ and $Arg_2 = \langle Claim_2, Support_2, q_2 \rangle$. Arg_1 defeats Arg_2, if Arg_1 attacks/undercuts Arg_2 and it is not the case that Arg_2 is preferred to Arg_1.*

Notice that, if Arg_1 defeats Arg_2, then Arg_1's claim has a support with more evidence/certainty that Arg_2. In order to illustrate those definitions, let us consider the following example.

Example 5 *Again let us consider Example 1. In that example, we want to decide which recipient is the best option for being transplanted his heart. We have already seen that the representation of this scenario is an inconsistent knowledge base. In fact, it has an inconsistent possibilistic answer set S. Hence, there are arguments which attack each other. Notice that for each possibilistic atom which belongs to S, there is a possibilistic argument which support that possibilistic atom.*

In order to be clear, we will only consider the arguments which are related to the predicate $elec$. Hence, we can build three arguments. The first one was introduced in Example 4.

$Arg_1 =$

$\langle elec(r_1, heart),$
$\{Maybe : elec(r_1, heart) \leftarrow compatible(r_1, heart, high);$
$Certain : compatible(r_1, heart, high)\},$
$maybe \rangle$

This argument suggests that it is probable *that the recipient r_1 could be eligible for transplanting because he has a high level of histocompatible with the heart. However, we can build another argument which attacks argument Arg_1.*

$Arg_2 =$

$\langle \neg elec(r_1, heart),$
$\{Likely : \neg elec(r_1, heart) \leftarrow temperature(r_1, high);$
$Certain : temperature(r_1, high)\},$
$Likely \rangle$

Arg_2 suggests that it is really likely that the recipient r_1 could not be eligible for transplanting because he has fever.

Notice that Arg_2 has a possibilistic quantifier which is stronger than the possibilistic quantifier of argument Arg_1. Hence, we can say that Arg_2 is preferred to Arg_1. Therefore Arg_2 defeats Arg_1. This means that r_1 is not a good candidate for being transplanted his heart.

There is a third argument which suggests that the recipient r_2 could be eligible for transplanting.

$Arg_3 =$

$\langle elec(r_2, heart),$
$\{unlikely : elec(r_2, heart) \leftarrow compatible(r2, heart, low);$
$certain : compatible(r2, heart, low)\}, unlikely \rangle$

Arg_3 suggests that the recipient r_2 is unlikely to be eligible for transplanting; however we have already known that the recipient r_1 is not a good candidate for transplanting because he has fever. Hence, we can say that r_2 is the best candidate for transplanting.

In order to formalize the managing of conflicts between possibilistic arguments, we will use Dung's approach [18] for solving the conflicts of a set of possibilistic arguments. Hence, we will define some basic concept that was introduced in [18] in terms of our possibilistic arguments.

Definition 9 (Possibilistic Argumentation Framework)
Given a possibilistic logic program, a possibilistic argumentation framework AF w.r.t. P is the tuple $AF = \langle \mathcal{ARG}_P, Attacks \rangle$, where $Attacks = attacks \cup undercuts$ such that $attacks$ contains the relations of attack between the arguments of $\mathcal{ARG}_P$ and $undercuts$ contains the relations of undercut between the arguments of $\mathcal{ARG}_P$.

Observe that essentially, we are instantiating the Dung's argumentation approach into possibilistic arguments. Remember that each possibilistic argument has as a possibilistic quantifier which was regarded as a preference level (Definition 7) for solving the conflict between two possibilistic arguments (Definition 8).

In order to illustrate the definition, let us consider the arguments of Example 5. Then $AF_{Example5}$ is the tuple $\langle \{Arg_1, Arg_2, Arg_3\}, \{(Arg_1, Arg_2), (Arg_2, Arg_1)\} \rangle$.

Once a structure for capturing the conflicts that exist within a set of possibilistic arguments is defined, the next objective is to choose a *reasonable* subset of arguments from $\mathcal{ARG}_P$. For this purpose, Dung defined a set of patterns, called argumentation semantics, that allow to evaluate a set of arguments in conflict. Some well-known Dung's argumentation semantics are defined as follows:

Definition 10 *Let $AF = \langle \mathcal{ARG}_P, Attacks \rangle$ be a possibilistic argumentation framework and $S \subseteq \mathcal{ARG}_P$.*

1. *S is a conflict-free set of arguments if there are no arguments a, b in S such that a attacks/undercuts b.*

2. *An argument $a \in \mathcal{ARG}_P$ is* acceptable *w.r.t. S if and only if for each argument $b \in \mathcal{ARG}_P$: If b attacks/undercuts a then b is defeated by an argument in S.*

3. *A conflict-free set of arguments S is* admissible *if and only if each argument in S is acceptable w.r.t. S.*

4. *A preferred extension of an argumentation framework AF is a maximal (w.r.t. inclusion) admissible set of AF.*

5. *A conflict-free set of arguments S is called a stable extension if and only if S attacks each argument which does not belong to S.*

In order to illustrate the definition, let us consider $AF_{Example5}$:

- If $S = \{Arg_2\}$, then Arg_2 is acceptable *w.r.t. S* because the only argument which attacks Arg_2 is Arg_1, but Arg_2 defeats Arg_1.

- If $S = \{Arg_1, Arg_2\}$, then S is not a conflict-free set because Arg_1 and Arg_2 attack each other. It is worth to comment that conflict-freeness is a simple but important requirement in the selection of sets of arguments. In fact, Baroni and Giocomin in [10] pointed out that conflict-freeness is the minimal requirement in the selection of coherent point of view in a conflict of arguments.

- If we consider $S = \{Arg_2, Arg_3\}$, then S is an admissible set. Observe that an admissible set intuitively represents a coherent point of view in a conflict of arguments.

- We can see that $AF_{Example5}$ has four admissible set: $S_1 = \{\}$, $S_2 = \{Arg_2\}$, $S_3 = \{Arg_3\}$, and $S_4 = \{Arg_2, Arg_3\}$. The maximal admissible set *w.r.t.* set inclusion is S_4, therefore S_4 is the only possibilistic preferred extension of $AF_{Example5}$ and also it is a stable extension.

4 Some properties of the possibilistic arguments

In this section, we will present some relevant properties *w.r.t.* possibilistic argumentation frameworks.

The first property that we want to introduce is one *w.r.t.* consistent information. It is worth to comment that in domains of high-risk, as medical domain, it is important to infer sound information. The possibilistic preferred semantics implies consistent information. This property is formalized with the following theorem:

Proposition 1 (Consistency Information) *Let $AF = \langle \mathcal{ARG}_P, Attacks \rangle$ be a possibilistic argumentation framework and $S \subseteq \mathcal{ARG}$. If S is a possibilistic preferred/stable extension, then the following condition holds: If $Cs = \{Claim | \langle Claim, Support, q \rangle \in S\}$, then Cs is a consistent set of literals.*

Proof: (sketch) This theorem follows from the fact that any preferred/stable extension is a conflict-free set. Then it is straightforward that Cs is consistent. ∎

As we have commented, our possibilistic argumentation approach is based on possibilistic logic and answer set programming. Possibilistic logic has a basic principle that:

> The strength of a conclusion is the strength of the weakest argument used in its proof.

In fact, according to Dubois and Prade [17], the contribution of possibilistic logic setting is to relate this principle (measuring the validity of an inference chain by its weakest link) to fuzzy set-based necessity measures in the framework of Zadeh's possibilistic theory.

This basic principle of possibilistic logic will give an interesting property to any possibilistic argument: The strength of a possibilistic argument $Arg = \langle Claim, Support, q \rangle$ will be the strength of the weakest possibilistic clause of $Support$.

In order to prove this property, let us introduce the following lemma:

Lemma 1 *Let P be a possibilistic logic program and $Arg = \langle Claim, Support, q \rangle$ be a possibilistic argument w.r.t. P. Then*

$$Support \subseteq P_q$$

Proof: (sketch) The result follows from the followings two observations:

1. By Definition 4, we know that $Support^M \vdash_{PL} (Claim, q)$ and $Support$ is minimal *w.r.t.* set inclusion.

2. By Proposition 11 of [16], it is true that $\Gamma \vdash_{PL} (Claim, q)$ if and only if $\Gamma_q \vdash_{PL} (Claim, q)$.

∎

By considering this lemma, we formalize the following result.

Proposition 2 (Weakest link) *Let P be a possibilistic logic program an $Arg = \langle Claim, Support, q \rangle$ be a possibilistic argument w.r.t. P. Then*

$$q = GLB\{n(r) | r \in Support\}$$

Proof: The result is direct by Lemma 1. ∎

The property of *the weakest link* has been discussed by some authors [7, 5] as an important property because it could help

- to allow an agent to compare different arguments in order to select the best one and

- to determine the acceptable arguments among the conflicting ones.

5. The Existence of Possibilistic Arguments

In §3, we defined how to build arguments from a possibilistic knowledge base based on the possibilistic answer sets of a possibilistic knowledge base. Now, a good question is:

Can we build possibilistic arguments from any possibilistic knowledge base?

The answer is: NO! The big problem is that there are possibilistic programs which have no any possibilistic answer set. For instance, let us consider the following possibilistic program P:

$$\alpha : a \leftarrow not\ a$$

This program has no any possibilistic answer set; hence, we cannot build any possibilistic argument from this possibilistic program. In fact, the existence of one clause of this form will affect all the possibilistic knowledge base in such a way that the possibilistic knowledge base will not have a possibilistic answer set; and therefore, it cannot also infer any possibilistic argument.

In [40], another approach for capturing the semantics of possibilistic disjunctive programs was defined. This approach is based on the possibilistic logic programming semantics *pstable*. Like in the possibilistic answer set semantics, the possibilistic pstable semantics is based on possibilistic logic; however it is less sensitive than the answer set semantics (in the sense of inconsistency). For instance the possibilistic logic program P has a possibilistic pstable moded which is $\{(a, \alpha)\}$. Hence by considering possibilistic pstable models instead of possibilistic answer sets in Definition 4, one can build the following possibilistic argument:

$$Arg = \langle a, \{\alpha : a \leftarrow not\ a\}, \alpha \rangle$$

Even though the possibilistic pstable semantics is less sensitive than the possibilistic answer set semantics, we can not ensure that by building possibilistic arguments under the possibilistic pstable semantics, any possibilistic knowledge base will infer possibilistic arguments. This is because there are possibilistic programs that have no possibilistic pstable as well. For instance, let us consider the following possibilistic program P_{inc} under the lattice $\langle \{0, 0.1, 0.2, \dots, 0.9, 1\}, \leq \rangle$:

$$
\begin{aligned}
0.3: &\quad a \leftarrow\ not\ b. \\
0.5: &\quad b \leftarrow\ not\ c. \\
0.6: &\quad c \leftarrow\ not\ a.
\end{aligned}
$$

Observe that P_{inc}^* has no answer sets neither pstable models; hence, P_{inc} has no possibilistic answer sets neither possibilistic pstable models. Hence, once again we cannot build any possibilistic argument from P_{inc}.

In order to restore consistency of an inconsistent possibilistic knowledge base, possibilistic logic deletes the set of possibilistic formulæ which are lower than the inconsistent degree of the inconsistent knowledge base. By considering this idea, the authors of [34] defined the concept of α-cut for possibilistic logic programs. This idea also was taken in [36] in order to restore consistency in any inconsistent possibilistic program, in the context of the possibilistic answer set and pstable semantics. For instance by applying α-cut to P_{inc}, we can reduce this program to the program $P_{ConsCutDeg(P_{inc})}$ is (see [36] for details):

$$
\begin{aligned}
0.5: &\quad b \leftarrow\ not\ c. \\
0.6: &\quad c \leftarrow\ not\ a.
\end{aligned}
$$

Observe that this program has a possibilistic answer set which is $\{(c, 0.6)\}^2$. Hence thanks to the strict α-cut of P, one is able to infer the following possibilistic argument:

$$Arg = \langle c, \{0.6 : c \leftarrow\ not\ a\}, 0.6 \rangle$$

Observe that by considering the argumentation approach presented in this paper and the natural features of the possibilistic logic programming approach presented in [39, 38, 40, 36], we are not only able to manage conflict information but also we are able to restore inconsistent knowledge based.

It is worth to comment that in [3], a possibilistic logic programming approach is defined over the many-valued *Gödel* logic. The syntax of this approach is restricted to a Horn-clause sublanguage of the many-valued *Gödel* logic; hence it is unable to capture default negation and disjunctive clauses.

6 Related work

Even though humans currently use arguments for explaining choices which are already made, or for eval-

---

[2]Remember that any possibilistic answer set is also a possibilistic pstable model.

uating potential choices, there are few proposals based on arguments for handling decision making where evidence/uncertainty plays a central role. In fact, we can point out three main approaches on this topic: Bonet and Geffner [11], works based on Logic of Argumentation (LA) [28] and more recently works based on the Possibilistic Logic (PL) [6, 1, 2]. From our point of view, all these approaches have relevant properties. However, their expressive power is quite limited for capturing *incomplete information* and *incomplete states of a knowledge base* at the same time. To find a representation of the information under evidence/uncertainty has been subject of much debate. For those steeped in probability, there is only one appropriate model for numeric uncertainty, and that is probability. But probability has its problems. For one thing, the numbers are not always available. For another, the commitment to numbers means that any two events must be comparable in terms of probability: either one event is more probable than the other, or they have equal probability [24].

In [13], it was proposed an interesting argumentation approach where the degree of belief in the argument's conclusion depends on the degree of belief in the argument's premises. This approach is so useful when the application domain permits to define *probability* links between *premises* and *conclusions* of an argument. In our definition of possibilistic argument, we also make a direct relation between the degree of belief in the argument's conclusion and the degree of belief in the argument's premises like [13]'s approach. However, our approach does not depend of probability relations. Mainly, it takes relevance when in an application domain it is difficult to define probability relations as it is the case in the medical domain. It is important to point out that sometimes when we are using a probability approach; one of the hardest parts for solving a problem is to identify the probability relations[3]. However, sometimes it is enough to have just relative likelihoods for modeling different levels of evidence/uncertainty *e.g., possible, probable, plausible, etc.*, where each relative likelihood is a possible world/class of beliefs. Also by considering a partial order $\preceq$ for ordering the relative likelihoods, we can provide a likelihood ordering for the worlds/classes of believes.

It is worth to comment that programming with uncertainty is an extensively research area. In fact, it has proceeded along various research lines of logic logic programming. An interesting historical recollection in this topic was recently presented by V. S. Subrahmanian in [44]. In this recollection he highlights some phases in the evolution of the topic from the viewpoint of a committed researcher.

Research on logic programming with uncertainty has dealt with various approaches of logic programming semantics, as well as different applications. Most of the approaches in the literature employ one of the following formalisms:

- annotated logic programming, *e.g.,* [27].

- probabilistic logic, *e.g.,* [33, 30, 26].

- fuzzy set theory, *e.g.,* [48, 42, 35].

- multi-valued logic, *e.g.,* [19, 29].

- evidence theoretic logic programming, *e.g.,* [8].

- possibilistic logic, *e.g.,* [15, 4, 3, 2, 34].

Basically, these approaches differ in the underlying notion of uncertainty and how uncertainty values, associated to clauses and facts, are managed.

To prioritize logic clauses, as it is done in our possibilistic approach, can be also regarded as a preference relation between rules. In fact, by considering the certainty degrees as *preferences*, it was defined a preference relations between arguments (Definition 7). The use of *qualitative preferences* in logic programming has been suggested by authors as G. Brewka in [12]. The Brewka's approach is also motivated from the fact that a variety of applications numerical information is hard to obtain. To have a correct understanding of the relationship between Brewka's approach and our approach requires a deep analysis.

7 Conclusions

In this paper we present an argumentation approach which has a rich specification language for encoding knowledge under imprecise or uncertain information. For instance, our approach permits to use two kinds of negation: *strong negation* and *negation as failure*, instead of only one, *strong negation*, as it is the case of all the well-known approaches. In fact, our approach is the result of the combination of a successful non-monotonic approach (Answer Set Programming [9]) and some standard ideas of the most representative argumentation approach *e.g.,* Dung's approach [18], LA's approach [28], and PL's approach [6].

We propose a possibilistic-based argumentation approach which has a rich specification language based on possibilistic answer set programming. The specification language is able to capture *incomplete information* and *incomplete states of a knowledge base* at the same time. By using a possibilistic specification language and the concept of possibilistic answer set, we define the concept of possibilistic argument. A possibilistic argument emphasizes in the evidence/uncertainty knowledge that supports its conclusion. Also by considering the evidence of each argument, it is presented a conflict managing approach between possibilistic arguments. This approach also manages the inconsistency of a possibilistic knowledge base.

---

[3]The reader could see [24], where it is presented a discussion of some of the problems to find a numerical representation for uncertainty.

Acknowledgements

References

[1] T. Alsinet, C. I. Chesñevar, L. Godo, S. Sadri, and G. R. Simari. Formalizing argumentative reasoning in a possibilistic logic programming setting with fuzzy unification. *International Journal of Approximate Reasoning*, page in press, 2008.

[2] T. Alsinet, C. I. Chesñevar, L. Godo, and G. R. Simari. A logic programming framework for possibilistic argumentation: Formalization and logical properties. *Fuzzy Sets and Systems*, 159(10):1208–1228, 2008.

[3] T. Alsinet and L. Godo. A Ccomplete Calculus for Possibilistic Logic Programming with Fuzzy Propositional Variable. In *Proceedings of the Sixteen Conference on Uncertainty in Artificial Intelligence*, 1-10, 2000. ACM Press.

[4] T. Alsinet and L. Godo. Towards an automated deduction system for first-order possibilistic logic programming with fuzzy constants. *Int. J. Intell. Syst.*, 17(9):887–924, 2002.

[5] L. Amgoud and C. Cayrol. Inferring from inconsistency in preference-based argumentation frameworks. *J. Autom. Reasoning*, 29(2):125–169, 2002.

[6] L. Amgoud and H. Prade. Using arguments for making decisions: A possibilistic logic approach. In *Proceedings of the 20th Annual Conference on Uncertainty in Artificial Intelligence (UAI-04)*, pages 10–17, Arlington, Virginia, 2004. AUAI Press.

[7] ASPIC:Project. *Deliverable D2.1: Theoretical framework for argumentation*. Argumentation Service Plarform with Integrated Components, 2004.

[8] J. F. Baldwin. Evidential support logic programming. *Fuzzy Sets and Systems*, 24(1):1–26, Octuber 1987.

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

[10] P. Baroni and M. Giacomin. On principle-based evaluation of extension-based argumentation semantics. *Artificial Intelligence.*, 171(10-15):675–700, 2007.

[11] B. Bonet and H. Geffner. Arguing for decisions: A qualitative model of decision making. In *Proceedings of the 12th Annual Conference on Uncertainty in Artificial Intelligence (UAI-96)*, pages 98–105, San Francisco, CA, 1996. Morgan Kaufmann Publishers.

[12] G. Brewka. Answer sets: From constraint programming towards qualitative optimization. In V. Lifschitz and I. Niemelä, editors, *Logic Programming and Nonmonotonic Reasoning, 7th International Conference, LPNMR 2004, Fort Lauderdale, FL, USA, January 6-8, 2004, Proceedings*, volume 2923 of *Lecture Notes in Computer Science*, pages 34–46. Springer, 2004.

[13] V. Carofiglio. Modelling argumentation with belief networks. In F. Grasso, C. Reed, and G. Carenini, editors, *CMNA IV, 4th Workshop on Computational Models of Natural Argument*, ECAI 2004, pages ?–?, Valencia, Spain, 2004.

[14] S. DLV. Vienna University of Technology. http://www.dbai.tuwien.ac.at/proj/dlv/, 1996.

[15] D. Dubois, J. Lang, and H. Prad. Towards possibilistic logic programming. In K. Furukawa, editor, *ICLP*, pages 581–595. The MIT Press, 1991.

[16] D. Dubois, J. Lang, and H. Prade. Possibilistic logic. In D. Gabbay, C. J. Hogger, and J. A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming, Volume 3: Nonmonotonic Reasoning and Uncertain Reasoning*, pages 439–513. Oxford University Press, Oxford, 1994.

[17] D. Dubois and H. Prade. Possibilistic logic: a retrospective and prospective view. *Fuzzy Sets and Systems*, 144(1):3–23, 2004.

[18] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2):321–358, 1995.

[19] M. Fitting. Bilattices and the semantics of logic programming. *Journal of Logic Programming*, 11(1&2):91–116, 1991.

[20] J. Fox and S. Modgil. From arguments to decisions: Extending the Toulmin view. In D. Hitchcock and B. Verheij, editors, *Arguing on the Toulmin model: New essays on argument analysis and evaluation*, pages 273–287. Springer Netherlands, 2006.

[21] J. Fox and S. Parsons. *Applications of Uncertainty Formalisms*, chapter Arguing about beliefs and actions, pages 266–302. Springer-Verlag, Berlin, 1998.

[22] M. Gelfond and V. Lifschitz. The Stable Model Semantics for Logic Programming. In R. Kowalski and K. Bowen, editors, *5th Conference on Logic Programming*, pages 1070–1080. MIT Press, 1988.

[23] M. Gelfond and V. Lifschitz. Classical Negation in Logic Programs and Disjunctive Databases. *New Generation Computing*, 9:365–385, 1991.

[24] J. Y. Halpern. *Reasoning about uncertainty*. The MIT Press, 2005.

[25] D. Kahneman, P. Slovic, and A. Tversky. *Judgment under uncertainty:Heuristics and biases*. Cambridge Univertisy Press, 1982.

[26] G. Kern-Isberner and T. Lukasiewicz. Combining probabilistic logic programming with the power of maximum entropy. *Artificial Intelligence*, 157(1-2):139–202, 2004.

[27] M. Kifer and V. S. Subrahmanian. Theory of generalized annotated logic programming and its applications. *J. Log. Program.*, 12(3&4):335–367, 1992.

[28] P. Krause, S. Ambler, M. Elvang-Gøransson, and J. Fox. A logic of argumentation for reasoning under uncertainty. *Computational Intelligence*, 11:113–131, 1995.

[29] L. V. S. Lakshmanan. An epistemic foundation for logic programming with uncertainty. In *FSTTCS*, pages 89–100, 1994.

[30] T. Lukasiewicz. Probabilistic logic programming. In *ECAI*, pages 388–392, 1998.

[31] J. McCarthy and P. J. Hayes. Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer and D. Michie, editors, *Machine Intelligence 4*, pages 463–502. Edinburgh University Press, 1969. reprinted in McC90.

[32] E. Mendelson. *Introduction to Mathematical Logic*. Chapman and Hall/CRC, Fourth edition 1997.

[33] R. T. Ng and V. S. Subrahmanian. Probabilistic logic programming. *Inf. Comput.*, 101(2):150–201, 1992.

[34] P. Nicolas, L. Garcia, I. Stéphan, and C. Lafèvre. Possibilistic Uncertainty Handling for Answer Set Programming. *Annal of Mathematics and Artificial Intelligence*, 47(1-2):139–181, June 2006.

[35] D. V. Nieuwenborgh, M. D. Cock, and D. Vermeir. An introduction to fuzzy answer set programming. *Ann. Math. Artif. Intell.*, 50(3-4):363–388, 2007.

[36] J. C. Nieves. *Modeling arguments and uncertain information — A non-monotonic reasoning approach*. PhD thesis, Software Department (LSI), Technical University of Catalonia, 2008.

[37] J. C. Nieves, M. Osorio, and U. Cortés. Supporting decision making in organ transplating using argumentation theory. In *LANMR 2006: 2nd Latin American Non-Monotonic Reasoning Workshop*, pages 9–14, 2006.

[38] J. C. Nieves, M. Osorio, and U. Cortés. Semantics for possibilistic disjunctive programs. In S. Costantini and R. Watson, editors, *Answer Set Programming: Advances in Theory and Implementation (ICLP-07 Workshop)*, pages 271–284, 2007.

[39] J. C. Nieves, M. Osorio, and U. Cortés. Semantics for possibilistic disjunctive programs (poster). In C. Baral, G. Brewka, and J. Schlipf, editors, *Ninth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR-07)*, number 4483 in LNAI, pages 315–320. Springer-Verlag, 2007.

[40] M. Osorio and J. C. Nieves. Pstable semantics for possibilistic logic programs. In *MICAI 2007: Advances in Artificial Intelligence, 6th Mexican International Conference on Artificial Intelligence*, number 4827 in LNAI, pages 294–304. Springer-Verlag, 2007.

[41] F. J. Pelletier and R. Elio. *Scope of Logic, Methodology and Philosophy of Science*, volume 1 of *Synthese Library*, chapter Logic and Computation, pages 137–156. Dordrecht: Kluwer Academic Press, 2002.

[42] M. Rodríguez-Artalejo and C. A. Romero-Dáz. Quantitative Logic Programming revisited. In J. Garrigue and M. Hermenegildo, editors, *9th International Symposium, FLOPS*, volume 4989 of *LNCS*, pages 272–288. Springer-Verlag Berlin Heidelberg, 2008.

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

[44] V. S. Subrahmanian. Uncertainty in logic programming. *Association for Logic Programming (ALP), Newsletter*, 20(2), May/June 2007.

[45] P. Tolchinsky, U. Cortés, S. Modgil, F. Caballero, and A. López-Navidad. Increasing Human-Organ Transplant Availability: Argumentation-Based Agent Deliberation. *IEEE Intelligent Systems: Special Issue on Intelligent Agents in Healthcare*, 21(5):30–37, November/December 2006.

[46] P. Tolchinsky, U. Cortés, J. C. Nieves, A. López-Navidad, and F. Caballero. Using arguing agents to increase the human organ pool for transplantation. In *Proc. of the Third Workshop on Agents Applied in Health Care (IJCAI 2005)*, 2005.

[47] S. E. Toulmin. *The Uses of Argument*. Cambridge University Press, 1958.

[48] M. H. van Emden. Quantitative deduction and its fixpoint theory. *Journal of Logic Programming*, 3(1):37–53, 1986.