The 27th annual workshop of the Swedish Artificial Intelligence Society (SAIS),
14–15 May 2012, Örebro, Sweden
Published by Linköping University Electronic Press
http://www.ep.liu.se/ecp_home/index.en.aspx?issue=071

Managing Inconsistent Possibilistic Knowledge Bases by An Argumentation Approach

Juan Carlos Nieves[1], Mauricio Osorio[2], Helena Lindgren[1]
[1]Department of Computer Science
Umeå University
SE-901 87, Umeå, Sweden
Email: {jcnieves,helena}@cs.umu.de

[2]Universidad de las Américas - Puebla
Dept. de Actuaría y Matemáticas
Sta. Catarina Mártir, Cholula, Puebla, 72820 México
Email: osoriomauri@gmail.com

Abstract

Inconsistent knowledge bases usually are regarded as an epistemic hell that have to be avoided at all costs. However, many times it is difficult or impossible to stay away of managing inconsistent knowledge bases. In this paper, we introduce an argumentation-based approach in order to manage inconsistent possibilistic knowledge bases. This approach will be flexible enough for managing inconsistent possibilistic models and the non-existence of possibilistic models of a possibilistic logic program.

1 Introduction

One of the purposes of argumentation theory is to provide tools for supporting decisions. For instance, argumentation theory is able to suggest arguments in favour a decision. Usually argumentation theory is adequate for supporting decisions in scenarios where the information is inconsistent and incomplete [13]. Indeed, an interesting feature of argumentation theory inference is that it is able to manage inconsistent information in a natural way.

In [8, 10, 12], a possibilistic framework for reasoning under uncertainty was proposed. This framework is a combination between Answer Set Programming (ASP) and Possibilistic Logic [4]. Possibilistic Logic is based on possibilistic theory where at the mathematical level, degrees of possibility and necessity are closely related to fuzzy sets. Thanks to the natural properties of possibilistic logic and ASP, this approach allows to deal with reasoning that is at the same time non-monotonic and uncertain. The expressiveness of this approach is rich enough for capturing sophisticated domains such as the medical domain [10] and river basin systems [1]. However, a common problem in these domains is that they are inconsistent and incomplete. It is not difficult to find scenarios where the specification of them is inconsistent. We take as example in this paper the observation made in [7] that some medical diagnoses in the dementia domain are based on incomplete and inconsistent information. For instance, in order to assess the diagnosis Alzheimer's disease the presence of episodic memory dysfunction is required according to diagnostic criteria. In the study presented in [7] this assessment was sometimes based on missing information and sometimes based on the absence of memory deficit.

In this paper, we explore an argumentation approach which consider any possibilistic logic programming semantics [8, 10, 12] for inferring information from any consistent or inconsistent possibilistic logic program.

Usually the inference in argumentation theory can be regarded by 4 steps:

1. *Argument construction*, based on a knowledge base;
2. *Argument valuation*, the assignment of a weight to arguments;
3. *Argumentation interaction*, the identification of conflicts between the arguments; and
4. *Argumentation status evaluation*, for deciding the winning or justified arguments.

In this paper essentially we will define a strategy for building possibilistic arguments by considering a possibilistic logic program. A possibilistic argument will have a standard structure (as in argumentation theory) and will be inferred by possibilistic logic programming semantics, such as the possibilistic stable semantics [8], the possibilistic answer set semantics [10] or the possibilistic pstable semantics [12], of the given possibilistic knowledge base.

In order to manage the conflict that could appear between possibilistic arguments, we will instantiate the argumentation framework structure introduce by Dung [5] in terms of possibilistic arguments. This instantiation will allow us to use argumentation semantics *e.g.,* preferred semantics, in order to infer justified arguments.

The rest of this paper is divided as follows: in Section 2, we will present some basic definitions of the logic programming syntaxis considered. In Section 3, we will present the definition of a possibilistic argument. After that, in Section 4, we formalize the interaction of possibilistic arguments. In Section 5, we define our approach for evaluating possibilistic arguments (due to lack of space all *proofs* will be omitted). Finally, in the last section, we outline our conclusions and future work.

2 Background

In this section, the syntaxis of the logic programs considered in the paper is briefly presented.

2.1 Non-Possibilistic Logic Programs

An *atom* is a propositional symbol such $a_0, a_1, \dots$ and *an extended atom* is a negated atom such as $\neg a_0, \neg a_1, \dots$. We will use the concept of atom without paying attention if it is an extended atom or not. 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* [2]. 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 [2]. 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:

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

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

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

Usually it is added a constraint of the form $\bot \leftarrow a, a'$, we will omit this constraint in order to allow complementary atoms in the models of a program. However the user could add this constraint without losing generality.

2.2 Possibilistic Logic Programs

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 (in all the paper, we will consider only finite lattices). 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. Given a finite set of atoms $\mathcal{A}$ and a lattice (Q,$\leq$), we denote by $\mathcal{PS}$ the power set $2^{\mathcal{A} \times Q}$. A possibilistic disjunctive clause is of the form:

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

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

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

where TOP_Q is the top of the lattice $(Q, \leq)$. As in possibilistic clauses, the projection $*$ for a possibilistic constraint is : $c^* = \ \leftarrow \mathcal{B}^+,\ not\ \mathcal{B}^-$. 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.

The semantics of possibilistic logic programs is captured by *possibilistic logic programming semantics.* A possibilistic logic programming semantics is a mapping from the class of all the possibilistic programs into $2^{\mathcal{PS}}$. In the literature, we can find several approaches for capturing the semantics of possibilistic logic programs [8, 10, 12].

3 Building possibilistic arguments

As we commented in Section 1, the first step in the inference process in argumentation theory is the construction of arguments. Hence, in this section, we start by defining how to build possibilistic arguments from a possibilistic program.

A possibilistic argument can be constructed by considering any possibilistic logic programming semantics *i.e.* the possibilistic stable semantics, the possibilistic answer set semantics, the possibilistic pstable semantics. Since one can consider the skeptical and credulous versions of possibilistic semantics as the possibilistic answer set semantics and the possibilistic pstable semantics, we will define two kinds of possibilistic arguments: *brave possibilistic arguments* and *cautious possibilistic arguments*[1].

Definition 1 (Possibilistic Arguments) *Let $P = \langle (\mathcal{Q}, \leq), N \rangle$ be a possibilistic logic program. A possibilistic argument Arg w.r.t. P is a tuple of the form $Arg = \langle Claim, Support, \alpha \rangle$ such that the following conditions hold:*

1. *$Support \subseteq N$.*
2. *Support is minimal w.r.t. set inclusion.*
3. *$\exists\ M \in S(Support)$ such that $(Claim, \alpha) \in M$ (in this case the possibilistic arguments Arg is called* brave. *When the existent quantified $\exists$ is changed by the for all quantified $\forall$, the possibilistic arguments Arg is called* cautious*).*

S is any possibilistic logic programming semantics. Brave-$\mathcal{ARG}_P^S$ gathers all the brave possibilistic arguments which can be constructed from P and the possibilistic logic programming semantics

[1] We adjectives of *brave* and *cautious* are motivated by the definitions of brave reasoning and cautious reasoning [6]

S. Cautious-$\mathcal{ARG}_P^S$ gathers all the cautious possibilistic arguments which can be constructed from P and the possibilistic logic programming semantics S.

In order to simplify the following definition, $\mathcal{ARG}_P^S$ will denote any set of possibilistic arguments constructed from P and based on the possibilistic logic programming semantics S. Observe that Definition 1 considers the first two steps of the inference in argumentation: *argumentation construction* and *argumentation valuation.*

Remark 1 *Before to follow on, we want to point out to the reader that to build an argument Arg with conclusion a from a program P, it does not mean that a is a correct conclusion of the whole program P. The acceptance of the conclusions will depend on the interaction of all the possibilistic arguments that one can build from P and the pattern of selection of arguments (argumentation semantics) that one uses for fixing the status of the arguments.*

4 Interaction between possibilistic arguments

Once we have defined how to build possibilistic arguments, we require to define how the possibilistic arguments will interact. In other words, we will define the cases when two possibilistic arguments will be in a conflict and then to define which arguments will be considered accepted according to a pattern of selection (argumentation semantics).

Definition 2 *Let Arg_1 and Arg_2 be two possibilistic arguments such that $Arg_1 = \langle Claim_1$, $Support_1$, $\alpha_1\rangle$ and $Arg_2 = \langle Claim_2$, $Support_2$, $\alpha_2\rangle$. We say that Arg_1 attacks Arg_2 if one of the following conditions hold:*

i) $Claim_1 = l$, $Claim_2 = \widetilde{l}$ *and* $\alpha_1 \geq \alpha_2$.

ii) $\exists(q : l \leftarrow \mathcal{B}^+,\ not\ \mathcal{B}^-) \in Support_2$ *such that* $\widetilde{Claim_1} \in \mathcal{B}^+$ *and* $\alpha_1 \geq \alpha_2$.

iii) $\exists(q : l \leftarrow \mathcal{B}^+,\ not\ \mathcal{B}^-) \in Support_2$ *and* $Claim_1 \in \mathcal{B}^-$.

Let us observe that only the first two conditions of the attack's definition have restrictions with respect to α's values. In the last condition, the attack relation is motivated by the fact that the claim of Arg_1 was assumed as false (by using *negation as failure*) in the support of Arg_2. Hence, the third condition has assigning less priority to literals which are negated by negation as failure which supporting a claim.

In Example 1 we will consider diagnosis of Alzheimer's disease (a), which is the most common dementia disease. Episodic memory dysfunction is a key finding (b), and we assume that we have two inconsistent pieces of information regarding memory. Furthermore, we know also that a large part of patients with Alzheimer's disease develop behavior and psychiatric symptoms (BPSD) (c).

Example 1 *Let P be the following possibilistic logic program:*

$0.1 : b \leftarrow \top.$	$0.8 : \neg b \leftarrow \top.$
$0.9 : \neg a \leftarrow\ not\ b.$	$0.5 : c \leftarrow a.$
	$0.5 : a \leftarrow \top.$

As we can see, one can construct at least the following four possibilistic arguments by considering any reasonable possibilistic argumentation semantics:

$$\begin{aligned}
Arg_1 &= \langle b, \{0.1 : b \leftarrow \top\}, 0.1\rangle\\
Arg_2 &= \langle \neg b, \{0.8 : \neg b \leftarrow \top\}, 0.8\rangle\\
Arg_3 &= \langle \neg a, \{0.9 : \neg a \leftarrow\ not\ b\}, 0.9\rangle\\
Arg_4 &= \langle c, \{0.5 : c \leftarrow a; 0.5 : a \leftarrow \top\}, 0.5\rangle
\end{aligned}$$

By instantiating Definition 2, one can identify the following conflicts between these possibilistic arguments:

Arg_2 attacks Arg_1 by condition i).
Arg_3 attacks Arg_4 by condition ii).
Arg_1 attacks Arg_3 by condition iii).

Observe that Arg_1 does not attack Arg_2 because Arg_1's possibilistic degree is less than Arg_2's possibilistic degree.

Once the relationship between possibilistic arguments has been identified, we need to evaluate these relationships. In our example the finding that memory dysfunction is absent attacks the finding that it is possibly present, and the claim that there is BPSD symptoms is attacked by the claim that there is no Alzheimer's disease to cause BPSD. Furthermore, the argument that there is memory dysfunction attacks the claim that there is no Alzheimer's disease.

This evaluation process will be the objective of the next section.

5 Argumentation status evaluation

The evaluation of the interaction between arguments is an important step in the inference of argumentation. In argumentation literature, there are several approaches [3, 13] in order to select coherent points of view from a set of arguments in conflict. In our case, we will follow Dung's argumentation style [5]. This approach is based on the structure called *argumentation framework*. We will generalize the concept of argumentation framework into the concept of *possibilistic argumentation framework*.

Definition 3 *Given a possibilistic logic program, a possibilistic argumentation framework AF w.r.t. P is the tuple* $AF_P^S = \langle \mathcal{ARG}_P^S, Attacks \rangle$*, where Attacks contains the relations of attack between the arguments of* $\mathcal{ARG}_P$*.*

We are essentially instantiating Dung's argumentation approach into possibilistic arguments.

Example 2 *Let us go back to Example 1. As we saw, one can construct four possibilistic arguments from P; this means that* $\mathcal{ARG}_P = \{Arg_1, Arg_2, Arg_3, Arg_4\}$ *and the relations of attacks between these arguments are:* Arg_2 *attacks* Arg_1*,* Arg_3 *attacks* Arg_4*,* Arg_1 *attacks* Arg_3*. Hence, we have the following possibilistic argumentation framework:*

$$AF_P = \langle \{Arg_1, Arg_2, Arg_3, Arg_4\}, \{(Arg_2, Arg_1), (Arg_3, Arg_4), (Arg_1, Arg_3)\} \rangle$$

Once we have instantiated a possibilistic program P into a possibilistic argumentation framework AF_P, we can apply an argumentation semantics to AF_P in order to infer information from P.

In argumentation literature, we find that the most accepted argumentation semantics are the *grounded, stable and preferred semantics* suggested by Dung in [5]. The objective of these semantics is to select subsets of arguments from a set of arguments such that these subsets of arguments represent coherent points of view from a conflict. By coherent point of view, we mean that a set of arguments inferred by an argumentation semantics must be *consistent* and moreover it must be *a defendable position* in a conflict of opinions.

In order to study a relationship between the argumentation inference and the inference of some possibilistic logic programming semantics, let us define the projection ϕ which is a relation from $\mathcal{ARG}_P$ into $2^{\mathcal{PS}}$ such that given a set of possibilistic arguments $\mathcal{ARG}$, $\phi(\mathcal{ARG}) = \{(a, \alpha) | \langle a, Support, \alpha \rangle \in \mathcal{ARG}\}$.

Example 3 *Let us consider the possibilistic logic program of Example 1 and the argumentation framework of Example 2. As we saw the possibilistic argumentation framework:*

$$AF_P = \langle \{Arg_1, Arg_2, Arg_3, Arg_4\}, \{(Arg_2, Arg_1), (Arg_3, Arg_4), (Arg_1, Arg_3)\} \rangle$$

is an instantiation of the possibilistic logic program P:

$0.1 : b \leftarrow \top.$	$0.8 : \neg b \leftarrow \top.$
$0.9 : \neg a \leftarrow \; not\ b.$	$0.5 : c \leftarrow a.$
	$0.5 : a \leftarrow \top.$

In order to illustrate the relation between the inference of argumentation theory and possibilistic logic programming semantics, let us consider the possibilistic answer set semantics defined in [9].

It is easy to see that the only possibilistic answer set S of the program P is the inconsistent set $\{(b, 0.1), (\neg b, 0.8), (c, 0.5)\}$*. On the other hand, by applying an argumentation semantics as the preferred semantics to* AF_P*, we can see that the only preferred extension PE of* AF_P *is* $\{Arg_2, Arg_3\}$*. This means that the argumentation inference based on the preferred semantics infers from P the possibilistic set* $\phi(PE) = \{(\neg b, 0.8), (\neg a, 0.9)\}$*. This means, interpreted by our medical example, that in this case there is no episodic memory dysfunction nor Alzheimer's disease present, which would be an accurate conclusion according to medical diagnostic criteria. Observe that the argumentation inference is removing the inconsistency of the set of possibilistic atoms which can be inferred from P. It is worth remember that the possibilistic argumentation framework* AF_P *can be constructed from P* *by considering the possibilistic answer set semantics.*

Now let us formalize some important properties *w.r.t.* our argumentation inference. The first property that we can identify is that whenever we apply our possibilistic-based argumentation inference based on an argumentation semantics which satisfies the basic property of conflict-freeness, we will infer consistent information. It is worth mentioning that an argumentation semantics S_{arg} satisfies the property of conflict-freeness, if given an argumentation framework AF then for all $E \in S_{arg}(AF)$, E does not contain two arguments a and b such that a attacks b.

Proposition 1 *Let S be a possibilistic logic programming semantics, S_{arg} be an argumentation semantics which satisfies the property of conflict-freeness and P be a possibilistic logic program. If $E \in S_{arg}(AF_P^S)$ then $\phi(E)$ is a consistent set of possibilistic atoms.*

Observe that this proposition is relevant whenever the information inferred from P by using the possibilistic logic programming semantics S is inconsistent. However, if the information inferred from a possibilistic logic program by a possibilistic logic programming semantics is consistent, there are cases where by considering our argumentation inference based on argumentation semantics as the preferred semantics, we are able to infer more information than applying the inference based possibilistic logic semantics (due to lack of space we do not illustrate this situation).

Proposition 2 *Let S be a possibilistic logic programming semantics, S_{arg} be the preferred semantics and P be a possibilistic logic program, If $E \in S(P)$, then there exists $E' \in S_{arg}(AF_P^S)$ such that $E = \phi(E')$.*

Another kind of inconsistency that can occur in possibilistic logic programs is the non-existent of models inferred from a possibilistic logic program by a possibilistic logic programming semantics *i.e.* the possibilistic answer set semantics and the possibilistic pstable semantics. For instance, let us consider the following program P_{inc}:

$0.3 : a \leftarrow \ not\ b.$
$0.5 : b \leftarrow \ not\ c.$
$0.6 : c \leftarrow \ not\ a.$

This program has neither possibilistic answer sets nor possibilistic pstable models. However, this kind of inconsistency can be managed by our argumentation-based inference. For instance, by considering either the possibilistic answer set semantics or the possibilistic pstable semantics, we have the following set of possibilistic arguments:

$$\mathcal{ARG}_{P_{inc}} = \{Arg_1 = \langle a, \{0.3 : a \leftarrow \ not\ b\}, 0.3\rangle, Arg_2 = \langle b, \{0.5 : b \leftarrow \ not\ c\}, 0.5\rangle, Arg_3 = \langle c, \{0.6 : c \leftarrow \ not\ a\}, 0.6\rangle \ \}$$

Hence, we can define the following possibilistic argumentation framework: $AF_{P_{inc}} = \langle\{Arg_1, Arg_2, Arg_3\}, \{(Arg_1, Arg_3), (Arg_3, Arg_2), (Arg_2, Arg_1)\}$.

Observe that if we apply an argumentation semantics based on admissible sets *e.g.,* the preferred semantics to $AF_{P_{inc}}$, we will not be able to infer any set of possibilistic atoms from P_{inc}. For these cases, we require to consider argumentation semantics which are not based on admissible sets. For instance, if we consider the argumentation semantics $MM_{Arg}^{*^r}$ presented in [11], we can see that $MM_{Arg}^{*^r}(AF_{P_{inc}})$ has three extensions: $\{\{Arg_1\}, \{Arg_2\}, \{Arg_3\}\}$. This means that $MM_{Arg}^{*^r}$ is suggesting that one can infer the following three sets of possibilistic atoms from P_{inc}:$\phi(Arg_1) = \{(a, 0.3)\}$, $\phi(Arg_2) = \{(b, 0.5)\}$ and $\phi(Arg_3) = \{(c, 0.6)\}$.

6 Concluding remarks

In this paper, we defined a possibilistic-based argumentation approach based on: i. the inference of possibilistic logic programming semantics and ii. Dung's argumentation semantics style.

This approach inherits all the expressiveness of the possibilistic logic programs [8, 10, 12] and offers some natural mechanisms for dealing with reasoning under inconsistent information. In fact, this approach does not require to apply *cuts* to an inconsistent possibilistic knowledge base, as it is done in possibilistic logic programming, in order to manage the non-existence of possibilistic models. Another interesting property of our approach is that any set of possibilistic atoms inferred by the possibilistic-based argumentation inference will be consistent.

Acknowledgement

This research is partly funded by VINNOVA (Sweden's innovation agency) and the Swedish Brain Power.

References

[1] Montse Aulinas, Juan Carlos Nieves, Manel Poch, and Ulises Cortés. Supporting Decision Making in Urban Wastewater Systems using a Knowledge-Based Approach. *Environmental Modelling and Software*, 26(5):562–572, 2011.

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

[3] Carlos Iván Chesñevar, Ana Gabriela Maguitman, and Ronald Prescott Loui. Logical models of argument. *ACM Comput. Surv.*, 32(4):337–383, 2000.

[4] 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.

[5] Phan Minh 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.

[6] Nicola Leone, Gerard Pfeifer, Wolfgang, Thomas Either, Georg Gottlob abd Simona Perri, and Francesco Scarcello. The DLV System for Knowledge Representation and Reasoning. *ACM Transaction on Computational Logic*, 7(3):499–562, July 2006.

[7] Helena Lindgren. Limitations in physicians knowledge when assessing dementia diseases - an evaluation study of a decision-support system. *Studies In Health Technology And Informatics*, 169:120–124, 2011.

[8] Pascal Nicolas, Laurent Garcia, Igor Stéphan, and Claire Lefèvre. Possibilistic Uncertainty Handling for Answer Set Programming. *Annals of Mathematics and Artificial Intelligence*, 47(1-2):139–181, June 2006.

[9] Juan Carlos Nieves, Mauricio Osorio, and Ulises Cortés. Semantics for possibilistic disjunctive programs. In Stefania Costantini and Richard Watson, editors, *Answer Set Programming: Advances in Theory and Implementation*, pages 271–284, 2007.

[10] Juan Carlos Nieves, Mauricio Osorio, and Ulises Cortés. Semantics for Possibilsitic Disjuntive Programs. *Theory and Practice of Logic Programming*, doi: 10.1017/S1471068411000408, 2011.

[11] Juan Carlos Nieves, Mauricio Osorio, and Claudia Zepeda. A Schema for Generating Relevant Logic Programming Semantics and its Applications in Argumentation Theory. *Fundamenta Informaticae*, 106(2-4):295–319, 2011.

[12] Mauricio Osorio and Juan Carlos 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.

[13] Henry Prakken and Gerard A. W. Vreeswijk. Logics for defeasible argumentation. In D. Gabbay and F. Günthner, editors, *Handbook of Philosophical Logic*, volume 4, pages 219–318. Kluwer Academic Publishers, Dordrecht/Boston/London, second edition, 2002.