Modality Argumentation Programming

Juan Carlos Nieves and Ulises Cortés

Universitat Politècnica de Catalunya,
Knowledge Engineering and Machine Learning Group,
Departament de Lleguatges i Sistemes Informàtics,
c/Jordi Girona 1-3, E08034, Barcelona, Spain
{jcnieves, ia}@lsi.upc.edu

Abstract. This work is focus in the following critical questions:
1. *How to incorporate modalities in the process of argumentation reasoning?*
2. *Is it possible to build arguments faced with incomplete information?*

Our proposal is based in a specification language which has the following properties: a) it permits to give specifications of modalities in a natural way; b) it defines a process of argumentation reasoning considering modalities; and c) it permits to build arguments from incomplete information.

Keywords: Decision-Making, Argumentation, Logic Programming, Rational Agents.

1 Introduction

Argumentation has proved to be a useful tool for representing and dealing with domains in which rational agents are not able to decide by themselves about something, and may encounter other agents with different preference values. The ability to reason effectively about what is the *best* or *most* appropriate course of action to take in a given situation is an essential activity for a rational agent. A simple rational agent may also use argumentation techniques to perform its individual reasoning as it needs to make rational decisions under complex preferences policies, or to reason about its commitments, its goals, *etc.*

A critical question about *how* to carry out argumentation theory to implementation systems still exists. For instance, one of the main objectives of the EU funded project ASPIC[1] is to provide a strong foundation for the design and implementation of a set of generic argument software components which can be used by 3^{rd} party applications.

1.1 Motivation

Since Aristotle, modalities have been an object of study for logicians especially in relation with the construction of arguments. Modalities are terms which indicate the level of *certainty* with which a claim can be made. One possible definition of *modality* is [3]:

[1] Consortium for argumentation technology. http://www.argumentation.org/

V. Torra et al. (Eds.): MDAI 2006, LNAI 3885, pp. 295–306, 2006.

> "The classification of logical propositions according to their asserting or denying the possibility, impossibility, contingency, or necessity of their content".

Research on rational agents has raised further questions about modalities in the context of argumentation, and the roles that arguments play in the pursuit of an agent's goals and plans.

In our own work on medical decision-making we have very different sources of examples of argumentation [1, 13]. The main objective is to discover the acceptable set of arguments that support a given claim in a given context. This is a purposeful and purposed process where the validity of arguments and the evidence of premises are both approached. One import point of our particular medical domain (organ transplant) is that there is small amount of information available *w.r.t.* the viability criterions which are applied whether a particular organ is viable to be transplanted. However, we have a hight-level of detail and quality information *w.r.t.* each medical case. Usually, the medical information (in our particular medical domain) is supported by a set of clinal tests.

Lattices have been used to model a wide range of problems. For instance, lattice domains are useful to perform aggregate operations which are a great tool for modeling decision-making in Artificial Intelligence. The use of lattice domains in declarative programming have shown high level of expressiveness. For example, the use of partial-order clauses and lattice domains in partial-order programming is particularly useful for expressing concise solutions to problems for graph theory, program analysis, and database querying [8, 10].

Osorio *et. al* [7] showed *how* to perform aggregate operations using negation as failure, also Nieves *et al* [6] showed how to perform relaxation in optimization problems using aggregate operations and negations as failure.

In this paper, we introduce a declarative language to handle arguments with modalities like *possible*, *probable*, *plausible*, *supported* and *open*. Modality is a category of linguistic meaning having to do with the expression of possibility and necessity. In [16] a study of the kinds of modal meaning can be found.

In §2 we put forward the syntax to be used and give a brief introduction to lattices and order. In §3 we introduce our framework and present some examples. In §4, we present the declarative semantics of our framework. Finally, in §5 we offer our conclusions.

2 Background

2.1 Syntax

The language of a propositional logic has an alphabet consisting of

(i) proposition symbols: $p_0, p_1, ...$
(ii) connectives : $\vee, \wedge, \leftarrow, \neg, \bot, \top$
(iii) auxiliary symbols : (,).

Where $\vee, \wedge, \leftarrow$ are 2-place connectives, $\neg$ is 1-place connective and $\bot, \top$ are 0-place connectives. The proposition symbols and $\bot$ stand for the indecomposable propositions, which we call *atoms*, or *atomic propositions*. A literal is an atom, a, or the negation of an atom $\neg a$. The *complement* of a literal is defined as $(a)^c = \neg a$ and $(\neg a)^c = a$.

A general clause, C, is denoted:

$$l_1 \vee \ldots \vee l_m \leftarrow l_1, \ldots, l_j, not\ l_{j+1}, \ldots, not\ l_n{}^2$$

where $m \geq 0$, $n \geq 0$, each l_i is a literal. When $n = 0$ and $m > 0$ the clause is an abbreviation of $l_1 \vee \ldots \vee l_m \leftarrow \top$[3], where $\top$ is $\neg\bot$. When $m = 0$ the clause is an abbreviation of $\bot \leftarrow l_1, \ldots, l_n{}^4$. Clauses of theses forms are called constraints (the rest, non-constraint clauses). A logic program is finite set of general clauses.

A signature $\mathcal{L}$ is a finite set of elements that we call atoms. By $\mathcal{L}_P$ we understand it to mean the signature of P, i.e. the set of atoms that occurs in P. Given a signature $\mathcal{L}$, we write $Prog_{\mathcal{L}}$ to denote the set of all programs defined over $\mathcal{L}$. A general semantics SEM is a function on $Prog_{\mathcal{L}}$ which associates with every program a partial interpretation.

We point out that we understand the negation $\neg$ as the classical negation and the negation *not* as the negation as failure [5].

2.2 Lattices and Order

In this section, we present some fundamental definitions of lattice theory in order to make this paper self contained (see [2] for more details).

Definition 1. *[2] Let P be a set. An order (or partial order) on P is a binary relation $\leq$ on P such that, for all $x, y, z \in P$,*

(i) $x \leq x$
(ii) $x \leq y$ *and* $y \leq x$ *imply* $x = y$
(iii) $x \leq y$ *and* $y \leq$ *imply* $x \leq z$

These conditions are referred to, respectively, as reflexivity, antisymmetry and transitivity.

A set P equipped with an order relation $\leq$ is said to be an order set (or partial ordered set).

Definition 2. *[2] Let P be an ordered set and let $S \subseteq P$. An element $x \in P$ is an upper bound of S if $s \leq x$ for all $s \in S$. A lower bound is defined dually. The set of all upper bounds of S is denoted by S^u (read as 'S upper') and the set of all lower bounds by S^l (read as 'S lower').*

If S^u has a least element x, then x is called the least upper bound (LUB) of S. Equivalently, x is the least upper bound of S if

[2] $l_1, \ldots, l_n$represents the formula $l_1 \wedge \ldots \wedge l_n$.
[3] Or simply $l_1 \vee \ldots \vee l_m$.
[4] In fact $\bot$ is used to define $\neg A$ as $A \rightarrow \bot$.

(i) x is an upper bound of S, and
(ii) $x \leq y$ for all upper bound y of S.

The least upper bound of S exists iff there exists $x \in P$ such that

$$(\forall y \in P)[((\forall s \in S)s \leq y) \Longleftrightarrow x \leq y],$$

and this characterizes the LUB of S. Dually, if S^l has a greatest element, x, then x is called the greatest lower bound (GLB) of S. Since least element and greatest elements are unique, LUB and GLB are unique when they exist.

The least upper bound of S is called the supremum of S and is denoted by *sup* S; the greatest lower bound of S is also called the infimum of S is denoted by $inf\ S$.

Definition 3. *[2] Let* P *be a non-empty order set.*

(i) *If* $sup\{x, y\}$ *and* $inf\{x, y\}$ *exist for all* $x, y \in P$*, then* P *is called lattice.*
(ii) *If sup* S *and* $inf\ S$ *exist for all* $S \subseteq P$*, then* P *is called a complete lattice.*

Example 1. Let us consider the set of labels $S :=$ { *Certain, Confirmed, Probable, Plausible, Supported, Open* } and let $\preceq$ be a partial order such that the following set of relations holds :

$\{Certain \preceq Confirmed, Confirmed \preceq Probable,$
$Confirmed \preceq Plausible, Probable \preceq Support,$
$Plausible \preceq Supported, Supported \preceq Open\}$

A graphic representation of S according to $\preceq$ is showed in Figure 1. It is not difficult to see that S is a lattice and even more it is complete lattice.

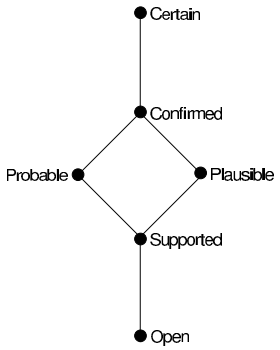

Fig. 1. A lattice

The labels given in Example 1 could be qualifiers of a knowledge base.

3 Modality Argumentation Programming

In this section, we are going to present our framework. We start by defining the syntax.

Definition 4 (Modality clause). *Let Q be a complete lattice. A modality clause is a clause of the form:*

$$Modality : C.$$

Where $Modality \in Q$ and C is a general clause.

Notice that by using a complete lattice Q, a modality clause categorizes the sentence expressed in the general clause C. This means that a modality clause locates a sentence in Q.

We understand a modality as a category of certain meaning having to do with the expression of possibility. Therefore, a set of possibilities could be categorized by a complete lattice. For instance, let $S :=$ { *Certain, Confirmed, Probable, Plausible, Supported, Open* } be a set of labels where each label is a possible category of believes, so this set could be categorized as it is shown in Figure 1.

Definition 5 (Modality logic program). *A modality logic program P is a tuple of the form $\langle Q, Modality\_Clauses \rangle$, where Q is a complete lattice and $Modality\_Clauses$ is a set of modality clauses such that for all $Modality : C \in Modality\_Clauses$, $Modality \in Q$.*

Example 2. Let Q be the lattice presented in Example 1 and let us consider the following propositions atoms which represent medical knowledge for organ transplantation.

- *dsve* = 'donor has streptococcus viridans endocarditis'.
- *risv* = 'recipient infected with streptococcus viridans'.
- *nv* = 'non viable'.

One possible modality logic program with its intuitive meaning could be described as follows:

Confirmed: *dsve*. (It is confirmed that the donor has been infected by streptococus viridas)
Plausible: *risv* ←*dsve*. (If the donor has been infected by streptococus viridas, then it is plausible that the recipient could be infected too.)
Probable : *nv*←*risv*. (If it is plausible that the recipient could be infected by streptococus viridas, then it is probable that his organs are not viable for transplanting)

Following the definition of argument presented in [11], we are going to define our concept of argument.

Definition 6 (Argument). *Let* $P := \langle Q, Modality\_Clauses\rangle$ *be a modality logic program, an argument Arg w.r.t. P is a tuple of the form*

$$Arg = \langle Claim, Support, Modality\_Qualifier\rangle$$

where Claim is a literal, Support is a finite set of modalities clauses such that:

1. *Support is consistent;*
2. $Support \models_I Claim$*;*
3. *Support is minimal, so no subset of Support satisfying both 1. and 2. exists.*

and $Modality\_Qualifier \in Q$*.*

Remark 1. The symbol $\models_I$ denotes logic consequence in Intuitionistic Logic (see [14] for details). Intuitionistic Logic has studied in the context of logic programming, specially in Answer Set Programming, with two kinds of negations [12, 9]. Notice that *Support* is minimal *w.r.t.* set inclusion and is not unique.

By definition, an argument has a *modality qualifier*. The modality qualifier has the objective of quantify the level of *certainty* of an argument. There are two kinds of quantifiers: *Pessimistic*, and *Optimistic*. So, we can define two kinds of arguments.

Definition 7 (Pessimistic argument). *Let Arg be an Argument of the form* $\langle Claim, Support, Modality\_Qualifier\rangle$*. Arg is a pessimistic argument if*

$$Modality\_Qualifier := GLB\{Modality\_Qualifier|$$
$$(Modality\_Qualifier : formula) \in Support\}$$

Definition 8 (Optimistic argument). *Let Arg be an argument of the form* $\langle Claim, Support, Modality\_Qualifier\rangle$*. Arg is an optimistic argument if*

$$Modality\_Qualifier := LUB\{Modality\_Qualifier|$$
$$(Modality\_Qualifier : formula) \in Support\}$$

Example 3. Let us consider again the lattice of Example 1 and the modality logic program presented in Example 2. One possible argument is:

$\langle nv, \{(Confirmed : dsve), (Plausible : risv \leftarrow dsve.), (Probable : nv \leftarrow risv.)\},$ **?**$\rangle$

So, a pessimistic argument is:

$\langle nv, \{(Confirmed : dsve), (Plausible : risv \leftarrow dsve.), (Probable : nv \leftarrow risv.)\},$ **Supported**$\rangle$

an optimistic argument:

$\langle nv, \{(Confirmed : dsve), (Plausible : risv \leftarrow dsve.), (Probable : nv \leftarrow risv.)\},$ **Confirmed**$\rangle$

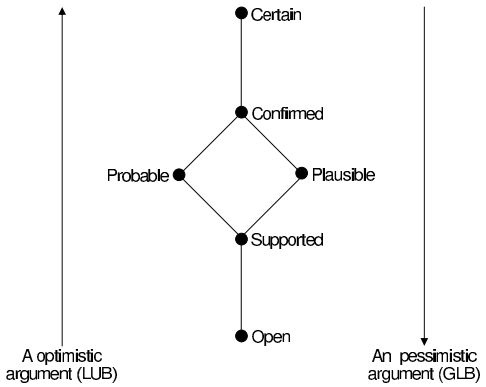

Fig. 2. A lattice of modalities

In the Figure 2 is presented the lattice with the levels of *certainty* that an argument could be defined.

Two arguments constructed from two different knowledge bases (two different rational agents) could be in conflict. We are going to define two well known kinds of conflicts: *attack* and *undercutting*.

Definition 9. *Let* $Arg_1 = \langle Claim_1, Support_1, Modality\_Qualifier_1 \rangle$ *and* $Arg_2 = \langle\ Claim_2, Support_2, Modality\_Qualifier_2\ \rangle$. Arg_1 *attacks* Arg_2, *if* $Claim_1 = l$ *and* $Claim_2 = (l)^c$.

Definition 10. *Let* $Arg_1 = \langle Claim_1, Support_1, Modality\_Qualifier_1 \rangle$ *and* $Arg_2 = \langle\ Claim_2, Support_2, Modality\_Qualifier_2\ \rangle$. Arg_1 *undercuts* Arg_2, *if* $Claim_1 = l_i$ *and there is a modality clause*

$$(Modality : l_1 \vee \ldots \vee l_m \leftarrow l_1, \ldots, l_j, not\ l_{j+1}, \ldots, not\ l_i, \ldots, l_n) \in Support_2$$

By considering the concepts of *attack* and *undercut*, we define our concept of *defeat*.

Definition 11. *Let* $Arg_1 = \langle\ Claim_1,\ Support_1,\ Modality\_Qualifier_1\ \rangle$ *and* $Arg_2 = \langle\ Claim_2, Support_2, Modality\_Qualifier_2\ \rangle$. Arg_1 *defeats* Arg_2, *if* Arg_1 *attacks/undercuts* Arg_2 *and* $LUB\{\ Modality\_Qualifier_1, Modality\_Qualifier_2\} = Modality\_Qualifier_1$.

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 4. Let us consider the lattice presented Example 1, and the following proposition atoms:

- a = 'donor is HIV positive'.
- b = 'organ is viable for transplanting'.
- c = 'organ has correct functions and correct structure '.
- q = 'positive clinical test'.

Let P_1 be the following modality logic program:

Probable: a. (It is probable that donor is HIV positive)
Probable: $\neg b \leftarrow a, not\ c$. (If donor is HIV positive and there is not evidence that the organ has correct functions and correct structure, then the organ is not viable for transplanting)

One possible argument Arg_1 from P_1 is :

$$\langle \neg b, \{(Probable : a.), (Probable : \neg b \leftarrow a, not\ c.)\}, Probable \rangle$$

This argument suggests that the organ is not viable for transplanting ($\neg b$). Now, let P_2 be the following modality logic program:

Confirmed: q. (It is confirmed that the organ has positive clinical tests)
Plausible: $c \leftarrow q$. (If the organ has positive clinical tests, then it is plausible that the organ has correct functions and correct structure.)

One possible argument Arg_2 from P_2 is :

$$\langle c, \{(confirmed : q.), (Plausible : c \leftarrow q)\}, Confirmed \rangle$$

One can see that Arg_2 undercuts Arg_1 and even more Arg_2 defeats Arg_1 because $LUB\{Probable, Confirmed\} = Confirmed$. So, one can not say explicitly that an organ is not viable for transplanting ($\neg b$).

This example is controversial in the medical domain, because usually an organ from a donor who is HIV positive is not viable for transplanting. However, there are cases where the recipient is also HIV positive then he could be receptor of an acceptable organ from a donor HIV positive.

4 Declarative Semantics

In this section, we are going to present the declarative semantics for our framework. This semantics is characterized in two parts. The first part determines the models of the modality logic program without considering the modality qualifier and the second one determines the modality qualifiers of the arguments using aggregate operations which are implemented by negation as failure.

Definition 12. *Let P be a modality logic program, $\Delta(P)$ is a logic program defined as follows:*

$$\Delta(P) := \{C | (Modality : C) \in P\}.$$

Definition 13. *Let S be a set of modality clauses, $\Gamma(S)$ is a set of clauses defined as follows:*

$$\Gamma(S) := \{Qualifier(Modality) | (Modality : C) \in S\}$$

The following two definitions are similar to Definition 4.11 of [7].

Definition 14 (GLB_basic_ext). *Given a complete lattice Q with the partial order $\preceq$ and a modality logic program P where the modality clauses of P are defines under Q. We definite GLB_basic_ext as the set of the following set of clauses:*

1. $f_{\preceq}(X) \leftarrow Qualifier(X).$
2. $f_{\preceq}(X) \leftarrow f_{\preceq}(X1), X1 \prec X.$
3. $f_{\prec}(X) \leftarrow f_{\preceq}(X1), X1 \prec X.$
4. $f_{=}(X) \leftarrow f_{\preceq}(X), \neg f_{\prec}(X).$
5. $f_{\preceq}(Z) \leftarrow f_{\preceq}(X), f_{\preceq}(Y), GLB_{\preceq}(X, Y, Z).$

Definition 15 (LUB_basic_ext). *Given a complete lattice Q with the partial order $\preceq$ and a modality logic program P where the modality clauses of P are defines under Q. We define LUB_basic_ext as the set of the following set of clauses:*

1. $f_{\preceq}(X) \leftarrow Qualifier(X).$
2. $f_{\preceq}(X) \leftarrow f_{\preceq}(X1), X1 \succ X.$
3. $f_{\prec}(X) \leftarrow f_{\preceq}(X1), X1 \succ X.$
4. $f_{=}(X) \leftarrow f_{\preceq}(X), \neg f_{\prec}(X).$
5. $f_{\preceq}(Z) \leftarrow f_{\preceq}(X), f_{\preceq}(Y), LUB_{\preceq}X, Y, Z).$

Now, we present how to build arguments from a modality logic program considering its models.

Definition 16. *Let P be a modality logic program and $M(P) := SEM(\Delta(P))$. $Arg := \langle Claim, Support, Modality\_Qualifier \rangle$ is an argument from P iff $Claim \in M(P)$ and $Modality\_Qualifier := modality$ where if Arg is a pessimistic argument*

$$f_{=}(modality) \in WFS(\Gamma(Support) \cup GLB\_basic\_ext)$$

or if Arg is optimistic argument

$$f_{=}(modality) \in WFS(\Gamma(Support) \cup LUB\_basic\_ext)$$

WFS(P) is a function which infers the well-founded model of the program P[5]*.*

In order to illustrate the above definitions, we present the following example.

Example 5. Let us consider again the lattice of Example 1 and the following proposition atoms which represent, like in Example 2, medical knowledge.

[5] See [4] for a formal definition of the well-founded semantics.

- dbd = 'donor is brain-dead'
- dma = 'discard metastatic abscess'
- $dbce$ = 'determine bacteria causing endocarditis'
- bsv = 'bacteria is streptoccocus viridians'

Let Π be the following modality logic program:

Confirmed: dbd. (The donor is brain-dead)
Probable: $dma \vee dbce \leftarrow dbd$. (It is probable that if a donor is brain-dead, then it is discarded a metastatic abscess or there is a bacteria causing endocarditis)
Confirmed: $dbce \leftarrow bsv$. (It is confirmed that if a donor has been infected by streptoccocus viridians, then it is diagnosed endocarditis)

Then

$$\Delta(\Pi) := \{(dbd.), (dma \vee dbce \leftarrow dbd.), (dbce \leftarrow bsv)\}$$

In this example, we consider SEM(P) as the stable models semantics [5]. Let us consider the stable models of $\Delta(\Pi)$ which are $\{dbd, dma\}$,$\{dbd, dbce\}$. This means that we can construct three different arguments:

1. $\langle dbd, \{(Confirmed : dbd.)\}, Qualifier_1\rangle$
2. $\langle dbce, \{(Confirmed : dbd.), (Probable : dma \vee dbce \leftarrow dbd)\}, Qualifier_2\rangle$
3. $\langle dma, \{Confirmed : dbd.\}, (Probable : dma \vee dbce \leftarrow dbd)\}, Qualifier_3\rangle$

These arguments have not defined their modality quantifiers yet. Let us consider the support of Argument 2. $S := \{(Confirmed : dbd.), (Probable : dma \vee dbce \leftarrow dbd)\}$, so

$$\Gamma(S) := \{(Qualifier(Confirmed).), (Qualifier(Probable).)\}$$

By considering $WFS(\Gamma(S) \cup GLB\_basic\_ext)$, we can infer the pessimist modality qualifier of Argument 2, it is not difficult to see that $f_{=}(Probable) \in WFS(\Gamma(S) \cup GLB\_basic\_ext)$, this means that $Qualifier_2 := Probable$.

$$\langle dbce, \{(Confirmed : dbd.), (Probable : dma \vee dbce \leftarrow dbd)\}, Probable\rangle$$

This means, that in this context, we have an argument that suggests that if an donor is brain-dead, then it is *probable* that he could be infected by a bacteria which is causing endocarditis.

Notice that, the use of disjunctive clauses allows to build arguments under incomplete information and also the quantification of the knowledged base permits to quantify the arguments. By using this kind of arguments, it is possible to support decisions taken under incomplete information.

5 Conclusions and Future Work

In this work we introduced an argumentation framework which allows to incorporate modalities during the process of argumentation reasoning. We understand a modality as a category of certain meaning having to do with the expression of possibility. Therefore, a set of possibilities could be categorized by a complete lattice.

Our argumentation framework is based in a specification language which permits to provide specifications with levels of *certainty* in a natural way. Also, the specification language allows to use disjunctive clauses, so it allows specifications in situations where the available information is incomplete, as in the medical domain showed in the examples. The declarative semantics of our language permits to build arguments such that any argument is supported by a set of modality clauses and the argument's claim is quantified *w.r.t.* its support. We present a couple of examples from our real application to manage the assignation process of human organs for transplantation [1, 13], although the examples are simple they permit to see the potential of our framework.

Among the future work, we have planned to deploy this framework in the context of multi-agent systems, in particular to our real multi-agent system called CARREL [15].

Acknowledgements

Thanks to Mauricio Osorio for his fruitful commends and discussions. We thank Antonio López-Navidad for his assessment to formulate the clinical information related with examples. Also we would like to thank the anonymous referees for their comments.

This research was partially supported by the grant FP6-IST-002307 (ASPIC). The views expressed in this paper are not necessarily those of ASPIC consortium.

References

1. U. Cortés, P. Tolchinsky, J. C. Nieves, A. López-Navidad, and F. Caballero. Arguing the discard of organs for tranplantation in CARREL. In *CATAI 2005*, pages 93–105, 2005.
2. B. A. Davey and H. A. Priestly. *Introduction to Lattices and Order*. Cambridge University Press, second edition, 2002.
3. M.-W. O. Diccionary. http://www.m-w.com/.
4. A. V. Gelder, K. A. Ross, and J. S. Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM*, 38(3):620–650, 1991.
5. M. Gelfond and V. Lifschitz. Classical Negation in Logic Programs and Disjunctive Databases. *New Generation Computing*, 9:365–385, 1991.
6. J. C. Nieves and B. Jayaraman. Relaxation in partial order programming. In *Workshop on Logic and Computation: MICAI'2002*, 2002.
7. M. Osorio and B. Jayaraman. Relating aggregation and negation-as-failure. *New Generation Comput.*, 17(3):255–284, 1999.

8. M. Osorio, B. Jayaraman, and D. A. Plaisted. Theory of partial-order programming. *Sci. Comput. Program.*, 34(3):207–238, 1999.
9. M. Osorio, J. A. Navarro, and J. Arrazola. Applications of Intuitionistic Logic in Answer Set Programming. *Theory and Practice of Logic Programming (TPLP)*, 4:325–354, May 2004.
10. M. Osorio, J. C. Nieves, and B. Jayaraman. Aggregation in functional query languages. *Journal of Functional and Logic Programming*, 2004(2), August 2004.
11. S. Parsons and P. McBurney. Argumentation-based dialogues for agent coordination. *Group Decision and Negotiation*, 12(5):415–439, 2003.
12. D. Pearce. Stable Inference as Intuitionistic Validity. *Logic Programming*, 38:79–91, 1999.
13. 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.
14. D. van Dalen. *Logic and structure*. Springer-Verlag, Berlin, 3rd., aumented edition edition, 1994.
15. J. Vázquez-Salceda, U. Cortés, J. Padget, A. López-Navidad, and F. Caballero. Extending the CARREL System to mediate in the organ and tissue allocation processes: A first approach. *Artificial Intelligence in Medicine*, 3:233–258, 2003.
16. K. von Fintel. Modality and language. In D. M. Borchert, editor, *Encyclopedia of Philosophy – Second Edition*. MacMillan, 2005.