

Weighted Multi Dimensional Logic Programs

Pierangelo Dell'Acqua

Department of Science and Technology - ITN
Linköping University, Norrköping, Sweden
`pier@itn.liu.se`

and

Centro de Inteligência Artificial - CENTRIA
Departamento de Informática, Faculdade de Ciências e Tecnologia
Universidade Nova de Lisboa, 2829-516 Caparica, Portugal

Abstract. We introduce a logical framework suitable to formalize structures of epistemic agents. Such a framework is based on the notion of weighted directed acyclic graphs (WDAGs) that allow one to assign a measure of strength to the knowledge relationships represented by the edges. We present the declarative and operational semantics of such a framework, and give results of correctness. We illustrate the usage of the framework by a number of examples.

1 Motivation

In previous papers [4, 6, 8] we presented a logical formalization of a framework for multi-agent systems where we embedded a flexible and powerful kind of epistemic agent. In fact, these agents are rational, reactive, abductive, able to prefer and they can update the knowledge base of other agents (including their own). There we presented a declarative semantics for this kind of agent. In [1] we provided a syntactical transformation that is at the basis for a proof procedure for updating and preferring in agents, and in [5] we presented a framework for handling the communication among epistemic agents asynchronously.

It is advocated, especially in open multi-agent systems (cf. [3, 14]), that there is a need to make the organizational elements as well as the formalization of the agent interactions of a multi-agent system externally visible rather than being embedded in the mental state of each agent, i.e., it is desired to explicitly represent the organizational structure and the agent interactions. Zambonelli [13] states that modelling and engineering interactions in complex and open multi-agent systems cannot simply rely on the agent capabilities for communicating. Rather, there is a need for concepts like organizational rules, social laws and active environments. For the effective engineering of multi-agent systems, high-level, inter-agent concepts, language extensions, and abstractions must be defined to explicitly model the organization, the society in which agents operate, and the associated organizational laws. With this purpose in mind, in [7] we elaborated over the approach proposed in [6], and usher in the more flexible notion of weighted directed acyclic graphs (WDAGs), generalizing the notion of

DAGs presented in [11], and permitting us to address the issues at hand by assigning a measure of strength to the knowledge sharing relationships represented by the edges. Arguably, the use of WDAGs as a representational tool allows one to formalize in a natural and abstract way several agent societies. It also promotes information hiding. Consider for example a society comprising two agents, α and β . To express that α is predominant over β , we can associate a weight to α greater than the weight associated to β without having to know the internal representational structure of the agents. We showed how to explicitly represent a number of organizational structures for epistemic agents in multi-agent systems.

The contribution of the paper is to provide the operational semantics for WMDLP, the framework proposed in [7], and to give results of correctness. The usage of WMDLP is illustrated by exploring agent societies based on confidence factors, and voting.

An implementation of WMDLP is available at <http://www.itn.liu.se/~piede>.

2 Background

Generalized Logic Programs

To represent negative information we allow default negation *not* A to occur in premises of rules as well as in their heads¹. By a *generalized logic program* P over a language \mathcal{L} we mean a finite or infinite set of rules of the form $L_0 \leftarrow L_1, \dots, L_n$, where each L_i is a *literal* (i.e. an atom A or its default negation *not* A). We use ';' to separate the rules in a program. For instance, if a program P contains the rule $a \leftarrow b, c$ together with the rule $e \leftarrow f$ and the fact d , we write $P = \{a \leftarrow b, c; e \leftarrow f; d\}$. In the following we syntactically represent generalized logic programs as propositional Horn theories. In particular, we represent default negation *not* A as a propositional variable. Given an arbitrary set \mathcal{K} of propositional variables, whose names do not begin with *not*, the propositional language \mathcal{L} generated by \mathcal{K} is the language whose set of propositional variables consists of:

$$\{A \mid A \in \mathcal{K}\} \cup \{\text{not } A \mid A \in \mathcal{K}\}.$$

If r is a rule of the form $L_0 \leftarrow L_1, \dots, L_n$, by *head*(r) we mean L_0 , and by *body*(r) we mean L_1, \dots, L_n . If *head*(r) = A (resp. *head*(r) = *not* A) then *not head*(r) = *not* A (resp. *not head*(r) = A). By a (2-valued) *interpretation* M of \mathcal{L} we mean any set of literals from \mathcal{L} that satisfies the condition that for any atom A , precisely one of the literals A or *not* A belongs to M . Given an interpretation M , we define $M^+ = \{A \mid A \in M\}$ and $M^- = \{\text{not } A \mid \text{not } A \in M\}$.

Following established tradition, wherever convenient we omit the default atoms when describing interpretations and models. We say that a (2-valued) interpretation M of \mathcal{L} is a *stable model* of a generalized logic program P if $M = \text{least}(P \cup M^-)$. The class of generalized logic programs can be viewed as a special case of yet broader classes of programs, introduced earlier in [12], and,

¹ For further motivation and intuitive reading of logic programs with default negations in the heads see [2].

for the special case of normal programs, their semantics coincides with the stable models semantics [9].

Directed Acyclic Graphs

A *directed graph* $D = (V, E)$ is a pair comprised of a finite set V of *vertices* and a finite set E of ordered pairs (v_1, v_2) called *edges*, where v_1 and v_2 are vertices in V with $v_1 \neq v_2$. The vertex v_1 is the *initial vertex* of the edge and v_2 the *terminal vertex*. The *in-valency* of a vertex v is the number of edges with v as their terminal vertex. The *out-valency* of a vertex v is the number of edges with v as their initial vertex. A *source* is a vertex with in-valency 0 and a *sink* a vertex with out-valency 0. A *path* in a directed graph is a sequence of consecutive edges in the graph that begins at an initial vertex and ends at a terminal vertex. If there exists a path from a vertex v_1 to a vertex v_2 we write $v_1 \prec v_2$, otherwise $v_1 \not\prec v_2$. We write $v_1 \preceq v_2$ if $v_1 \prec v_2$ or $v_1 = v_2$. We write $v_1 \not\preceq v_2$ if $v_1 \not\prec v_2$ and $v_1 \neq v_2$. Sometime to make explicit the vertices occurring in a path $v_1 \prec v_n$ we write the sequence $\langle v_1 v_2 \dots v_n \rangle$ of all vertices occurring in the path. A *cycle* is a path in which the initial vertex of the path is also the terminal vertex. A *directed acyclic graph* (DAG) is a directed graph that does not contain any cycle.

3 Weighted Directed Acyclic Graphs

In this section, we first introduce weighted directed acyclic graphs (WDAGs) that generalize the notion of DAGs to associate weights to their edges, and then we presents a number of notions based on WDAGs.

Definition 1. A weighted directed graph is a tuple (V, E, w) where V is a set of vertices, E a set of edges and $w : E \rightarrow \mathbb{R}^+$ a function mapping the edges in E to positive real numbers in \mathbb{R}^+ . A weighted directed acyclic graph (WDAG) is a weighted directed graph that does not contain any cycle.

Given two vertices v_1 and v_n in a WDAG, the following definition distinguishes the paths from v_1 to v_n that are dominant.

Definition 2. Let $D = (V, E, w)$ be a WDAG and $v_1 \prec v_n$ ($1 < n$) a path with vertices $\langle v_1 v_2 \dots v_n \rangle$. Then, $v_1 \prec v_n$ is a dominant path iff for every path $\langle a_1 a_2 \dots a_m \rangle$ ($1 < m$) such that $a_1 = v_1$, $a_m = v_n$ and $v_i = a_j$ for some i, j with $1 < i \leq n$, $1 < j \leq m$ it holds that $w((v_{i-1}, v_i)) \geq w((a_{j-1}, a_j))$.

Example 1. Let $D = (V, E, w)$ be the WDAG depicted in Fig. 1a, where $V = \{v_1, v_2, v_3, v_4\}$, E consists of the edges: $e_1 = (v_2, v_1)$, $e_2 = (v_3, v_2)$, $e_3 = (v_3, v_1)$, $e_4 = (v_2, v_4)$ and $e_5 = (v_4, v_1)$. Let $w(e_1) = 0.4$, $w(e_2) = 0.8$, $w(e_4) = 0.3$ and $w(e_3) = w(e_5) = 0.6$. Then there exist two paths from v_3 to v_1 that are dominant: the path $\langle v_3 v_1 \rangle$ and the path $\langle v_3 v_2 v_4 v_1 \rangle$. Note that the path $\langle v_3 v_2 v_1 \rangle$ is not dominant.

Definition 3. Let $D = (V, E, w)$ be a WDAG and $v_1 \prec v_n$ a dominant path with vertices $\langle v_1 v_2 \dots v_n \rangle$. Then, the *prevailment relation* is defined as follows:

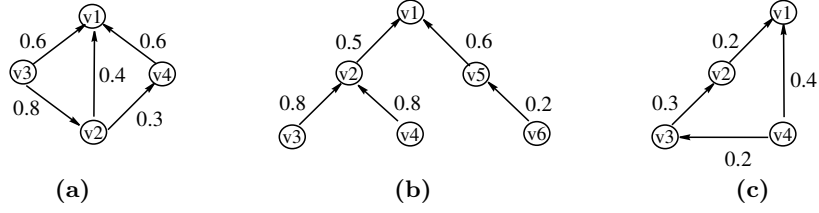


Fig. 1. WDAGs of Example 1, 2 and 3

- Every vertex v_i prevails v_1 wrt. v_n , for every $1 < i \leq n$.
- If there exists a path $a_1 \prec v_i$ with vertices $\langle a_1 \dots a_m v_i \rangle$ ($1 \leq m$), for some $1 < i \leq n$, and $w((v_{i-1}, v_i)) < w((a_m, v_i))$, then every vertex a_j prevails v_1 wrt. v_n , for every $1 \leq j \leq m$.

In the following, we write $v_1 \underset{v}{\subset} v_2$ to indicate that v_2 prevails v_1 wrt. v , and $v_1 \not\underset{v}{\subset} v_2$ to indicate that v_2 does not prevail v_1 wrt. v .

Example 2. Let $D = (V, E, w)$ be the WDAG depicted in Fig. 1b, where $V = \{v_1, v_2, v_3, v_4, v_5, v_6\}$, E consists of the following edges $e_1 = (v_2, v_1)$, $e_2 = (v_3, v_2)$, $e_3 = (v_4, v_2)$, $e_4 = (v_5, v_1)$ and $e_5 = (v_6, v_5)$. Let $w(e_1) = 0.5$, $w(e_2) = w(e_3) = 0.8$, $w(e_4) = 0.6$ and $w(e_5) = 0.2$. Then, the prevalence relation includes among the others the following: $v_4 \underset{v_1}{\subset} v_1$, $v_2 \underset{v_1}{\subset} v_1$, $v_6 \underset{v_1}{\subset} v_5$, $v_2 \underset{v_1}{\subset} v_5$ and $v_4 \underset{v_1}{\subset} v_6$.

Example 3. Let $D = (V, E, w)$ be the WDAG depicted in Fig. 1c, where $V = \{v_1, v_2, v_3, v_4\}$, E consists of the following edges $e_1 = (v_2, v_1)$, $e_2 = (v_3, v_2)$, $e_3 = (v_4, v_1)$ and $e_4 = (v_4, v_3)$. Let $w(e_1) = 0.2$, $w(e_2) = 0.3$, $w(e_3) = 0.4$ and $w(e_4) = 0.2$. Then, the prevalence relation contains: $v_3 \underset{v_1}{\subset} v_1$, $v_4 \underset{v_1}{\subset} v_1$, $v_2 \underset{v_1}{\subset} v_4$ and $v_3 \underset{v_1}{\subset} v_4$. The vertex v_3 does not prevail v_4 since the unique dominant path from v_4 to v_1 is $\langle v_4 v_1 \rangle$.

The next result presents the basic properties of the prevalence relation.

Lemma 1. Let $D = (V, E, w)$ be a WDAG and $\underset{v}{\subset}$ the prevalence relation wrt. a vertex $v \in V$ over D . Then, it holds that:

- a) $\underset{v}{\subset}$ is irreflexive, i.e., $\forall v_1 \in V. v_1 \not\underset{v}{\subset} v_1$,
- b) $\underset{v}{\subset}$ is not antisymmetric, i.e., it does not hold $\forall v_1, v_2 \in V. v_1 \underset{v}{\subset} v_2 \Rightarrow v_2 \not\underset{v}{\subset} v_1$,
- c) $\underset{v}{\subset}$ is not transitive.

The next examples illustrate two WDAGs whose prevalence relation is neither antisymmetric nor transitive.

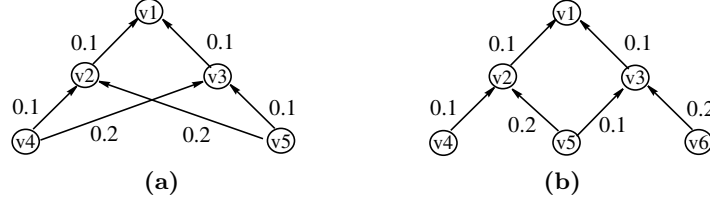


Fig. 2. WDAGs of Example 4 and 5

Example 4. Let $D = (V, E, w)$ be the WDAG depicted in Fig. 2a, where $V = \{v_1, v_2, v_3, v_4, v_5\}$, E consists of the edges $e_1 = (v_2, v_1)$, $e_2 = (v_3, v_1)$, $e_3 = (v_4, v_2)$, $e_4 = (v_5, v_2)$, $e_5 = (v_4, v_3)$ and $e_6 = (v_5, v_3)$. Let $w(e_1) = w(e_2) = w(e_3) = w(e_4) = 0.1$ and $w(e_5) = w(e_6) = 0.2$. Then, we have that $v_4 \subset_{v_1} v_5$ and $v_5 \subset_{v_1} v_4$. Thus, \subset_{v_1} is not antisymmetric.

Example 5. Let $D = (V, E, w)$ be the WDAG depicted in Fig. 2b, where $V = \{v_1, v_2, v_3, v_4, v_5, v_6\}$, E consists of the edges $e_1 = (v_2, v_1)$, $e_2 = (v_3, v_1)$, $e_3 = (v_4, v_2)$, $e_4 = (v_5, v_2)$, $e_5 = (v_5, v_3)$ and $e_6 = (v_6, v_3)$. Let $w(e_1) = w(e_2) = w(e_3) = w(e_5) = 0.1$ and $w(e_4) = w(e_6) = 0.2$. Then, we have that $v_4 \subset_{v_1} v_5$ and $v_5 \subset_{v_1} v_6$. Since $v_4 \not\subset_{v_1} v_6$, \subset_{v_1} is not transitive.

To avoid cases of vertices that can reciprocally prevail one another, we introduce the notion of strong prevalence.

Definition 4. Let $D = (V, E, w)$ be the WDAG. A vertex $v_2 \in V$ strongly prevails a vertex $v_1 \in V$ wrt. a vertex $v \in V$ iff $v_1 \subset_v v_2$ and $v_2 \not\subset_v v_1$.

We write $v_1 \triangleleft_v v_2$ to indicate that v_2 strongly prevails v_1 wrt. v . The properties of the strong prevalence relation are summarized by the next result.

Lemma 2. Let $D = (V, E, w)$ be a WDAG and \triangleleft_v the strong prevalence relation wrt. a vertex $v \in V$ over D . Then, it holds that:

- a) \triangleleft_v is irreflexive,
- b) \triangleleft_v is antisymmetric, and
- c) \triangleleft_v is not transitive.

4 Logic Framework

In this section, we generalize MDLPs (introduced in [11]) to allow for states to be represented by the vertices of WDAGs and their relations by the corresponding weighted edges. This enables us to prioritize the dimensions of a representational

updatable system. Such a system is suitable to formalize among others organizational structures for epistemic agents (cf. [7]), and societies of epistemic agents based on confidence factors. In this setting, *WMDLP* assigns semantics to sets of generalized logic programs, depending on how they stand in relation to one another.

The next definition extends the original definition of MDLP to take into consideration WDAGs.

Definition 5. *Let \mathcal{L} be a propositional language. A weighted multi-dimensional dynamic logic program (WMDLP) \mathcal{P} is a pair (\mathcal{P}_D, D) , where $D = (V, E, w)$ is a WDAG and $\mathcal{P}_D = \{P_v \mid v \in V\}$ is a set of generalized logic programs over \mathcal{L} indexed by the vertices $v \in V$.*

Following the established terminology of MDLP, we call *states* the vertices of WDAGs. We can now introduce the declarative semantics for WMDLPs.

Definition 6. *Let $\mathcal{P} = (\mathcal{P}_D, D)$ be a WMDLP, where $D = (V, E, w)$ and $\mathcal{P}_D = \{P_v \mid v \in V\}$. Let $s \in V$ be a state and \triangleleft_s the strong prevalence relation wrt. s over D . An interpretation M is a stable model of \mathcal{P} at state s iff*

$$M = \text{least}([\mathcal{Q}(\mathcal{P}, s) - \text{Reject}(\mathcal{P}, s, M)] \cup \text{Default}(\mathcal{Q}(\mathcal{P}, s), M))$$

where:

$$\begin{aligned} \mathcal{Q}(\mathcal{P}, s) &= \bigcup_{v \triangleleft_s} P_v \\ \text{Reject}(\mathcal{P}, s, M) &= \{r \in P_{v_2} \mid \exists r' \in P_{v_1}, \text{head}(r) = \text{not head}(r'), M \models \text{body}(r'), \\ &\quad \text{and } v_2 \triangleleft_s v_1\} \\ \text{Default}(\mathcal{Q}(\mathcal{P}, s), M) &= \{\text{not } A \mid \nexists r \in \mathcal{Q}(\mathcal{P}, s), \text{head}(r) = A \text{ and } M \models \text{body}(r)\}. \end{aligned}$$

$\mathcal{Q}(\mathcal{P}, s)$ contains all rules of all programs that are indexed by a state along all paths to a state s , i.e. all rules that are potentially relevant to determine the semantics at s . The set $\text{Reject}(\mathcal{P}, s, M)$ of rejected rules contains those rules belonging to a program indexed by a state v_2 that are overridden by the head of another rule with true body in state v_1 such that v_1 strongly prevails v_2 wrt. s . Note that we need to use strong prevalence (and not prevalence) otherwise we can have situations where rules reject each other. For instance, given the two programs $P_{v_2} = \{a\}$ and $P_{v_1} = \{\text{not } a\}$, if the two states v_1 and v_2 reciprocally prevail one another, then both rules a and $\text{not } a$ would be rejected. $\text{Default}(\mathcal{Q}(\mathcal{P}, s), M)$ contains default negations $\text{not } A$ of all unsupported atoms A , i.e., those atoms A for which there is no rule in $\mathcal{Q}(\mathcal{P}, s)$ whose body is true in M .

It is worth noting that MDLP [11] is a special case of WMDLP where all the edges have the same weight. In this case, the strong prevalence relation reduces to the path relation over which MDLP is based. Hence, at the semantical level the sets of rejected rules in MDLP and WMDLP are the same.

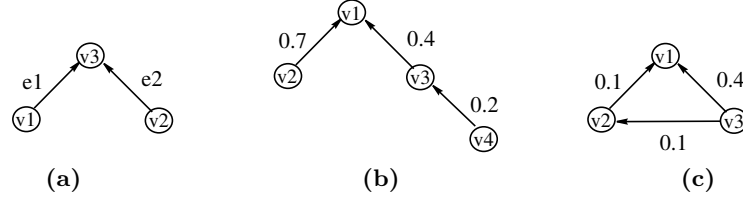


Fig. 3. WDAGs of Example 6, 7 and 8

Example 6. Let $\mathcal{P} = (\mathcal{P}_D, D)$ be the WMDLP depicted in Fig. 3a, where $D = (V, E, w)$ and $V = \{v_1, v_2, v_3\}$. E consists of $e_1 = (v_1, v_3)$ and $e_2 = (v_2, v_3)$. Let $\mathcal{P}_D = \{P_{v_1}, P_{v_2}, P_{v_3}\}$ with $P_{v_1} = \{a\}$, $P_{v_2} = \{\text{not } a\}$ and $P_{v_3} = \{\}$. If $w(e_1) = w(e_2)$, then there exists no stable model of \mathcal{P} at state v_3 . In fact, $Q(\mathcal{P}, v_3) = \{a; \text{not } a\}$ and $\text{Reject}(\mathcal{P}, v_3, M) = \{\}$, $\text{Default}(Q(\mathcal{P}, v_3), M) = \{\}$, for any interpretation M . Thus, there exists no interpretation M such that $M = \text{least}(Q(\mathcal{P}, v_3))$. If instead $w(e_1) > w(e_2)$, then there exists a unique stable model $M = \{a\}$ of \mathcal{P} at state v_3 . In fact, $\text{Reject}(\mathcal{P}, v_3, M) = \{\text{not } a\}$ as $v_2 \triangleleft_{v_3} v_1$, $\text{Default}(Q(\mathcal{P}, v_3), M) = \{\}$ and $M = \text{least}(\{a\})$.

Example 7. Let $\mathcal{P} = (\mathcal{P}_D, D)$ be the WMDLP depicted in Fig. 3b, where $D = (V, E, w)$ and $V = \{v_1, v_2, v_3, v_4\}$. E consists of $e_1 = (v_2, v_1)$, $e_2 = (v_3, v_1)$ and $e_3 = (v_4, v_3)$, and $w(e_1) = 0.7$, $w(e_2) = 0.4$ and $w(e_3) = 0.2$. Let $\mathcal{P}_D = \{P_{v_1}, P_{v_2}, P_{v_3}, P_{v_4}\}$ with $P_{v_1} = \{b\}$, $P_{v_2} = \{a\}$, $P_{v_3} = \{\text{not } a; d \leftarrow a, b\}$ and $P_{v_4} = \{\text{not } b\}$. The unique stable model of \mathcal{P} at state v_1 is $M = \{a, b, d\}$. In fact, as it holds that $v_4 \triangleleft_{v_1} v_1$ and $v_3 \triangleleft_{v_1} v_2$, we have that $\text{Reject}(\mathcal{P}, v_1, M) = \{\text{not } a; \text{not } b\}$ and $\text{Default}(Q(\mathcal{P}, v_1), M) = \{\}$. Thus, $M = \text{least}(\{a; b; d \leftarrow a, b\})$.

Example 8. Let $\mathcal{P} = (\mathcal{P}_D, D)$ be the WMDLP depicted in Fig. 3c, where $D = (V, E, w)$ and $V = \{v_1, v_2, v_3\}$. E consists of $e_1 = (v_2, v_1)$, $e_2 = (v_3, v_1)$ and $e_3 = (v_3, v_2)$, and $w(e_1) = 0.1$, $w(e_2) = 0.4$ and $w(e_3) = 0.1$. Let $\mathcal{P}_D = \{P_{v_1}, P_{v_2}, P_{v_3}\}$ with $P_{v_1} = \{c \leftarrow a, b, \text{not } d\}$, $P_{v_2} = \{b; \text{not } a\}$ and $P_{v_3} = \{a; \text{not } c\}$. The unique stable model of \mathcal{P} at state v_1 is $M = \{a, b, c\}$. In fact, as it holds that $v_2 \triangleleft_{v_1} v_1$, $v_3 \triangleleft_{v_1} v_1$ and $v_2 \triangleleft_{v_1} v_3$, we have that $\text{Reject}(\mathcal{P}, v_1, M) = \{\text{not } a; \text{not } c\}$ and $\text{Default}(Q(\mathcal{P}, v_1), M) = \{\text{not } d\}$. Thus, $M = \text{least}(\{a; b; \text{not } d; c \leftarrow a, b, \text{not } d\})$.

5 Syntactic Transformation for WMDLPs

We next present a syntactical transformation that, given a WMDLP \mathcal{P} , produces a generalized logic program whose stable models coincide with the stable models of \mathcal{P} . Thus this transformation provides the grounds for implementing WMDLPs. The transformation is established based on the proven correct syntactical transformation for the original definition of MDLP over DAGs given in [11]. The transformation is based on the following definitions.

Given a set \mathcal{K} of propositional variables and a set V of states, we write \mathcal{K}^* to indicate the following set of propositional variables:

$$\mathcal{K}^* = \mathcal{K} \cup \{A^-, A_s, A_s^-, A_{P_s}, A_{P_s}^-, \text{reject}(A_s), \text{reject}(A_s^-)\}$$

for every atom $A \in \mathcal{K}$ and state $s \in V \cup \{s_0\}$, where $s_0 \notin V$ is a reserved state called *initial state*. Let \mathcal{L}^* be the propositional language generated by \mathcal{K}^* . In the remaining of the paper we assume that every WDAG D does not contain the initial state s_0 among its vertices. Instead, s_0 belongs to the relevancy WDAG of D , defined next.

Definition 7. Let $D = (V, E, w)$ be a WDAG and $s \in V$ a state. The relevancy WDAG of D wrt. a state s is the WDAG $D' = (V', E', w')$ where:

$$\begin{aligned} X &= \{v \mid v \in V \text{ and } v \preceq s\} \\ Y &= \{(v_1, v_2) \mid (v_1, v_2) \in E \text{ and } v_2 \in X\} \\ V' &= X \cup \{s_0\} \\ E' &= Y \cup \{(s_0, v) \mid \text{for every vertex } v \in X \text{ that is a source}\} \\ w'(e) &= \begin{cases} w(e) & \text{if } e \in Y \\ 0.1 & \text{if } e \in (E' - Y) \end{cases} \end{aligned}$$

The relevancy WDAG of D wrt. a state s is the subgraph of D consisting of all vertices and edges contained in all paths to s together with the initial state s_0 and the set of edges (s_0, v) connecting the initial state to all the sources v in X . Note that the value 0.1 of the weight associated to every edge of the form (s_0, v) is irrelevant since that is the unique edge incoming to v (by construction).

Since the relevancy WDAGs contain the initial state s_0 and new edges outgoing from it, we need to define the prevalence relation for relevancy WDAGs. The idea is to let every vertex of a relevancy WDAG prevail s_0 wrt. any other vertex.

Definition 8. Let $D = (V, E, w)$ be a WDAG and \triangleleft_s the strong prevalence relation wrt. a state $s \in V$ over D . Let $D' = (V', E', w')$ be the relevancy WDAG of D wrt. s . Then, the strong prevalence relation \diamond_s wrt. s over D' is defined as:

$$\begin{aligned} \forall v_1, v_2 \in V'. v_1 \triangleleft_s v_2 &\Rightarrow v_1 \diamond_s v_2 \\ \forall v \in V'. s_0 \neq v &\Rightarrow s_0 \diamond_s v \end{aligned}$$

The following example illustrates the use of relevancy WDAGs.

Example 9. Let $D = (V, E, w)$ be the WDAG of Example 4. The relevancy WDAG D' of D wrt. v_1 is $D' = (V', E', w')$ where $V' = \{s_0, v_1, v_2, v_3, v_4, v_5\}$ and E' consists of the edges $e_1 = (v_2, v_1)$, $e_2 = (v_3, v_1)$, $e_3 = (v_4, v_2)$, $e_4 = (v_5, v_3)$, $e_5 = (v_4, v_3)$, $e_6 = (v_5, v_2)$, $e_7 = (s_0, v_4)$ and $e_8 = (s_0, v_5)$. The weight function is $w'(e_1) = w'(e_2) = w'(e_3) = w'(e_4) = w'(e_7) = w'(e_8) = 0.1$ and

$w'(e_5) = w'(e_6) = 0.2$. Thus, the strong prevalence relation $\overset{v_1}{\diamond}$ over D' wrt. v_1 is defined as: $v_2 \overset{v_1}{\diamond} v_1, v_3 \overset{v_1}{\diamond} v_1, v_4 \overset{v_1}{\diamond} v_1, v_5 \overset{v_1}{\diamond} v_1, s_0 \overset{v_1}{\diamond} v_1, v_4 \overset{v_1}{\diamond} v_2, v_5 \overset{v_1}{\diamond} v_2, s_0 \overset{v_1}{\diamond} v_2, v_4 \overset{v_1}{\diamond} v_3, v_5 \overset{v_1}{\diamond} v_3, s_0 \overset{v_1}{\diamond} v_3, s_0 \overset{v_1}{\diamond} v_4$ and $s_0 \overset{v_1}{\diamond} v_5$.

The following proposition establishes that when determining the stable models of a WMDLP \mathcal{P} at a state s , we can restrict our attention to the part of \mathcal{P} corresponding to the relevancy WDAG wrt. s .

Proposition 1. *Let $\mathcal{P} = (\mathcal{P}_D, D)$ be a WMDLP. Suppose that $D = (V, E, w)$ and $\mathcal{P}_D = \{P_v \mid v \in V\}$. Let $s \in V$ be a state. Let $\mathcal{P}' = (\mathcal{P}_{D'}, D')$ be a WMDLP such that $D' = (V', E', w')$ is the relevancy WDAG of D wrt. s , and $\mathcal{P}_{D'} = \{P_v \mid v \in V'\}$ where $P_{s_0} = \{\}$. Then, M is a stable model of \mathcal{P} at state s iff M is a stable model of \mathcal{P}' at state s .*

Note that the program P_{s_0} associated to the initial state does not contain any rule.

The following definition presents a syntactic transformation that allows one to map WMDLPs into generalized logic programs. Such a transformation is based on the syntactic transformation for MDLPs [11]. They differ only in the treatment of the rejection rules.

Definition 9 (Mapping Φ). *Let $\mathcal{P} = (\mathcal{P}_D, D)$ be a WMDLP over the propositional language \mathcal{L} . Suppose that $D = (V, E, w)$. Given a state $s \in V$, the generalized logic program $\Phi(\mathcal{P}, s)$ consists of the following generalized rules over the language \mathcal{L}^* .*

Let $\mathcal{P}' = (\mathcal{P}_{D'}, D')$ be a WMDLP where $D' = (V', E', w')$ is the relevancy WDAG of D wrt. s , and $\mathcal{P}_{D'} = \{P_v \mid v \in V'\}$ where $P_{s_0} = \{\}$.

(RP) Rewritten program clauses:

$$A_{P_v} \leftarrow B_1, \dots, B_m, C_1^-, \dots, C_n^- \quad (1)$$

or

$$A_{P_v}^- \leftarrow B_1, \dots, B_m, C_1^-, \dots, C_n^- \quad (2)$$

for any clause:

$$A \leftarrow B_1, \dots, B_m, \text{not } C_1, \dots, \text{not } C_n \quad (3)$$

respectively, for any clause:

$$\text{not } A \leftarrow B_1, \dots, B_m, \text{not } C_1, \dots, \text{not } C_n \quad (4)$$

in the program $P_v \in \mathcal{P}_{D'}$. The rewritten clauses are obtained from the original ones by replacing atoms A (respectively, the default atoms $\text{not } A$) occurring in their heads by the atoms A_{P_v} (respectively, $A_{P_v}^-$) and by replacing negative premises $\text{not } C$ by C^- .

(IR) Inheritance rules:

$$A_v \leftarrow A_u, \text{ not reject}(A_u) \quad (5)$$

$$A_v^- \leftarrow A_u^-, \text{ not reject}(A_u^-) \quad (6)$$

for every atom $A \in \mathcal{K}$ and every edge $(u, v) \in E'$. The inheritance rules say that an atom A is true (respectively, false) in the state $v \in V'$ if it is true (respectively, false) in any ancestor state u and it is not rejected, i.e., forced to be false (respectively, true).

(RR) Rejection Rules:

$$\text{reject}(A_u^-) \leftarrow A_{P_v} \quad (7)$$

$$\text{reject}(A_u) \leftarrow A_{P_v}^- \quad (8)$$

for every atom $A \in \mathcal{K}$ and every state $u, v \in V'$ such that $u \overset{s}{\diamond} v$, where $\overset{s}{\diamond}$ is the strong prevalence relation wrt. s over the relevancy $\overset{s}{\text{WDAG}} D'$. The rejection rules say that if an atom A is true (respectively, false) in the program P_v , then it rejects inheritance of any false (respectively, true) atom of any state u that is strongly prevailed by v wrt. s .

(UR) Update rules:

$$A_v \leftarrow A_{P_v} \quad (9)$$

$$A_v^- \leftarrow A_{P_v}^- \quad (10)$$

for every atom $A \in \mathcal{K}$ and every state $v \in V'$. The update rules state that an atom A must be true (respectively, false) in the state v if it is true (respectively, false) in the program P_v .

(DR) Default Rules:

$$A_{s_0}^- \quad (11)$$

for every atom $A \in \mathcal{K}$. Default rules describe the initial state s_0 by making all atoms initially false.

(CS_s) Current State Rules:

$$A \leftarrow A_s \quad (12)$$

$$A^- \leftarrow A_s^- \quad (13)$$

$$\text{not } A \leftarrow A_s^- \quad (14)$$

for every atom $A \in \mathcal{K}$. Current state rules specify the current state s in which the program is being evaluated and determine the values of the atoms A and A^- , and the default atom $\text{not } A$.

The following result establishes the correctness of the syntactic transformation.

Theorem 1. *Given a WMDLP \mathcal{P} over the language \mathcal{L} , the stable models of $\Phi(\mathcal{P}, s)$, restricted to \mathcal{L} , coincide with the stable models of \mathcal{P} at state s .*

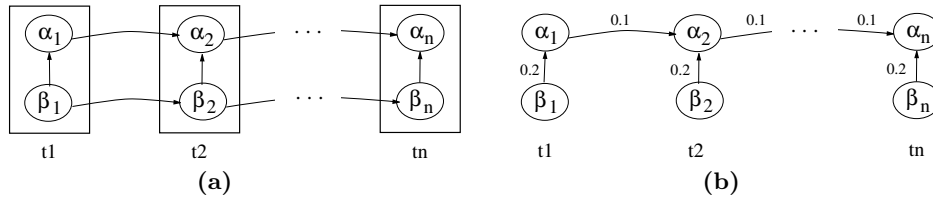


Fig. 4. (a) Temporal evolution of an agent society (b) Time prevailment

6 Modelling Updatable Information Systems

In [10] Leite discusses the representation of updatable information systems. These systems possess information about some aspect of the world, and are able to evolve by updating their information to reflect the dynamic behavior of the world. For simplicity, he considers information systems based on bi-dimensional representation structures: a *hierarchical dimension* and a *temporal dimension*. However, the approach can equally be applied to systems based on multi-dimensional structures. An example in the context of multi-agent systems is the representation of an agent society. Suppose that each agent has a well-defined role within the society, and the role structure of the society is hierarchical. Consider a society that consists of two agents α and β , where the role of α is predominant over the role of β . This means that the information of α prevails over the information of β . At every time stamp, an agent may receive new information and thereby evolve to the next state. An agent may acquire new information via its sensing actions, communication acts, and so on. Fig. 4a illustrates the temporal dimension of the agent society starting from time stamp t_1 until time stamp t_n . At each time stamp t_i , the agents are both at the same state i , represented as α_i and β_i . The hierarchical dimension of the agents' roles is depicted by an arrow from β_i to α_i , for each time stamp t_i . The new incoming information that α and β receive at a state i is represented by the generalized logic programs P_{α_i} and P_{β_i} , respectively.

Leite illustrates a number of ways to correlate the hierarchical and temporal dimension of the agent society by means of MDLPs, among which the time prevailing representation. We illustrate how this representation can be formalized via WMDLPs, and compare it with the solution based on MDLPs given in [10].

Consider the case where the temporal dimension prevails over the hierarchical dimension. Consequently, any rule defined in a more recent time overrides any older rule irrespective of hierarchy. Such a situation can be formalized via a WDAG as depicted in Fig. 4b. Here the temporal dimension is expressed by connecting the states at the top of the hierarchy. The edges formalizing the hierarchical dimension have a weight greater than the weight associated to the edges representing the temporal dimension. This allows the rules defined in a state with a more recent time stamp prevail the rules defined in any state with an

older time stamp. In this setting, the conflicts due to the equal role representation do not arise. This is shown by the next example.

Example 10. Let $\mathcal{P} = (\mathcal{P}_D, D)$ be the WMDLP depicted in Fig. 4b with $n = 2$. Let $D = (V, E, w)$ where $V = \{\alpha_1, \alpha_2, \beta_1, \beta_2\}$. E consists of $e_1 = (\beta_1, \alpha_1)$, $e_2 = (\beta_2, \alpha_2)$ and $e_3 = (\alpha_1, \alpha_2)$. Let $w(e_1) = w(e_2) = 0.2$, and $w(e_3) = 0.1$. Let $\mathcal{P}_D = \{P_{\alpha_1}, P_{\alpha_2}, P_{\beta_1}, P_{\beta_2}\}$ with $P_{\alpha_2} = P_{\beta_1} = \{\}$, $P_{\alpha_1} = \{a\}$ and $P_{\beta_2} = \{\text{not } a\}$. Then, there exists a (unique) stable model $M = \{\text{not } a\}$ of \mathcal{P} at state α_2 . In fact, we have that $Q(\mathcal{P}, \alpha_2) = \{a; \text{not } a\}$ and $\text{Default}(Q(\mathcal{P}, \alpha_2), M) = \{\}$. Now, since it holds that $\alpha_1 \triangleleft_{\alpha_2} \beta_2$, we have that $\text{Reject}(\mathcal{P}, \alpha_2, M) = \{a\}$. Finally, it holds that $M = \text{least}(\{\text{not } a\})$.

The formalization of the time prevailing representation via WMDLPs differs from the one based on MDLPs since the first employs WDAGs while the second DAGs. Employing WDAGs allows one to consider only the kind of prevailment one wants to enforce, and to abstract away from the structural details of how to achieve it (cf. the solution based on DAGs proposed in [10]).

7 Agent Societies Based on Confidence Factors

We describe a society whose agents have the ability to associate a confidence factor (i) to the information incoming from other agents, (ii) to the information outgoing to other agents, and (iii) to its own information. Confidence factors can be used for a number of purposes, for example, to indicate the level of trust/confidence of an agent towards another agent, the relevance of the information of a source agent, the confidence that an agent has about its own information, the strength with which an agent supports its information towards another agent, etc. The following example illustrates a situation where agents have different levels of trust and belief.

Example 11. Consider a society consisting of three agents Adam, Bob and Carl. Suppose that Adam needs to buy a car, and considers buying a Fiat. Adam will buy a Fiat, if it is a good car. Adam believes that a Fiat is not a good car. Adam being a non-expert on cars has low confidence about himself. Both Bob and Carl believe that a Fiat is a good car. Bob who is Fiat car seller, always insists (with customers) that a Fiat is a good car. Assume that Adam does not rely too much on what car sellers say, and has instead high confidence on what his friend Carl believes.

To formalize an agent society based on confidence factors, we need a way to represent the logical structure of the society. We do so via the notion of CDAGs.

Definition 10. Let \mathbb{R}^+ be a set of positive real numbers. A directed acyclic graph with confidence factors (CDAG) is a tuple (V, E, w_s, w_i, w_t, w) where V is a set of vertices, E a set of edges containing the edge (v, v) for any vertex $v \in V$, $w_s : V \rightarrow \mathbb{R}^+$, $w_i : E \rightarrow \mathbb{R}^+$, $w_t : E \rightarrow \mathbb{R}^+$, and $w : E \rightarrow \mathbb{R}^+$ a numeric function defined in terms of w_s , w_i , and w_t .

CDAGs extend WDAGs to allow cycle edges of the form (v, v) only, and to include several weight functions. Given an edge $e = (v_1, v_2)$, the weight functions $w_i(e)$ and $w_t(e)$ represent the confidence factor given by the initial vertex v_1 and terminal vertex v_2 to e . Thus, $w_t(e)$ represents the confidence factor given by v_2 to the information incoming from v_1 (step (i) above); $w_i(e)$ represents the confidence factor given by v_1 to the information outgoing to v_2 (step (ii) above). Given a vertex v , the weight function $w_s(v)$ indicates the level of confidence that v has of itself (self-confidence factor). The final weight of an edge is given by the function w .

To formalize the information inherent to the society, we employ generalized logic programs. As we did for WMDLPs, we associate a generalized logic program to the vertices of CDAGs.

Definition 11. *Let \mathcal{L} be a propositional language. A multi-dimensional dynamic logic program with confidence factors (CMDLP) \mathcal{P} is a pair (\mathcal{P}_D, D) , where $D = (V, E, w_s, w_i, w_t, w)$ is a CDAG and $\mathcal{P}_D = \{P_v \mid v \in V\}$ is a set of generalized logic programs over \mathcal{L} indexed by the vertices $v \in V$.*

Example 12. The car problem of Example 11 can be expressed by a CMDLP $\mathcal{P} = (\mathcal{P}_D, D)$ as illustrated in Fig. 5a. We use a, b and c to denote Adam, Bob and Carl. $D = (V, E, w_s, w_i, w_t, w)$ with $V = \{a, b, c\}$, $E = \{(b, a), (c, a), (a, a), (b, b), (c, c)\}$, $w_s(a) = 0.7$, $w_s(b) = 0.6$, $w_s(c) = 0.9$, $w_i((b, a)) = 0.9$, $w_t((b, a)) = 0.3$, $w_i((c, a)) = 0.7$, and $w_t((c, a)) = 0.8$. Suppose the the weight function w is defined as:

$$w(e) = \begin{cases} w_s(v) & \text{if } e = (v, v) \\ \frac{w_i(e) + w_t(e)}{2} & \text{otherwise} \end{cases}$$

$\mathcal{P}_D = \{P_a, P_b, P_c\}$ where $P_a = \{buy(fiat) \leftarrow good_car(fiat); not\ good_car(fiat)\}$ and $P_b = P_c = \{good_car(fiat)\}$.

The intended model of Adam is $M_a = \{buy(fiat), good_car(fiat)\}$. Since Adam has low self-confidence, he believes whatever Carl believes. In fact, the final weight of the edge (c, a) is greater than the self-confidence of Adam, i.e., $w((c, a))$ is greater than $w_s(a)$. (Note that the self-confidence of Adam is greater than the final weight of the edge (b, a) .)

The declarative and procedural semantics of CMDLPs is given in terms of the semantics of WMDLPs. To do so, we first show how to code CMDLPs into WMDLP.

Definition 12 (Mapping II). *Let $\mathcal{P} = (\mathcal{P}_D, D)$ be a CMDLP. Suppose that $D = (V, E, w_s, w_i, w_t, w)$ and $\mathcal{P}_D = \{P_v \mid v \in V\}$. Then, the WMDLP induced*

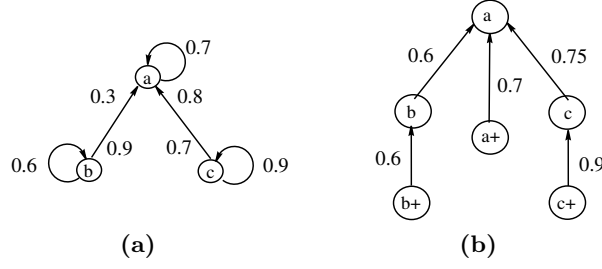


Fig. 5. (a) CDAG D of Example 12 (b) WDAG $\Pi(\mathcal{P})$

by \mathcal{P} is the WMDLP $\Pi(\mathcal{P}) = (\mathcal{P}_{D_2}, D_2)$ where:

$$\begin{aligned}
 D_2 &= (V_2, E_2, w_2) \\
 V_2 &= V \cup \{v^+ \mid \text{for any } v \in V\} \\
 E_2 &= \{(u, v) \in E \mid u \neq v\} \cup \{(v^+, v) \mid \text{for any } v \in V\} \\
 w_2(e) &= \begin{cases} w((v, v)) & \text{if } e = (v^+, v) \\ w(e) & \text{otherwise} \end{cases} \\
 \mathcal{P}_{D_2} &= \{Q_{v^+} \mid \text{for any } v^+ \in (V_2 - V) \text{ with } Q_{v^+} = P_v\} \cup \\
 &\quad \{Q_v \mid \text{for any } v \in V \text{ with } Q_v = \{\}\}
 \end{aligned}$$

The CMDLP \mathcal{P} of Example 12 is coded into the WMDLP depicted in Fig. 5b. The new vertices v^+ are needed to remove the edges (v, v) from the CDAG. This is achieved by simply assigning the weight $w_s(v)$ to the edges (v^+, v) . Now, we can give the declarative semantics for CMDLPs.

Definition 13. Let \mathcal{P} be a CMDLP, and s a state in \mathcal{P} . An interpretation M is a stable model of \mathcal{P} at state s iff M is a stable model of $\Pi(\mathcal{P})$ at state s .

Example 13. Consider the CMDLP \mathcal{P} of Example 12. Then, the interpretation $M_a = \{buy(fiat), good_car(fiat)\}$ is a stable model of \mathcal{P} at state a since it is a stable model of $\Pi(\mathcal{P})$ at state a .

8 Concluding Remarks

Our framework builds on the notion of prevalence of vertices. However, other notions of prevalence can be accommodated within it. To do so it is necessary to incorporate these new notions both at the semantical level, that is to modify the definition of $Reject(\mathcal{P}, s, M)$ in Def. 6, and at the syntactical level, that is to modify the Rejection Rules (RR) in Def. 9 to reflect the new relation.

An interesting notion that can be incorporated is to represent *agent societies based on voting*. The idea is to engineer societies where, although each agent

has a role which drives its behavior, some degree of freedom can be provided to the agents. One possibility in this direction is to incorporate a *voting system* in the society. The voting system can be based on the incoming edges of a certain node. Thus rules can be rejected because they are outweighed or outvoted, e.g., by opting for the best positive or negative average, or even for none. A receiving node, once it weighs the incoming edges, “commits” to the ones not rejected by the voting. This can be achieved by defining the weight function w of a WDAG as $w : E \times L \rightarrow \mathbb{R}^+$ mapping the edges in E and the literals in L to positive real numbers in \mathbb{R}^+ . Then we can adopt a prevalence relation of the form $u \underset{s}{\overset{a}{\triangleleft}} v$ stating that a vertex v prevails a vertex u wrt. a vertex s in what concerns the literal a .

A weighed voting can be employed to resolve multiple (or simply paired) contradictions. More sophisticated weighing schemes, including the introduction of (outgoing) rule weights, and their combination with the (incoming) edge weights, shall be the subject of future research.

We have presented a logical framework that allows one to model structures of epistemic agents. In doing so, we have first introduced the notion of WDAG that extends DAGs to associate weights to every edge of the graph. Then, we have presented the logical framework with the corresponding declarative and operational semantics, and we have given results of correctness. The framework having a formal semantics will allow us to study and prove the properties of structures for epistemic agents.

In a previous work [7], we explored how to explicitly represent organizational structures in epistemic multi-agent systems (eMAS), including groups of agents, institutions, and complex organizational structures for agent societies. Here, we have illustrated the usage of WMDLPs to represent agent societies based on confidence factors and voting. For simplicity, we have focused only on the agent structures, rather than on the agent theories. Therefore, we haven’t considered the dynamic aspects of the agent knowledge like, for example, when an agent updates its knowledge to incorporate new incoming information (via updates). These aspects are discussed in [6].

An interesting direction for future work is to represent the logical framework within the theory of the agent members of the society. That is, we can code the graph structure of the agents and the links among them into the theory of the agents themselves. Doing so will empower the agents with the ability to reason about and to modify the structure of their own graph together with the general group structure comprising the other agents. At the level of each single agent, declaratively expressing the graph structure enables that agent to reason over it in a declarative way. At the level of the group structure, this ability will permit the managing of open societies where agents can enter/leave the society. This in fact can be achieved by updating the graph structure representing the group by adding/removing vertices and edges. Therefore, encoding the graph structure within the language of the agents makes the system updatable to capture the dynamic aspects of the system, i.e., of the open society.

References

1. J. J. Alferes, P. Dell'Acqua, and L. M. Pereira. A compilation of updates plus preferences. In S. Flesca, S. Greco, N. Leone, and G. Ianni, editors, *Logics in Artificial Intelligence*, LNAI 2424, pages 62–74, Berlin, 2002. Springer-Verlag.
2. J. J. Alferes, J. A. Leite, L. M. Pereira, H. Przymusińska, and T. C. Przymusiński. Dynamic updates of non-monotonic knowledge bases. *The J. of Logic Programming*, 45(1-3):43–70, 2000. A short version titled *Dynamic Logic Programming* appeared in A. Cohn and L. Schubert (eds.), *KR'98*, Morgan Kaufmann.
3. A. Artikis and G. Pitt. A formal model of open agent societies. *Proc. of Autonomous Agents*, 2001.
4. P. Dell'Acqua, J. A. Leite, and L. M. Pereira. Evolving multi-agent viewpoints - an architecture. In P. Brazdil and A. Jorge, editors, *Progress in Artificial Intelligence, 10th Portuguese Int. Conf. on Artificial Intelligence (EPIA '01)*, LNAI 2258, pages 169–182. Springer-Verlag, 2001.
5. P. Dell'Acqua, U. Nilsson, and L. M. Pereira. A logic based asynchronous multi-agent system. *Computational Logic in Multi-Agent Systems (CLIMA02)*. *Electronic Notes in Theoretical Computer Science (ENTCS)*, Vol. 70, Issue 5, 2002.
6. P. Dell'Acqua and L. M. Pereira. Preferring and updating in abductive multi-agent systems. In A. Omicini, P. Petta, and R. Tolksdorf, editors, *Engineering Societies in the Agents' World (ESAW 2001)*, LNAI 2203, pages 57–73. Springer-Verlag, 2001.
7. P. Dell'Acqua and L. M. Pereira. A Logical Framework for Modelling eMAS. In V. Dahl and P. Wadler, editors, *Fifth Int. Symp. on Practical Aspects of Declarative Languages (PADL03)*, LNCS 2562, pages 241–255, Berlin, 2003. Springer-Verlag.
8. P. Dell'Acqua and L. M. Pereira. Preferring and updating in logic-based agents. In O. Bartenstein, U. Geske, M. Hannebauer, and O. Yoshie, editors, *Web-Knowledge Management and Decision Support. Selected Papers from the 14th Int. Conf. on Applications of Prolog (INAP)*, LNAI 2543, pages 70–85, Berlin, 2003. Springer-Verlag.
9. M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In R. Kowalski and K. A. Bowen, editors, *ICLP'88*, pages 1070–1080. MIT Press, 1988.
10. J. A. Leite. *Evolving Knowledge Bases*. *Frontiers in Artificial Intelligence and Applications*. IOS Press, Amsterdam, The Netherlands, 2003.
11. J. A. Leite, J. J. Alferes, and L. M. Pereira. Multi-dimensional dynamic logic programming. In F. Sadri and K. Satoh, editors, *Procs. of the CL-2000 Workshop on Computational Logic in Multi-Agent Systems (CLIMA'00)*, pages 17–26, 2000.
12. V. Lifschitz and T. Woo. Answer sets in general non-monotonic reasoning (preliminary report). In B. Nebel, C. Rich, and W. Swartout, editors, *KR'92*. Morgan-Kaufmann, 1992.
13. F. Zambonelli. Abstractions and infrastructures for the design and development of mobile agent organizations. In M. J. Wooldridge, G. Weiß, and P. Ciancarini, editors, *Agent-Oriented Software Engineering II, Second International Workshop, AOSE 2001*, LNCS 2222, pages 245–262, Berlin, 2001. Springer-Verlag.
14. F. Zambonelli, N. R. Jennings, and M. Wooldridge. Organisational abstractions for the analysis and design of multi-agent systems. In P. Ciancarini and M. Wooldridge, editors, *Agent-Oriented Software Engineering*, LNCS 1957, pages 127–141, Berlin, 2001. Springer-Verlag.