A Logical Framework for Modelling eMAS

Pierangelo Dell'Acqua[*] [†] Luís Moniz Pereira[†]

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

[†] Centro de Inteligência Artificial - CENTRIA
Departamento de Informática, Faculdade de Ciências e Tecnologia
Universidade Nova de Lisboa, 2829-516 Caparica, Portugal
lmp@di.fct.unl.pt

Abstract. We investigate how to explicitly represent organizational structures in epistemic multi-agent systems (eMAS). We introduce a logical framework $\mathcal{F}$ suitable for representing organizational structures for epistemic agents, and provide its declarative and procedural semantics. We show how a number of organizational structures can be represented in $\mathcal{F}$ and discuss their properties.

1 Motivation

In previous papers [6, 8, 9] 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). The knowledge state of each agent is represented by an abductive logic program in which it is possible to express rules, integrity constraints, active rules, and priorities among rules. This allows the agents to reason, to react to the environment, to prefer among several alternatives, to update both beliefs and reactions, and to abduce hypotheses to explain observations. 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 [7] we presented a framework for handling the communication among epistemic agents asynchronously.

Within the multi-agent system framework proposed in [8] we can represent coalitions of agents, groups, teams implicitly based on the internal mental states of the members. It is advocated, especially in open multi-agent systems (cf. [3, 11, 16]), 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.

Organizational structures and rules are required to provide control over the member agents with respect to their actions and interactions. Castelfranchi [4] argues that social order is a major problem in multi-agent systems. The idea of a total control and a technical prevention against chaos, conflicts and deception in computers is unrealistic and even self-defeating in some cases, e.g. in the building up of trust. He states that there is some illusion in Computer Science about solving this problem by rigid formalization and rules, constraining infra-structures, security devices, etc. and there is skepticism or irritation towards more soft and social approaches, that leave more room to spontaneous emergence, or to decentralized control, or to normative "stuff" which is not externally imposed but internally managed by the agents. He claims that the most effective solution to this problem is social modelling and that it should leave some flexibility and try to deal with emergent and spontaneous form of organizational building (that is, decentralized and autonomous social control). The problem with this approach is that of modelling the feedback from the global results to the local/individual layer.

Zambonelli [15] states that modelling and engineering interactions in complex and open multi-agent systems cannot simply rely on the agent capabilities for communicating and of acting according to the expectations of each agent of the system. 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.

In this paper, with the above purpose in mind, we elaborate over the approach proposed in [8], and usher in the more flexible notion of weighted directed acyclic graphs (WDAGs), generalizing the notion of DAGs presented in [12], and permitting us to address the issues at hand by assigning a measure of strength to the knowledge sharing relationships represented by the edges. We introduce a logical framework $\mathcal{F}$, provide its declarative semantics, and sketch its procedural semantics in some detail. We next show how several organizational structures in multi-agent systems (eMAS) for epistemic agents can indeed be explicitly represented in $\mathcal{F}$. $\mathcal{F}$ having a formal semantics enables us to study and prove properties of eMAS structures.

2 Background

To represent negative information in logic programs, we need a language that allows default negation $not\,A$ not only in premises of rules but also in their heads[1]. By a *generalized logic program* P over a language $\mathcal{L}$ we mean a finite or infinite set of propositional rules r of the form $L_0 \leftarrow L_1 \wedge \ldots \wedge L_n$, where each L_i is a literal (i.e. an atom A or its default negation $not\,A$). It is convenient to syntactically represent generalized logic programs as propositional Horn theories. In particular, we represent default negation $not\,A$ as a propositional variable. If r is a rule, by $head(r)$ we mean L_0, and by $body(r)$ we mean $L_1 \wedge \ldots \wedge L_n$. If $head(r) = A$ (resp. $head(r) = not\,A$) then $not\,head(r) = not\,A$ (resp.

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

$not\, head(r) = A$). 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 $M^+ = \{A : A \in M\}$ and $M^- = \{not\, A : not\, A \in M\}$. 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 $M = least(P \cup M^-)$. The class of generalized logic programs can be viewed as a special case of yet broader classes of programs, introduced earlier in [14], and, for the special case of normal programs, their semantics coincides with the stable models semantics [10].

3 Weighted 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, w) called *edges*, where v and w are vertices in V. The vertex v is the *initial vertex* of the edge and w 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 weighted directed graph is a directed graph in which a positive real number has been assigned to each edge. A *weighted directed graph* is a tuple (V, E, w) where V is a set of vertices, E a set of edges and $w : E \to \mathbb{R}^+$ a function mapping the edges in E to positive real numbers in $\mathbb{R}^+$. 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 $v_1 \prec v_n$ with $1 < n$ we write the sequence $v_1, v_2, \ldots, v_n$ of all vertices occurring in it. A *cycle* is a path in which the initial vertex of the path is also the terminal vertex.

Definition 1 (WDAG). *A weighted directed acyclic graph (WDAG) is a weighted directed graph that does not contain any cycles.*

Definition 2 (Path dominance). *Let $D = (V, E, w)$ be a WDAG and $a_1 \prec a_n$ $(1 < n)$ a path with vertices $a_1, a_2, \ldots, a_n$. Then $a_1 \prec a_n$ is a dominant path iff there exists no other path $b_1, b_2, \ldots, b_m$ $(1 < m)$ such that $b_1 = a_1$, $b_m = a_n$ and there exist i, j with $1 < i \leq n, 1 < j \leq m$ such that $a_i = b_j$ and $w((a_{i-1}, a_i)) < w((b_{j-1}, b_j))$.*

Example 1. Let $D = (V, E, w)$ be a WDAG, 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 v_3, v_1 and the path v_3, v_2, v_4, v_1.

Definition 3 (Prevalence wrt. a vertex a_n). *Let $D = (V, E, w)$ be a WDAG and $a_1 \prec a_n$ a dominant path with vertices $a_1, a_2, \ldots, a_n$. Then, every vertex a_i prevails a_1 wrt. a_n, for every $1 < i \leq n$. If in addition there exists a path $b_1 \prec a_i$ with vertices $b_1, \ldots, b_m, a_i$ $(1 \leq m)$, for some $1 < i \leq n$, and $w((a_{i-1}, a_i)) < w((b_m, a_i))$, then every vertex b_j prevails a_1 wrt. a_n, for every $1 \leq j \leq m$.*

In the following, we write $v_2 \underset{s}{\lhd} v_1$ to indicate that v_1 prevails v_2 wrt. s.

Example 2. Let $D = (V, E, w)$ 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, for instance, the following prevalences hold: $v_4 \underset{v_1}{\lhd} v_1$, $v_2 \underset{v_1}{\lhd} v_1$, $v_6 \underset{v_1}{\lhd} v_5$, $v_2 \underset{v_1}{\lhd} v_5$ and $v_4 \underset{v_1}{\lhd} v_6$.

Example 3. Let $D = (V, E, w)$ 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 $w(e_4) = 0.2$. The following prevalences holds: $v_3 \underset{v_1}{\lhd} v_1$, $v_4 \underset{v_1}{\lhd} v_1$, $v_2 \underset{v_1}{\lhd} v_4$ and $v_3 \underset{v_1}{\lhd} v_4$. The vertex v_3 does not prevail v_4 since the unique dominant path from v_4 to v_1 is v_4, v_1.

The next notion allows us to connect vertices of distinct WDAGs.

Definition 4 (Link). *Let $D_1 = (V_1, E_1, w_1)$ and $D_2 = (V_2, E_2, w_2)$ be WDAGs. A link from D_1 to D_2 is an ordered pair (v_1, v_2) whose initial vertex v_1 belongs to V_1 and terminal vertex v_2 belongs to V_2.*

To simplify the notation, in the remaining of the paper we assume that each vertex in a WDAG has a unique name, that is, given two arbitrary WDAGs D_1 and D_2 we assume that $V_1 \cap V_2 = \{\}$.

Definition 5 (WDAGs joining). *Let $D = \{D_1, \ldots, D_n\}$ be a set of WDAGs, L a set of links over D and $w_L : L \to \mathbb{R}^+$ a function. Assume that $D_i = (V_i, E_i, w_i)$ for every $1 \leq i \leq n$. The joining of the WDAGs in D w.r.t. L and w_L, written as $\sqcup(D, L, w_L)$, is the WDAG (V, E, w) where:*

$$V = \bigcup_{1 \leq i \leq n} V_i \qquad E = \bigcup_{1 \leq i \leq n} E_i \cup L \quad \textit{and} \quad w(e) = \begin{cases} w_i(e) & \textit{if } e \in E_i \\ w_L(e) & \textit{if } e \in L \end{cases}$$

Example 4. Let $D = \{D_1, D_2\}$, where $D_1 = (\{a_1, a_2, a_3\}, \{(a_1, a_2), (a_2, a_3)\}, w_1)$ and $D_2 = (\{b_1, b_2, b_3\}, \{(b_1, b_3), (b_2, b_3)\}, w_2)$. Let $L = \{(a_2, b_1), (b_2, a_3)\}$ and w_L a weight function over L. Then $(a_1, a_2), (a_2, b_1), (b_1, b_3)$ is a path $a_1 \prec b_3$ over $\sqcup(D, L, w_L)$.

4 Logic Framework

In this section we present a logical framework suitable to formalize organizational structures for epistemic agents. We start by defining the notion of generalized logic programs indexed by WDAGs. The next definition extends the original definition of MDLP (presented in [12, 13]) to take into consideration WDAGs.

Definition 6 (Multi-dimensional Dynamic Logic Program). *Let $\mathcal{L}$ be a propositional language. A Multi-dimensional Dynamic Logic Program (MDLP), $\mathcal{P}$, is a pair $(\mathcal{P}_D, D)$, where $D = (V, E, w)$ is a WDAG and $\mathcal{P}_D = \{P_v : v \in V\}$ is a set of generalized logic programs over the language $\mathcal{L}$ indexed by the vertices $v \in V$. We call* states *such vertices of D. For simplicity, we often leave the language $\mathcal{L}$ implicit.*

To characterize the models of $\mathcal{P}$ at any given state we will keep to the basic intuition of logic program updates, whereby an interpretation is a stable model of the update of a program P by a program U iff it is a stable model of a program consisting of the rules of U together with a subset of the rules of P comprised by those that are not rejected (do not carry over by inertia) due to their being overridden by the updating program U. With the introduction of WDAGs to index programs, a program may have more than a single ancestor. This has to be dealt with, the desired intuition being that a program $P_v \in \mathcal{P}_D$ can be used to reject rules of any program $P_u \in \mathcal{P}_D$ if v prevails u. Moreover, if an atom A is not defined in any program P_u in $\mathcal{P}_D$ ancestor of P_v, then its negation $not\, A$ is assumed by default in P_v.

Definition 7 (Stable Models of MDLP at state s). *Let $\mathcal{P} = (\mathcal{P}_D, D)$ be a MDLP where $\mathcal{P}_D = \{P_v : v \in V\}$ and $D = (V, E, w)$. Let s be a state in V. An interpretation M is a* stable model of $\mathcal{P}$ *at state s iff*

$$
\begin{aligned}
M &= least(O \cup Default(O, M)) \text{ where:} \\
Q &= \bigcup_{v \preceq s} P_v \\
Reject(s, M) &= \{r \in P_{v_2} : \exists r' \in P_{v_1}\, head(r) = not\, head(r'), M \vDash body(r'), \\
&\qquad \text{and } v_2 \underset{s}{\lhd} v_1\} \\
O &= Q - Reject(s, M) \\
Default(O, M) &= \{not\, A : \nexists r \in O, head(r) = A \text{ and } M \vDash body(r)\}.
\end{aligned}
$$

Intuitively, the set $Reject(s, M)$ 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 prevails v_2 wrt. s. Q contains all rules of all programs that are indexed by a state along all paths to state s, i.e. all rules that are potentially relevant to determine the semantics at state s. The set $Default(O, M)$ contains default negations $not\, A$ of all unsupported atoms A, i.e., those atoms A for which there is no rule in O (i.e. the non-rejected rules of Q) whose body is true in M.

Example 5. Let $\mathcal{P} = (\mathcal{P}_D, D)$ be a MDLP where $D = (V, E, w)$, $V = \{v_1, v_2, v_3\}$. E consists of $e_1 = (v_1, v_3)$ and $e_2 = (v_2, v_3)$. Let $\mathcal{P}_D = \{P_1, P_2, P_3\}$ with $P_1 = \{a\}$, $P_2 = \{not\, a\}$ and $P_3 = \{\}$. If $w(e_1) = w(e_2)$, then there exists no stable model M of $\mathcal{P}$ at state v_3. In fact, $Q = \{a, not\, a\}$ and $Reject(s, M) = \{\}$, $Default(O, M) = \{\}$, for any interpretation M. Thus, there exists no model M

such that $M = least(O)$. 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, $Reject(s, M) = \{not\, a\}$ as $v_2 \underset{v_3}{\lhd} v_1$, $Default(O, M) = \{\}$ and $M = least(\{a\})$.

Example 6. Let $\mathcal{P} = (\mathcal{P}_D, D)$ be a MDLP where $D = (V, E, w)$, $V = \{v_1, v_2, v_3, v_4\}$. 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 $\mathcal{P}_D = \{P_1, P_2, P_3, P_4\}$ with $P_1 = \{b\}$, $P_2 = \{a\}$, $P_3 = \{not\, a, d \leftarrow a \wedge b\}$ and $P_4 = \{not\, b\}$. The stable model of $\mathcal{P}$ at state v_1 is $\{a, b, d\}$. In fact, as it holds that $v_4 \underset{v_1}{\lhd} v_1$ and $v_3 \underset{v_1}{\lhd} v_2$, we have that $Reject(s, M) = \{not\, a, not\, b\}$ and $Default(O, M) = \{\}$.

We can now introduce the logical framework $\mathcal{F}$ and its declarative semantics. The idea of the approach is to formalize epistemic agents by MDLPs and to organize them into structures by means of $\mathcal{F}$.

Definition 8 (Logical Framework $\mathcal{F}$). *The logical framework $\mathcal{F}$ is a tuple $(\mathcal{A}, L, w_L)$ where $\mathcal{A} = \{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$ is a set of MDLPs $\mathcal{P}_i = (\mathcal{P}_{D_i}, D_i)$, L is a set of links over the D_is and $w_L : L \to \mathbb{R}^+$ is a function.*

Definition 9 (Joined MDLP). *Let $\mathcal{F} = (\mathcal{A}, L, w_L)$ be a logical framework. Assume that $\mathcal{A} = \{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$ and each $\mathcal{P}_i = (\mathcal{P}_{D_i}, D_i)$ for $1 \le i \le n$. The joined MDLP induced by $\mathcal{F}$, if it is a WDAG, is $\mathcal{P} = (\mathcal{P}_D, D)$ where $D = (V, E, w)$ is $\sqcup(\{D_1, \ldots, D_n\}, L, w_L)$ and $\mathcal{P}_D = \{P_v : v \in V$ and $P_v \in \bigcup_{1 \le i \le n} \mathcal{P}_{D_i}\}$.*

Definition 10 (Stable Models of $\mathcal{F}$ at state s). *Let $\mathcal{F} = (\mathcal{A}, L, w_L)$ be a logical framework, s be a state and $\mathcal{P}$ the joined MDLP induced by $\mathcal{F}$. An interpretation M is a stable model of $\mathcal{F}$ at state s iff M is a stable model of $\mathcal{P}$ at state s.*

5 Syntactic Transformation for WDAGs

By Def. 10, to have a proof procedure for the logical framework $\mathcal{F}$ we need a proof procedure for MDLPs. This can be achieved via a syntactical transformation that, given a multi-dimensional logic program $\mathcal{P}$, produces a generalized logic program whose stable models coincide with the stable models of $\mathcal{P}$. Thus this transformation provides the grounds for implementing $\mathcal{F}$.

The transformation is established based on the proven correct syntactical transformation for the original definition of MDLP over DAGs given in [12]. The original contains transformation among others the following two axioms.

- *Inheritance rules:* $A_v \leftarrow A_u \wedge not\ reject(A_u)$
  for all atoms A and edges (u, v). It states that an atom A is true in a state v if it is true in any ancestor state u and it is not rejected, i.e., forced to be false.

– *Rejection rules:* $reject(A_u) \leftarrow A^-_{P_v}$
for all atoms A and all vertices u, v such that $u \prec v$. The rejection rules say that if an atom A is false (represented by A^-) in the program P_v, then it rejects inheritance of any true atom A of any ancestor u.

To generate a transformation for MDLPs based on a WDAG $D = (V, E, w)$, we can explicitly represent the edges in E by $edge(u, v, x)$ for all edges $(u, v) \in E$ whose weight is x, and the prevalence relation $u \triangleleft v$ by $prevail(u, v)$. Then, we can substitute the axioms above with:

– *Inheritance rules*:* $A_v \leftarrow A_u \wedge not\ reject(A_u) \wedge edge(u, v, x)$
for all atoms A. It states that an atom A is true in a state v if it is true in a state u, it is not rejected, and there exists an edge x between u and v.
– *Rejection rules*:* $reject(A_u) \leftarrow A^-_{P_v} \wedge prevail(u, v)$
for all atoms A. The rejection rules say that if an atom A is false in the program P_v and v prevails over some a vertex u, then the rule rejects inheritance of any true atom A at vertex u.

6 Modelling Organizational Structures

Typically, agents operate in the context of multi-agent systems. Some of these systems can be understood as a computational society whose role is to allow its members to coexist in a shared environment and to pursue their goals. Several attempts to define organizational structures for agents have been proposed, e.g., coalitions, groups, teams and institutions. In this section we show how a number of organizational structures can be formalized within the proposed logical framework, and discuss some of their properties. The idea is to formalize both agents and organizational structures via MDLPs and to glue them together via $\mathcal{F}$. We assume that the WDAG of the MDLP of each agent is equipped with a sink[2] v, with an empty program P_v associated with it, indicating where the semantics of the agent can be determined. v is called the *inspection point* of the agent. Following Davidsson [5], agent societies can be categorized with respect to the following properties: *openness*, the possibility for agents to join the society without any restriction; *flexibility*, the degree to which agents are restricted in their behavior by the society; *stability*, predictability of the consequences of actions, and *trustfulness*, the extent to which the owners of the agents trust the society.

6.1 Groups

A *group* is a system of agents that are constrained in their mutual interactions. A group can be formalized in $\mathcal{F}$ in a flexible way: the behavior of the agents in the group can be restricted by the group itself to different degrees depending on its purpose. The formalization of the group supports also trustfulness. In fact, restricting the behavior of the members of the group with norms and regulations may enhance the degree of trust vis--vis the group.

[2] A vertex with out-valency 0.

Fig. 1 depicts a group of three agents: A, B and C. (The oval in the picture represents the group.) Suppose that A is the leader of the group.

We want to formalize a situation where B must operate (strictly) in accordance with A, while C has a certain degree of freedom. In this scenario, B and C are secretaries to A and both believe that it is not their duty to answer phone calls. In contrast, A believes that one of the duties of a secretary is indeed to answer the phone. This situation can be formalized as follows.

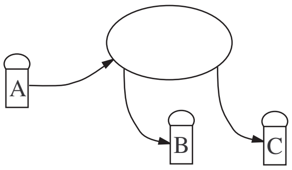

Fig. 1. Group of agents

We let $\mathcal{P}_A$, $\mathcal{P}_B$ and $\mathcal{P}_C$ be the MDLPs of the agents A, B and C, and $\mathcal{P}_G$ be the MDLP of the group. Let $\mathcal{P}_A = (\mathcal{P}_{D_A}, D_A)$ where $D_A = (\{v_1\}, \{\}, w_A)$ and $P_{v_1} = \{\mathit{answerPhone} \leftarrow \mathit{secretary} \wedge \mathit{phoneRing}\}$. Let $\mathcal{P}_B = (\mathcal{P}_{D_B}, D_B)$ where $D_B = (\{v_3, v_4\}, \{(v_4, v_3)\}, w_B)$, $w_B((v_4, v_3)) = 0.6$, $P_{v_3} = \{\}$ and $P_{v_4} = \{\mathit{phoneRing}, \mathit{secretary}, \mathit{not\ answerPhone}\}$. Let $\mathcal{P}_C = (\mathcal{P}_{D_C}, D_C)$ where $D_C = (\{v_5, v_6\}, \{(v_6, v_5)\}, w_C)$, $w_C((v_6, v_5) = 0.6$, $P_{v_6} = P_{v_4}$ and $P_{v_5} = P_{v_3}$. Let $\mathcal{P}_G = (\mathcal{P}_{D_G}, D_G)$ where $D_G = (\{v_2\}, \{\}, w_G)$ and $P_{v_2} = \{\}$.

$\mathcal{F} = (\mathcal{A}, L, w_L)$ where $\mathcal{A} = \{\mathcal{P}_A, \mathcal{P}_B, \mathcal{P}_C, \mathcal{P}_G\}$, $L = \{(v_1, v_2), (v_2, v_3), (v_2, v_5)\}$ and $w_L((v_1, v_2)) = 0.5$, $w_L((v_2, v_3)) = 0.7$ and $w_L((v_2, v_5)) = 0.5$.

Let v_3 and v_5 be the inspection points of the agents B and C, respectively. Then, according to the declarative semantics of $\mathcal{F}$, B will answer the phone, while C will not do it. In fact, the semantics of B, whose model is determined at its inspection point v_3, is $M_{v_3} = \{\mathit{phoneRing}, \mathit{secretary}, \mathit{answerPhone}\}$, while the semantics of C is $M_{v_5} = \{\mathit{phoneRing}, \mathit{secretary}, \mathit{not\ answerPhone}\}$. Although the theories of B and C are the same, their models are different because the edges (v_2, v_3) and (v_2, v_5) have different weights. M_{v_3} is a stable model of the joined MDLP $\mathcal{P}$ induced by $\mathcal{F}$ at state v_3 since we have that:

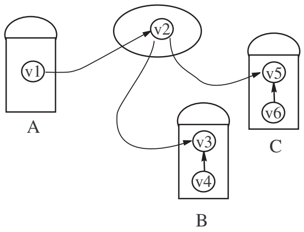

Fig. 2. Group of agents

$$
\begin{aligned}
M_{v_3} &= \mathit{least}(O \cup \mathit{Default}(O, M_{v_3})) \text{ where:}\\
Q &= \{\mathit{phoneRing}, \mathit{secretary}, \mathit{not\ answerPhone},\\
&\quad \mathit{answerPhone} \leftarrow \mathit{secretary} \wedge \mathit{phoneRing}\}\\
\mathit{Reject}(v_3, M_{v_3}) &= \{\mathit{not\ answerPhone}\} \text{ since } v_4 \underset{v_3}{\lhd} v_1\\
O &= Q - \mathit{Reject}(v_3, M_{v_3})\\
\mathit{Default}(O, M_{v_3}) &= \{\}.
\end{aligned}
$$

Consequently, by Def. 10 it holds that M_{v_3} is a stable model of $\mathcal{F}$ at state v_3 (the inspection point of B).

6.2 Bi-directional Information Exchange between a Group and its Members

A different situation occurs when the exchange of information between the group and its members is bi-directional. That is, an agent makes part of its information available to the group when it joins it. This situation occurs, for example, in systems whose functioning requires different areas of expertise.

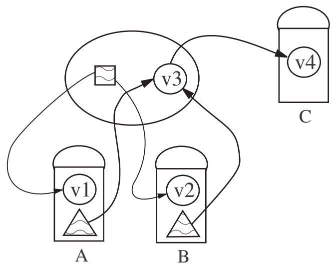

Fig. 3. Bi-directional exchange of information

Suppose that each agent in the group has its own area of expertise. When an agent joins a group, the agent receives the information made available from the group, and in turn makes its area of expertise available it. Fig. 3 depicts a group of three agents A, B and C. The group provides part of its information (i.e., the part of its WDAG depicted by the box in the oval) to both A and B. They in turn make available to the group the part of their WDAGs depicted by the corresponding triangles. Finally, a third agent C has visibility over the part of the information made available to the group by the other two agents.

6.3 Groups with Roles

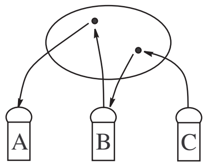

Fig. 4. Group with roles

$\mathcal{F}$ can be used to formalize groups whose members have an associated role. In this context a role can be understood as a system of prescribed behavior. We are not concerned here how the agent's mental state (represented by its MDLP) is influenced by the social context in which the agent is situated. Fig. 4 depicts a group with three agents. The role of agent C dominates the role of agent B which in turn dominates the role of agent A. Note that this formalization of the group does not guarantee trustfulness. That is, an agent may not respect the prevalence of another agent with a superior role. This occurs for instance in the following example.

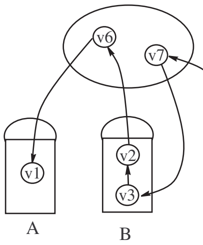

Fig. 5. Abuse of roles

Consider the group in Fig. 4 where C prevails over B which in turn prevails over A. We let $\mathcal{P}_A$, $\mathcal{P}_B$ and $\mathcal{P}_C$ be the MDLPs of the agents A, B and C, and $\mathcal{P}_G$ be the MDLP of the group.

Let $\mathcal{P}_A = (\mathcal{P}_{D_A}, D_A)$ where $D_A = (\{v_1\}, \{\}, w_A)$ and $P_{v_1} = \{\}$. Let $\mathcal{P}_B = (\mathcal{P}_{D_B}, D_B)$ where $D_B = (\{v_2, v_3\}, \{(v_3, v_2)\}, w_B)$, $w_B((v_3, v_2)) = 0.3$, $P_{v_2} = \{\}$ and $P_{v_3} = \{b, not\, a\}$. Let $\mathcal{P}_C = (\mathcal{P}_{D_C}, D_C)$ where $D_C = (\{v_4, v_5\}, \{(v_5, v_4)\}, w_C)$, $w_C((v_5, v_4) = 0.3$, $P_{v_4} = \{\}$ and $P_{v_5} = \{a \leftarrow b\}$. Let $\mathcal{P}_G = (\mathcal{P}_{D_G}, D_G)$ where $D_G = (\{v_6, v_7\}, \{\}, w_G)$ and $P_{v_6} = P_{v_7} = \{\}$.

Let $\mathcal{F} = (\mathcal{A}, L, w_L)$ where $\mathcal{A} = \{\mathcal{P}_A, \mathcal{P}_B, \mathcal{P}_C, \mathcal{P}_G\}$, $L = \{(v_4, v_7), (v_7, v_3), (v_2, v_6), (v_6, v_1)\}$ and $w_L((v_i, v_j)) = 0.5$, for every $(v_i, v_j) \in \mathcal{L}$.

The intended model of B at state v_2 is $M = \{a, b\}$ because the knowledge of C is intended to override the knowledge of B. Agent B instead overrides the knowledge of C (due to the edge from v_7 to v_3). Thus, the unique stable model of A (i.e., the stable model of $\mathcal{F}$ at state v_1) is $M_{v_1} = \{b, not\, a\}$.

Note that this problem could be easily solved by removing the edge (v_7, v_3) from $\mathcal{F}$ and by adding the edge (v_7, v_2) with sufficiently high weight to overcome the weight of (v_3, v_2). However, not always the group can guarantee that each member will respect the other agent's roles. This may happen when agents are able to self-modify their WDAGs.

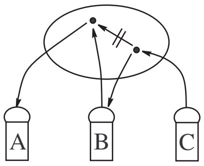

Fig. 6. Enforcing prevalence

To solve this problem we need to enforce a prevalence within the WDAG of the formalization of the group. Specifically, we need to add an edge (v_7, v_6) with high weight (i.e., the edge with double marks in Fig. 6) with the aim of making the new edge prevail over the edge (v_2, v_6) coming out from B. In this way the path $v_4 \prec v_6$ via v_7 will be a dominant path and consequently $v_3 \underset{v_6}{\lhd} v_5$. However, this solution, by enforcing the prevalence of vertices at the group level, would give less autonomy to the group's members. In general, nevertheless, it may increase the stability of the system. The study of the various options, and their connection to other problems, is earmarked for future work.

6.4 Groups with Equal Roles

Our approach caters for the possibility of capturing the equivalent interdependency of agents in a group (that is, when two or more agents reciprocally influence one another). This implies that contradicting rules in the two agents simply reject one another. (An alternative semantics could instead introduce the disjoint availability of each rule in distinct stable models.)

Consider a group with two agents A and B having equal roles. Let $\mathcal{P}_A$ and $\mathcal{P}_B$ be the MDLPs of the agents A and B, and $\mathcal{P}_G$ be the MDLP of the group.

Let $\mathcal{P}_A = (\mathcal{P}_{D_A}, D_A)$ where $D_A = (\{v_4, v_5\}, \{(v_5, v_4)\}, w_A)$, $w_A((v_5, v_4)) = 0.1$ and $P_{v_5} = \{a, c\}$. Let $\mathcal{P}_B = (\mathcal{P}_{D_B}, D_B)$ where $D_B = (\{v_6, v_7\}, \{(v_7, v_6)\}, w_B)$, $w_B((v_7, v_6)) = 0.1$ and $P_{v_7} = \{b, \mathit{not}\, a \leftarrow b \wedge c\}$. Let $\mathcal{P}_G = (\mathcal{P}_{D_G}, D_G)$ where $D_G = (\{v_1, v_2, v_3\}, \{(v_2, v_1), (v_3, v_1)\}, w_G)$ and $w_G((v_2, v_1)) = w_G((v_3, v_1)) = 0.1$.

Let $\mathcal{F} = (\mathcal{A}, L, w_L)$ where $\mathcal{A} = \{\mathcal{P}_A, \mathcal{P}_B, \mathcal{P}_G\}$, $L = \{(v_4, v_2), (v_4, v_3), (v_6, v_2), (v_6, v_3)\}$ and $w_L((v_4, v_2)) = w_L((v_6, v_3)) = 0.4$ and $w_L((v_4, v_3)) = w_L((v_6, v_2)) = 0.1$.

Then, the unique stable model of $\mathcal{F}$ at state v_1 is $M = \{b, c\}$. In fact, the inspection points v_4 and v_6 of A and B have both two dominant paths to v_1, and consequently it holds that $v_5 \underset{v_1}{\lhd} v_7$ and $v_7 \underset{v_1}{\lhd} v_5$. This means that the rules in v_5 and v_7 reject one another.

6.5 Institutions

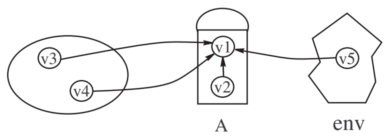

Fig. 7. Institution

An institution is a system of agents having a legal standing. The following example illustrates the use of an institution to enforce norms and regulations over the behavior of an agent A. To this aim, we formalize the environment where A is situated via a MDLP named *env*, and extend the language $\mathcal{L}$ to contain integrity constraints and active rules. (For a more detailed presentation of such a language see [8].)

Informally, an *integrity constraint* is a rule of the form $\mathit{false} \leftarrow L_1 \wedge \ldots \wedge L_n$ where every L_i $(1 \leq i \leq n)$ is a literal. Integrity constraints are rules that enforce some condition over the state, and therefore always take the form of denials, without loss of generality, in a 2-valued semantics.

An *active rule* is a rule of the form $L_1 \wedge \ldots \wedge L_n \Rightarrow Z$, where every L_i $(1 \leq i \leq n)$ is a literal and Z is a project. Active rules are rules that can modify the current state, to produce a new state updated with the project, when triggered. If the body $L_1 \wedge \ldots \wedge L_n$ of the active rule is satisfied, then the project (fluent) Z can be selected and executed. Z can take the form $\mathit{do}(\mathit{action})$ to indicate the project of the agent to execute *action* on the environment.

Consider a scenario where the agent A formalizes the theory of a driver, and the institution I formalizes norms and regulations about driving in a certain country. We let $\mathcal{P}_A$, $\mathcal{P}_I$ and $\mathcal{P}_{env}$ be the MDLPs of agent A, the institution I and the environment.

Let $\mathcal{P}_A = (\mathcal{P}_{D_A}, D_A)$ where $D_A = (\{v_1, v_2\}, \{(v_2, v_1)\}, w_A)$, $w_A((v_2, v_1)) = 0.7$, $P_{v_1} = \{\}$ and

$$P_{v_2} = \left\{ \begin{array}{r} \mathit{false} \leftarrow \mathit{drive} \wedge \mathit{do}(\mathit{usePhone}) \\ \mathit{not}\ \mathit{lowSpeed} \leftarrow \mathit{late} \\ \mathit{currentSpeed}(100) \leftarrow \\ \mathit{drive} \leftarrow \\ \\ \mathit{currentSpeed}(S) \wedge \mathit{speedLimit}(L) \wedge S > L \Rightarrow \mathit{do}(\mathit{decreaseSpeed}) \\ \mathit{lowSpeed} \wedge \mathit{currentSpeed}(S) \wedge S > 90 \Rightarrow \mathit{do}(\mathit{decreaseSpeed}) \\ \mathit{phoneRing} \Rightarrow \mathit{do}(\mathit{usePhone}) \end{array} \right\}$$

The first rule in P_{v_2}, for example, is an integrity constraint stating that it is not possible to drive and to use the phone contemporarily (this holds for example in Portugal). The first active rule says that if the current speed S is greater than the speed limit L, then the speed must be decreased (i.e., A executes the project to decrease its speed). Let v_1 be the inspection point of A. Note that if we determine the (only) stable model of A in isolation from the context $\mathcal{F}$ (where A is situated), then the integrity constraint prevents A to use the phone when driving.

Let $\mathcal{P}_I = (\mathcal{P}_{D_I}, D_I)$ where $D_I = (\{v_3, v_4\}, \{\}, w_I)$ and

$$P_{v_3} = \left\{ \begin{array}{r} \mathit{not}\ \mathit{false} \leftarrow \mathit{drive} \wedge \mathit{do}(\mathit{usePhone}) \\ \mathit{speedLimit}(110) \leftarrow \end{array} \right\}$$

$$P_{v_4} = \{\, \mathit{lowSpeed} \leftarrow \mathit{traffic} \,\}$$

This institution formalizes the norms of a certain country that must be obeyed when A is driving in that country. It is natural that A has its own rules for driving, but when it enters another country it must respect the regulations of that country. P_{v_3} contains an integrity constraint (with a default atom in its head) stating that it is not forbidden to drive and to use the phone. The institution contains also the speed limit of the country (these rules hold for example in Sweden). The institution may also contain norms of behavior that are not strict, rather they are advised for safety reasons, like to lower the speed in case of traffic. The idea is that we would like to enforce the norms that are strict within the institution, while letting the driver free to decide about the merely suggested rules.

Let $\mathcal{P}_{env} = (\mathcal{P}_{D_{env}}, D_{env})$ where $D_{env} = (\{v_5\}, \{\}, w_{env})$ and

$$P_{v_5} = \left\{ \begin{array}{r} \mathit{traffic} \leftarrow \\ \mathit{late} \leftarrow \\ \mathit{phoneRing} \leftarrow \end{array} \right\}$$

Let $\mathcal{F} = (\mathcal{A}, L, w_L)$ where $\mathcal{A} = \{\mathcal{P}_A, \mathcal{P}_I, \mathcal{P}_{env}\}$, $L = \{(v_3, v_1), (v_4, v_1), (v_5, v_1)\}$ and $w_L((v_3, v_1)) = 0.9$, $w_L((v_4, v_1)) = 0.5$ and $w_L((v_5, v_1)) = 1$.

To enforce the norms of the institution, $\mathcal{F}$ associates a high weight to the edge (v_3, v_1) and a low weight to (v_4, v_1) since v_4 contains the suggested rules. $\mathcal{F}$ associates the maximum weight to the environment. The stable model of A is $M_{v_1} =$

$\{traffic, late, phoneRing, speedLimit(110), drive, currentSpeed(100), do(usePhone)\}$. Note that M_{v_1} does not contain $do(decreaseSpeed)$.

To provide a legal standing to the institution, we may add a new vertex v_6 to the environment with the aim to maintain the history of all the projects executed by A, through updates to it. Then we can add a new agent B to the institution whose job is to control the environment and the projects executed. As soon as A executes a non-legal project, B can detect it and in response take a suitable action.

6.6 Agent Societies

More complex organizational structures can be formalized in $\mathcal{F}$, e.g., group of groups. In principle, there is no restriction on the membership of agents in groups: an agent can belong to several groups provided that the joined MDLP induced by $\mathcal{F}$ does not contain any cycle. We can then empower agents with communication abilities so that they can have multiple means of interaction amongst each other, and make the knowledge of the agents updatable and self-evolvable to incorporate new incoming information along the lines outlined in [8]. All these ingredients assembled into a single eMAS architecture will allow us to engineer configurable agent societies where the agents can self-organize themselves with respect to their goals, and self-evolve. In this way, the overall "emerging" structure will be flexible and dynamic.

6.7 Agent Societies Based on Voting

The presented 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 $Reject(s, M)$ in Def. 7, and at the syntactical level, that is to modify the *rejection rules*.

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, we can provide some degree of freedom to the agents as a whole. 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. 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.

7 Concluding Remarks

We have shown a logical framework that allows us to model structures of epistemic agents. In Section 3 we have introduced the notion of WDAG that extends DAGs to associate weights to every edge of the graph, and in Section 4 we have presented the logical framework $\mathcal{F}$ with the corresponding declarative semantics. Its procedural semantics has been sketched in Section 5. Having the framework $\mathcal{F}$ a formal semantics will allow us to study and prove the properties of eMAS structures. For simplicity, this paper focuses 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 [8].

Another interesting direction for future work is to represent the logical framework $\mathcal{F}$ within the theory of the agent members of the group. 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 links of $\mathcal{F}$ within the language of the agents makes $\mathcal{F}$ updatable to capture the dynamic aspects of the system, i.e., of the open society.

A further interesting extension is to investigate how to represent agent societies based on voting.

Acknowledgements

L. M. Pereira acknowledges the support of POCTI project 40958 "FLUX - FleXible Logical Updates" and P. Dell'Acqua a Research Grant by the *Stiftelsen Lars Hiertas Minne*.

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, Proc. 8th European Conf. (JELIA02)*, LNAI 2424, pages 62–73. Springer, 2002.
2. J. J. Alferes, J. A. Leite, L. M. Pereira, H. Przymusinska, and T. C. Przymusinski. 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. C. Castelfranchi. Engineering Social Order. In Andrea Omicini, Robert Tolksdorf, and Franco Zambonelli, editors, *Engineering Societies in the Agents World. 1st Int. Workshop ESAW 2000. Revised Papers*, LNAI 1972, pages 1–18, Berlin, 2000. Springer-Verlag.
5. P. Davidsson. Categories of artificial societies. In A. Omicini, P. Petta, and R. Tolksdorf, editors, *Engineering Societies in the Agents' World (ESAW 2001)*, LNAI 2203, pages 1–9. Springer-Verlag, 2001.
6. 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.
7. 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.
8. 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.
9. P. Dell'Acqua and L. M. Pereira. Preferring and updating in logic-based agents. In: Web-Knowledge Management and Decision Support. Selected Papers from the 14th Int. Conf. on Applications of Prolog (INAP), Tokyo, 2001. To appear in LNAI, 2001.
10. 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.
11. N. R. Jennings. On agent-based software engineering. *Artificial Intelligence*, 117:277–296, 2000.
12. 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.
13. J. A. Leite, J. J. Alferes, and L. M. Pereira. Multi-dimensional dynamic knowledge representation. In T. Eiter, W. Faber, and M. Truszczynski, editors, *Procs. of the Sixth Int. Conf. on Logic Programming and Nonmonotonic Reasoning (LPNMR'01)*, LNAI 2173, pages 365–378. Springer, 2001.
14. 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.
15. F. Zambonelli. Abstractions and infrastructures for the design and development of mobile agent organizations. In M. J. Wooldridge, G. Weiβ, and P. Ciancarini, editors, *Agent-Oriented Software Engineering II, Second International Workshop, AOSE 2001*, LNCS 2222, pages 245–262, Berlin, 2001. Springer-Verlag.
16. 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.