On Modelling Multi-agent Systems Declaratively

Andrea Bracciali[1], Paolo Mancarella[1], Kostas Stathis[1,2], and Francesca Toni[1,3]

[1] Dipartimento di Informatica, Università di Pisa
{braccia, paolo}@di.unipi.it
[2] Department of Computing, City University London
kostas@soi.city.ac.uk
[3] Department of Computing, Imperial College London
ft@doc.ic.ac.uk

Abstract. We propose a declarative framework for modelling multi-agent systems and specify a number of properties of these systems and agents within them. The framework is parametric with respect to an input/output semantics for agents, whereby inputs are the agents' observations, and outputs are their actions. The observations include actions performed by other agents and events happening in the world. We define the semantics of a multi-agent system via a stability condition over the individual agents' semantics. We instantiate the framework with respect to simple abductive logic agents. We illustrate the framework and the proposed properties by means of a simple example of agent negotiation.

1 Introduction

The ever-growing use of agents and multi-agent systems in practical applications poses the problem of formally verifying their properties; the idea being that by verifying properties of the overall system we can make informed judgements about the suitability of agents and multi-agent systems in solving problems posed within application domains. For example, if a multi-agent system is to be used to negotiate on behalf of people, in order to solve problems of re-allocation and sharing of resources (e.g., as in [1]), the problem arises as to whether a specific set of agents/multi-agent system can actually solve a concrete problem of resource-reallocation.

We specify a set of generic properties, which we believe to be interesting, of individual agents, multi-agent systems and agents within multi-agent systems. Rather than proposing a specific architecture or theory for agents, we view agents as "black-boxes", whose "semantics" is expressed solely in terms of (i) their *observable behaviour*, which is public and thus visible to other agents in the same multi-agent system, and (ii) their *mental state*, which is private and thus inaccessible to other agents in the same multi-agent system. Our proposed properties can be instantiated for any concrete agent architecture/theory that can be abstracted away in terms of the aforementioned "semantics", and apply to systems consisting of architecturally heterogeneous agents, including legacy systems. Thus, our approach is not concerned with the specification or programming of agents and agents' applications, but rather it is tailored towards the specification of properties of agents, which is to serve for their verification.

J. Leite et al. (Eds.): DALT 2004, LNAI 3476, pp. 53–68, 2005.

The observable behaviour of an agent is expressed in terms of an output set of actions from a pool of actions that the agent can perform, given an input set of observations from a pool of observations that the agent can make. Actions and observation can be communicative or not. Actions of one agent may be observations of another. Observations may include also events in the world in which agents are situated. The set of visible events and actions by other agents that an agent can observe in the world constitute its environment. If all agents in a multi-agent system can observe all events happening in the world and all actions performed by the other agents, then we call the multi-agent system *fully transparent*. Otherwise, we call the system *partially transparent*. The mental state is seen as a set of beliefs by the agent. Actions, observations, events and beliefs are seen as atoms in some logical languages.

Given the "semantics" of agents as described above, we define the semantics of a multi-agent system via a definition of *stability* on the set of all actions performed by all agents in the system, possibly arising from their communication and interaction via observation: a set of actions (by the different agents) is stable if, assuming that an "oracle" could feed each of the agents with all the actions in the set performed by the other agents (and all events happening in the world), then each agent would do exactly what is in the set, namely their observable behaviour would be exactly what the set envisages.

We specify properties of individual success of agents, overall success of a multi-agent system, robustness and world-dependence of a multi-agent system, as well as a number of properties of agents within systems. We then instantiate our framework by means of simple abductive logic agents, whose mental state and observable behaviour can be computed by applying an adaptation of the T_p operator of logic programs (see e.g., [2]) starting from the observations of the agents. If a multi-agent system consists of these simple agents, we show how stable sets of actions by all the agents can be computed incrementally. We also illustrate the framework and the properties we propose in the context of multi-agent systems consisting of the simple abductive logic agents.

2 Preliminaries

A *multi-agent system* $\langle \mathcal{A}, \mathcal{W} \rangle$ consists of a set $\mathcal{A}$ of n *agents* ($n \geq 2$) that we refer to simply as $1, \ldots, n$, and a *world* $\mathcal{W}$ in which events may happen which the agents may perceive. Until section 5, we will abstract away from the details of the agents' architecture and model, and simply rely upon the existence of a *semantics* of agents, as understood below. Thus, note that our model applies to systems of architecturally heterogeneous agents. We will also abstract away from the details of the world, except for assuming that it is characterised by a (possibly empty, possibly infinite) set of *events*, which may be observed by the agents. We will refer to these events as $E(\mathcal{W})$.

Each agent i is associated with a (possibly empty, possibly infinite) set of potential *actions* that it can perform, indicated as $A(i)$, and a (possibly empty, possibly infinite) set of *observations* it can make, indicated as $O(i)$. Without loss of generality, we will assume that $A(i) \cap A(j) = \emptyset$, for $i \neq j$, namely no action can be performed by two different agents. For example, the action whereby agent 1 asks agent 2 for some resource can only be performed by agent 1, while the action whereby agent 2 asks agent

3 for some resource can only be performed by agent 2, and so on. For simplicity, we do not explicitly deal with the representation of time, but we assume that actions are distinguished by their execution time (i.e. the same action executed at different instants will be represented by different elements in $A(i)$) and executed in the "proper" order. Also, given some set Δ, we will denote by $\Delta(j)$ the set of actions in Δ pertaining to the agent j, namely $\Delta(j) = \Delta \cap A(j)$.

Actions performed by one agent may be observations of another, namely the language in which actions and observations are represented is common amongst the agents. E.g., actions may be outgoing communication and observations may be incoming communication, and the language in which they are represented may be an agent communication language. Observations by agents may also be events happening in the world, taken from $E(\mathcal{W})$. Formally,

$$\bigcup_{i \in \mathcal{A}} O(i) \subseteq E(\mathcal{W}) \cup \bigcup_{i \in \mathcal{A}} A(i)$$

In Section 3.1 we will first consider the case in which each agent can observe all other agents' actions as well as the whole world. In Section 3.2 we will consider the case in which each agent may have only a partial visibility both of other agents' actions and of the world. This may be due to its inability to fully observe the other agents and the world, as well as to the unwillingness of some agents to disclose all their actions to every other agent. The portion of the world and of the (actions performed by) other agents visible to an agent can be seen as the *environment* in which this agent is situated.

The semantics of agent i is indicated as

$$\mathcal{S}^i(\Delta_{in}, \Delta_0) = \langle M, \Delta_{out} \rangle,$$

where

- $\Delta_{in} \subseteq O(i)$ is a (possibly infinite) set of observations by agent i,
- $\Delta_0 \subseteq A(i)$ is a (possibly infinite) set of actions by agent i,
- M is a (possibly infinite) set of atomic sentences (from a given "private" language that the agent is equipped with), understood as the *mental state* of the agent, and
- $\Delta_{out} \subseteq A(i)$ is a (possibly infinite) set of actions performed by agent i, understood as the *observable behaviour* of the agent.

Δ_0 will typically belong to some initial plan of the agent i, allowing i to achieve its *goals* or *desires*, according to its mental state. We will refer to the goals of agent i as G_i. Syntactically, goals are sets of atoms in the internal language of the agent. In particular, the set of goals may be empty. M can be seen as the set of atomic beliefs held by the agent, and *private* to the agent itself. It may be $\bot$, indicating the inconsistency of a mental state of the agent. Δ_{out} is instead the *public* side of the agent. Given Δ_{in} and Δ_0, $\mathcal{S}^i(\Delta_{in}, \Delta_0)$ may not be unique (namely $\mathcal{S}^i$ may not be a function in general).

Although this declarative formulation of our model can deal with infinite sets, e.g., accounting for reactive agent behaviour, its operational counterparts for verification will typically revert to finite instances of agents' behaviour (as in well-known verification methodologies, like finite model checking). Section 5 proposes a possible way to construct a concrete such semantics for agents based on abductive logic programming.

3 Semantics of a Multi-agent System

We define a semantics for a multi-agent system, parametric with respect to the semantics of the individual agents. This semantics relies upon the notion of stable set of actions (by all agents in the system). Agents are assumed to start with (possibly empty) initial plans $\Delta_0^1, \ldots, \Delta_0^n$. Moreover, the world is supposed to provide a set $\Delta_E \subseteq E(\mathcal{W})$ of happened events. We provide two definitions for the notion of stable set, according to whether the agents fully or partially perceive the world and the other agents.

3.1 Fully Transparent Multi-agent Systems

In this section we assume that each agent has full perception of each other agent as well as of the world. We call such a multi-agent system *fully transparent*.

Definition 1. *A fully transparent multi-agent system* $\langle \mathcal{A}, \mathcal{W} \rangle$ *is* stable *if there exists* $\Delta \subseteq \bigcup_{i \in \mathcal{A}} A(i)$*, such that*

i. $\bigcup_{i \in \mathcal{A}} \Delta_{out}^i = \Delta$
ii. $\mathcal{S}^i(\Delta^{-i} \cup \Delta_E, \Delta_0^i) = \langle M^i, \Delta_{out}^i \rangle$
iii. $\Delta \supseteq \bigcup_{i \in \mathcal{A}} \Delta_0^i$

where Δ^{-i} *is the set of all actions performed by all agents except agent* i*, namely*

$$\Delta^{-i} = \bigcup_{\substack{j \in \mathcal{A} \\ j \neq i}} \Delta(j)$$

The set Δ *is called a* stable *set for* $\langle \mathcal{A}, \mathcal{W} \rangle$*.*

By the previous definition, the sets $\Delta_{out}^1, \ldots, \Delta_{out}^n$, if they exist, are a solution for the set of mutually recursive equations

$$\begin{aligned} \mathcal{S}^1(\Delta^{-1} \cup \Delta_E, \Delta_0^1) &= \langle M^1, \Delta_{out}^1 \rangle \\ &\vdots \\ \mathcal{S}^n(\Delta^{-n} \cup \Delta_E, \Delta_0^n) &= \langle M^n, \Delta_{out}^n \rangle \end{aligned}$$

where each Δ^{-i} occurring on the left-hand side of the $i-th$ equation is defined in terms of the Δ_{out}^j sets, occurring in all the other equations. Intuitively speaking, a set of actions (by the different agents) is stable if, assuming that an "oracle" could feed each of the agents with all the actions in the set performed by the other agents (and all events happening in the world), then each agent would do exactly what is in the set, namely their observable behaviour would be exactly what the set envisages. Note that the assumption on the existence of an "oracle" is justified by the fact that we are providing a semantics for multi-agent systems, rather than relying upon their execution model.

Note that conditions *i.* and *ii.* in Definition 1 imply that $\Delta_0^i \subseteq \Delta_{out}^i$, namely that agents cannot change their initial plans. This condition could be relaxed.

3.2 Partially Transparent Multi-agent Systems

We model now multi-agent systems where each agent may have only a partial visibility of the rest of the system and of the world. We call such multi-agent systems *partially transparent*. We assume that the perception of the world by every agent i is given by $\Delta_E^i \subseteq \Delta_E$, as opposed to the whole Δ_E in Definition 1(*ii.)*. Δ_E^i could be defined via a suitable projection function. Clearly, for fully transparent multi-agent systems $\Delta_E^i = \Delta_E$.

Definition 2. *A partially transparent multi-agent system* $\langle \mathcal{A}, \mathcal{W} \rangle$ *is* stable *if there exists* $\Delta \subseteq \bigcup_{i \in \mathcal{A}} A(i)$ *such that*

i. $\bigcup\limits_{i \in \mathcal{A}} \Delta_{out}^i = \Delta$
ii. $\mathcal{S}^i(\Delta^{-i} \cup \Delta_E^i, \Delta_0^i) = \langle M^i, \Delta_{out}^i \rangle$
iii. $\Delta \supseteq \bigcup\limits_{i \in \mathcal{A}} \Delta_0^i$

where

$$\Delta^{-i} \subseteq \bigcup_{\substack{j \in \mathcal{A} \\ j \neq i}} \Delta(j)$$

The set Δ *is called a* stable *set for* $\langle \mathcal{A}, \mathcal{W} \rangle$.

Moreover, the set Δ^{-i} does not consists, in the general case, of the whole set of actions performed by other agents. Concretely, for each agent i and set $\Delta \subseteq \bigcup\limits_{i \in \mathcal{A}} A(i)$, the set Δ^{-i} can be given by a suitable *visibility projection function* which filters out the elements of Δ that are not visible to agent i. For example

$$\Delta^{-i} = \bigcup_{\substack{j \in \mathcal{A} \\ j \neq i}} v_i^j(\Delta(j))$$

where v_i^j is the visibility projection function of agent i on agent j, expressing what agent i sees of what agent j does. Necessarily, $v_i^j(X) \subseteq X$, and, for fully transparent multi-agent systems, $v_i^j(X) = X$. Actions performed by j and not "seen" by i may be private to j, or simply not under i's jurisdiction. Note that the visible environment of i, given Δ_E and Δ, can be formally defined as

$$\mathcal{E}(i) = \Delta_E^i \cup \bigcup_{\substack{j \in \mathcal{A} \\ j \neq i}} v_i^j(\Delta(j))$$

4 Properties

In this section we define properties of individual agents, of multi-agent systems, and of agents in multi-agent systems. These properties rely upon agents having the semantics we describe in section 2 and multi-agent systems having the semantics we describe in sections 3.1 and 3.2, depending on whether they are fully or partially transparent.

4.1 Individual Agents

Definition 3. *(Successful agent)*
Assume that agent i is equipped with a set of desires G_i. We say that the agent is successful *with respect to input Δ_{in} and initial plan Δ_0 (for G_i) if $\mathcal{S}^i(\Delta_{in}, \Delta_0) = \langle M, \Delta_{out} \rangle$ and $G_i \subseteq M$.*

Namely, a successful agent is one that achieves its desires, in that its desires hold in the mental state of the agent. Note that our notion of success is local and subjective to the agent, namely, an agent may believe to be successful without being so in the world. Note also that, if the agent has no desires, then success amounts to its mental state being different from $\bot$. This is required also in the case of the agent being equipped with desires.

4.2 Multi-agent Systems

Definition 4. *(Overall successful system)*
$\langle \mathcal{A}, \mathcal{W} \rangle$ is overall successful wrt some Δ_E, $\Delta_0^1, \ldots, \Delta_0^n$, if there exists a stable Δ such that each i is successful, wrt Δ^{-i} and Δ_0^i.

Namely, overall success amounts to individual success for all the agents. Note that this is a rather weak notion of overall success, as it only requires for *one* successful stable set to exist. Stronger versions could also be interesting. Note also that, if agents have no desires, then overall success amounts to the existence of a stable set and to the property that no agent has $\bot$ as its mental state.

Definition 5. *(Robust system)*
An overall successful system $\langle \mathcal{A}, \mathcal{W} \rangle$ is robust if there exists no $i \in \mathcal{A}$ such that $\langle \mathcal{A} \setminus \{i\}, \mathcal{W} \rangle$ is not.

Namely, a robust system is one that does not need any of its agents to be overall successful, or, alternatively, one in which no agent needs any of the others in order to be successful.

Definition 6. *(World-dependent system)*
$\langle \mathcal{A}, \mathcal{W} \rangle$ is world-dependent if it is not overall successful wrt $\Delta_E = \emptyset$ (and any $\Delta_0^1, \ldots, \Delta_0^n$) but it is overall successful wrt some $\Delta_E \neq \emptyset$ (and some $\Delta_0^1, \ldots, \Delta_0^n$).

Namely, a world-dependent multi-agent system is one that cannot do without the world, and events happening in it, to be successful.

4.3 Agents in Multi-agent Systems

Definition 7. *(Aware agent)*
Let $\langle \mathcal{A}, \mathcal{W} \rangle$ be a (fully or partially) transparent multi-agent system, and $i \in \mathcal{A}$. Given input Δ_{in}, initial plan Δ_0, and set of events Δ_E, let $\mathcal{S}^i(\Delta_{in}, \Delta_0) = \langle M^i, \Delta^i_{out} \rangle$. Then, we say that agent $i \in \mathcal{A}$ is

- world aware*, if $\Delta^i_E \cap \Delta_{in} \subseteq M^i$,*
- j-aware*, for some $j \in \mathcal{A}$, $j \neq i$, if $A(j) \cap \Delta_{in} \subseteq M^i$,*
- environment aware*, if it is world-aware and j-aware, for all $j \in \mathcal{A}$, $j \neq i$.*

Namely, a world-aware agent is one that holds, within its mental state, a belief of all the events that have happened in the world and that it has observed. An other-agent aware agent is one that believes in all the observations it made upon the other. An environment-aware agent is one that believes in everything it observes, including events in the world and actions by other agents it can observe.

Definition 8. *(System dependent agent)*
Let $\langle \mathcal{A}, \mathcal{W} \rangle$ be a (fully or partially) transparent multi-agent system, and $i \in \mathcal{A}$. Given Δ_E and G^i, assume that for no initial plan Δ_0, agent i is successful with respect to Δ_E and Δ_0. We say that agent i is system dependent *if there exists a stable set Δ for $\langle \mathcal{A}, \mathcal{W} \rangle$ such that agent i is successful with respect to Δ^{-i} and some initial plan Δ_0.*

Namely, a system-dependent agent is one that cannot be successful alone, but it can be successful if with other agents in a multi-agent system. Thus, this agent has a motivation to look for other agents with which to join forces.

Definition 9. *(Dispensable agent)*
Let $\langle \mathcal{A}, \mathcal{W} \rangle$ be a (fully or partially) transparent multi-agent system, and $i \in \mathcal{A}$. Agent i is dispensable within $\langle \mathcal{A}, \mathcal{W} \rangle$ if $\langle \mathcal{A} \setminus \{i\}, \mathcal{W} \rangle$ is overall successful.

Namely, a dispensable agent is one that is not needed to guarantee success of the other agents in the system. So, designers of a multi-agent systems, or individual agents having control over which agents belong to the system, could exclude any dispensable agent from it (e.g., to reduce communication costs).

Definition 10. *(Dangerous agent)*
Let $\langle \mathcal{A}, \mathcal{W} \rangle$ be a (fully or partially) transparent multi-agent system. $i \notin \mathcal{A}$ is dangerous to $\langle \mathcal{A}, \mathcal{W} \rangle$ if $\langle \mathcal{A}, \mathcal{W} \rangle$ is overall successful but $\langle \mathcal{A} \cup \{i\}, \mathcal{W} \rangle$ is not.

Namely, a dangerous agent is one that can undermine the overall success of a multi-agent system, if added to it. So, designers of a multi-agent systems, or individual agents having control over which agents belong to the system, should make sure that no dangerous agent belong to the system.

5 A Concrete Multi-agent Semantics

We illustrate our framework by means of a simple example where agents are *abductive logic agents*. Abductive logic programming has been recently used to describe agents

and their interactions (see e.g., [3, 4, 5]). The semantics $\mathcal{S}$ of a single (abductive) agent is defined by means of a bottom-up construction, in the spirit of the T_p operator for logic programs [2], and adapted here for abductive logic programs. Informally, given a *"partial semantics",* the operator returns a more defined semantics, if it exists, by adding the immediate consequences of it. The (possibly infinite) repeated application of the operator is proved to converge to a semantics which is taken as the semantics $\mathcal{S}$ of the agent. This kind of semantics is then lifted to multi-agent systems by defining a bottom-up semantics in terms of the operators of the single agents the multi-agent system is made up of. This construction of $\mathcal{S}$ is not to be interpreted as the execution model of the agent. For simplicity, we concentrate upon fully transparent multi-agent systems.

5.1 Single Agent Language and Semantics

Due to lack of space, we assume that the reader has some familiarity with abductive logic programming (ALP for short, see e.g., [6]). An agent i consists of an abductive theory $\langle P, O \cup A, IC\rangle$, where P is a logic program, $O \cup A$ is a set of *abducible atoms* partitioned in *observations* and *actions*, and IC is a set of *integrity constraints*. [1] P consists of a set of clauses of the form

$$p \leftarrow p_1, \ldots, p_n \qquad n \geq 0$$

where p is a non-abducible atom and $p_1, \ldots, p_n$ are (possibly abducible) atoms. As usual in ALP, we assume that abducibles have no definition in P. The integrity constraints IC are of the form

$$p_1, \ldots, p_n \Rightarrow false \qquad\qquad p_1, \ldots, p_n \Rightarrow a$$

where $false$ is a special symbol denoting *integrity violation*, each p_j is a (possibly abducible) atom and a is an action, namely $a \in A$. Notice that $\bot$ can occur only in the conclusion of integrity constraints. We assume that variables occurring in clauses and integrity constraints are implicitly universally quantified from the outside, with scope the entire formula in which they occur. Moreover, we assume that no variable occurs in the conclusion of an IC that does not occur in its body. As usual in logic programming, given an abductive logic agent as defined above, we will denote by $ground(P)$ (resp. $ground(IC)$) the (possibly infinite) set of all possible ground instantiations of the clauses in P (resp. of the integrity constraints in IC). Moreover, given a set of ground abducibles $\Delta \subseteq O \cup A$, we indicate with I an interpretation for $P \cup \Delta$. Roughly speaking, the semantics of an abductive theory $\langle P, O \cup A, IC\rangle$, if it exists, can be given as a pair $\langle I, \Delta\rangle$, where $\Delta \subseteq O \cup A$, I is a model of $P \cup \Delta \cup IC$ and $false \notin I$ (see e.g., [7]).

In the sequel, given an abductive logic agent, we define its input/output semantics $\mathcal{S}(\Delta_{in}, \Delta_0)$ by a suitable $\mathcal{T}$ operator, which step-wise approximates both the mental state and the observable behaviour of the agent, and which is a simple generalization of the immediate consequences operator T_P of logic programming, suitably extended in order to take integrity constraints into account.

[1] The sets O and A correspond to the sets $O(i)$ and $A(i)$ of Section 2, respectively.

Definition 11 ($\mathcal{T}$ operator). *Given an abductive logic agent $\langle P, O \cup A, IC\rangle$, let $\Delta \subseteq O \cup A$ and let I be an interpretation. The $\mathcal{T}$ operator is defined as:*

$$\mathcal{T}(I, \Delta) = \langle I', \Delta'\rangle$$

where:
$I' = \{p \mid p \leftarrow l_1, \ldots\ l_n \in ground(P) \ \wedge\ \{l_1, \ldots\ l_n\} \subseteq I \cup \Delta\}$,
$\Delta' = \Delta \cup \{a \in A \cup \{false\} \mid l_1, \ldots\ l_n \Rightarrow a \in ground(IC) \wedge \{l_1, \ldots\ l_n\} \subseteq I \cup \Delta\}$.

It is not difficult to see that the $\mathcal{T}$ operator is monotonic. For simplicity, in the sequel we use $\subseteq$ to denote pairwise set inclusion.

Lemma 1 ($\mathcal{T}$ is monotonic). *Let be $\langle I_1, \Delta_1\rangle \subseteq \langle I_2, \Delta_2\rangle$, then:*

$$\mathcal{T}(I_1, \Delta_1) \subseteq \mathcal{T}(I_2, \Delta_2).$$

Proof. Let $\mathcal{T}(I_1, \Delta_1) = \langle I_1', \Delta_1'\rangle$ and $\mathcal{T}(I_2, \Delta_2) = \langle I_2', \Delta_2'\rangle$. We show that $I_1' \subseteq I_2'$ (the proof of $\Delta_1' \subseteq \Delta_2'$ is analogous). Let $p \in I_1'$. Then there exists a clause in $ground(P)$ of the form $p \leftarrow l_1, \ldots\ l_n$ such that $\{l_1, \ldots\ l_n\} \subseteq I_1 \cup \Delta_1$. Since, by hypothesis, $I_1 \cup \Delta_1 \subseteq I_2 \cup \Delta_2$, $\{l_1, \ldots\ l_n\} \subseteq I_2 \cup \Delta_2$ and hence $p \in I_2'$. □

The monotonicity of $\mathcal{T}$ ensures that, given a set of observations $\Delta_{in} \subseteq O$ and an initial plan $\Delta_0 \subseteq A$, we can define the semantics of an abductive logic agent i in terms of the least fix-point of the $\mathcal{T}$ operator, that we denote by $\mathcal{T}_\infty(\emptyset, \Delta_{in} \cup \Delta_0)$, starting from the initial pair $\langle \emptyset, \Delta_{in} \cup \Delta_0\rangle$.

Definition 12. *Given an abductive logic agent* i*, an initial set of observations Δ_{in} and an initial plan Δ_0, let $\mathcal{T}_\infty(\emptyset, \Delta_{in} \cup \Delta_0) = \langle M, \Delta\rangle$. Then*

$$\mathcal{S}^i(\Delta_{in}, \Delta_0) = \begin{cases} \langle M, \Delta(i)\rangle & \text{if } false \notin M \\ \langle \bot, \Delta(i)\rangle & \text{otherwise} \end{cases}$$

Example: A Concrete Agent. Consider a simple agent 1 who can achieve some goal g by asking to get a resource from a friend (we assume that resources can be shared amongst agents and be re-used as many times as required). This simplifying assumption allows us to present our model within a monotonic framework. Agent 1 believes that agent 2 is a friend. Agent 1 can observe that another agent gives something to it and can perform the actions of paying and thanking. It is forced to thank a friend or pay an enemy for a received resource.

P: $g \leftarrow friend(Y), ask(1, Y, r), getfrom(Y, r)$
$getfrom(Y, r) \leftarrow give(Y, 1, r)$
$friend(2)$
IC: $give(Y, 1, r), friend(Y) \Rightarrow thanks(1, Y)$
$give(Y, 1, r), enemy(Y) \Rightarrow pay(1, Y)$

O: $give(Y, 1, r)$
A: $thank(1, Y)$
$pay(1, Y)$
$ask(1, Y, r)$

We also assume here an implicit treatment of time, so that an asking action is performed before the asked resource is obtained.

Let us imagine that the agent has the initial plan to ask for the resource from agent 2, i.e., $ask(1,2,r) \in \Delta_0$, and that agent 2 is actually giving the owned resource to 1, as confirmed by the observation $give(2,1,r) \in \Delta_{in}$. The semantics of the agent is then defined as follows (note that in this case the fix-point has been reached in few iterations):

$$\mathcal{T}_1(\emptyset, \{give(2,1,r), ask(1,2,r)\}) = \\ \langle \{friend(2), getfrom(2,r)\}, \{give(2,1,r), ask(1,2,r)\}\rangle$$
$$\mathcal{T}_2(\emptyset, \{give(2,1,r), ask(1,2,r)\}) = \\ \langle \{friend(2), getfrom(2,r), g\}, \{give(2,1,r), ask(1,2,r)\}\rangle$$
$$\mathcal{T}_3(\emptyset, \{give(2,1,r), ask(1,2,r)\}) = \\ \langle \{friend(2), getfrom(2,r), g\}, \{give(2,1,r), ask(1,2,r), thank(1,2)\}\rangle$$
$$\mathcal{T}_4(\emptyset, \{give(2,1,r), ask(1,2,r)\}) = \mathcal{T}_3(\emptyset, \{give(2,1,r), ask(1,2,r)\})$$

that is, the agent satisfies its goal g. In the notation of Section 3.1:

$$\mathcal{S}^1(\{give(2,1,r)\}, \{ask(1,2,r)\}) = \\ \langle \{friend(2), getfrom(2,r), g\}, \{ask(1,2,r), thank(1,2)\}\rangle.$$

Instead, considering the case in which agent 1 asks another agent not believed to be a friend, say agent 3 that behaves as agent 2, it still acquires the resource, but fails its goal g:

$$\mathcal{S}^1(\{give(2,1,r)\}, \{ask(1,3,r)\}) = \\ \langle \{friend(2), getfrom(2,r)\}, \{ask(1,3,r), thank(1,2)\}\rangle.$$

5.2 Multi-agent Semantics

A fully transparent multi-agent system, as defined in Section 3.1, can consist of agents whose concrete semantics is the one defined in Section 5.1. We first show a simple example of the resulting semantics for a multi-agent system consisting of agent 1 previously introduced, and two new agents. Then, we define an operational bottom-up semantics for the multi-agent system, by lifting the single agent semantics. Semantics hence consists of a set of mutually recursive $\mathcal{T}^j$, one for each agent participating into the system. Finally, we prove that, under specific circumstances, the operational semantics entails the one defined in Section 3.1.

Example: A Fully Transparent Multi-agent System. Let us consider a system consisting of agent 1 of Section 1, together with agents 2 and 3, as below defined:

2:
- P: $have(r) \leftarrow offer(Y,2,r)$
- A: $give(2,X,r)$
- O: $ask(X,2,r)$, $offer(Y,2,r)$
- IC: $ask(X,2,r), have(r) \Rightarrow give(2,X,r)$

3:
- P: $friend(2)$, $have(r)$
- A: $offer(3,X,r)$
- IC: $have(r), friend(X) \Rightarrow offer(3,X,r)$

Agent 2 has a resource if it observes that the resource has been offered by someone. In this case the agent is forced to give the resource to anybody who requires it. Agent

3 has the resource and a friend, and it must give the owned resource to the friend. Agent 1 is the only agent having a goal, g namely, while all the others have a reactive behaviour with respect to (their representation of) the world and the behaviour of the other agents. Given their knowledge bases, agents are able to cooperate and allow agent 1 to accomplish its goal, as soon as it adopts the initial plan to ask for the resource ($\Delta_0^1 = \{ask(1,2,r)\}$).

Assuming that no other information is provided by the environment $\Delta_E = \emptyset$, and that agents 2 and 3 have empty initial plans, $\Delta_0^2 = \Delta_0^3 = \emptyset$,

$$\Delta = \{ask(1,2,r), give(2,1,r), thank(1,2), \mathit{offer}(3,2,r)\}$$

is a *stable set* for the multi-agent system $\langle \mathcal{A} = \{1,2,3\}, \mathcal{W}\rangle$ with $E(\mathcal{W}) = \emptyset$. Indeed, we have

$$\begin{aligned}
&\mathcal{S}^1(\Delta^{-1}, \{ask(1,2,r)\}) = \langle \{g, friend(2), getfrom(2,r)\},\\
&\qquad\qquad\qquad\qquad \{\mathbf{ask(1,2,r)}, \mathbf{thank(1,2)}\}\rangle\\
&\mathcal{S}^2(\Delta^{-2}, \emptyset) = \langle \{have(r)\}, \{\mathbf{give(2,1,r)}\}\rangle\\
&\mathcal{S}^3(\Delta^{-3}, \emptyset) = \langle \{friend(2), have(r)\}, \{\mathbf{offer(3,2,r)}\}\rangle
\end{aligned}$$

and $\bigcup_{i\in\mathcal{A}} \Delta_{out}^i = \Delta \supseteq \bigcup_{i\in\mathcal{A}} \Delta_0^i$, where Δ_{out}^i are boldface. Notice how some of the *actions* performed by an agent are interpreted as *observations* by the other agents (e.g., $ask(1,2,r)$ for agents 1 and 2, respectively).

The multi-agent system is thus *overall successful*, but it is not *robust* (e.g., 2 is needed for the overall success of the system, and so is 3). Agent 1 is *system-dependent*, whereas agents 2, 3 are not. Finally, $\langle \mathcal{A} = \{1,2,3\}, \mathcal{W}\rangle$ is obviously not *world-dependent*.

5.3 Fully Transparent Multi-agent System Operational Semantics

Similarly to the case of the single agent operational semantics presented in Section 5.1, also multi-agent system can be provided with a bottom-up semantics in the case of the simple agent language taken into account. The semantics of a system builds upon the semantic operators $\mathcal{T}^i$ of the single agents i belonging to the system. The overall semantics is then obtained by the mutual interaction of agent semantics, where each application of the semantic operators takes into account not only the single agent so-far approximated, but also the observable semantics, namely the actions, produced up to now by the repeated application of the semantic operators of the other agents. In this way, agents "react" to the output actions by the other agents in the system as soon as they are observed.

The operational counterpart of $S^j(\Delta_{in}^j, \Delta_0^j)$ within the context of the chosen language, is defined on top of the single agent operational semantics as a class of mutually recursive operators, which step-wise approximate the semantics of the system. In the following we will use the short-hand $\langle \overline{I}, \overline{\Delta}\rangle$ for the tuple $\langle \langle I^1, \Delta^1\rangle, \ldots, \langle I^n, \Delta^n\rangle\rangle$, where $1, \ldots, n$ are the agents in $\mathcal{A}$. On the other hand, when clear from the context, $\langle I^i, \Delta^i\rangle$ will denote the $i-$th component of the tuple $\langle \overline{I}, \overline{\Delta}\rangle$. Finally, given two tuples $\langle \overline{I}, \overline{\Delta}\rangle$ and $\langle \overline{J}, \overline{\Gamma}\rangle$, we will write $\langle \overline{I}, \overline{\Delta}\rangle \subseteq \langle \overline{J}, \overline{\Gamma}\rangle$ as a shorthand for the conjunction $\langle I^1, \Delta^1\rangle \subseteq \langle J^1, \Gamma^1\rangle \wedge \ldots \langle I^n, \Delta^n\rangle \subseteq \langle J^n, \Gamma^n\rangle$.

For simplicity, in this section we consider multi-agent systems where the world component $\mathcal{W}$ is not present. Hence, in the sequel we refer to a multi-agent system consisting only of a set $\mathcal{A}$, where each agent i is an abductive logic agent $\langle P_i, O_i \cup A_i, IC_i\rangle$ (as introduced in Section 5.1). For each agent $i \in \mathcal{A}$ we denote by $\mathcal{T}^i$ its operator as defined in Definition 11.

Definition 13 ($\mathcal{T}^{\mathcal{A}}$)**.** *Let $\mathcal{A} = \{1, \ldots, n\}$, I^i and Δ^i be an interpretation and a subset of abducibles for each agent i, respectively. The $\mathcal{T}_{\mathcal{A}}$ operator is defined as follows*

$$\mathcal{T}_{\mathcal{A}}(\overline{I}, \overline{\Delta}) = \langle \overline{J}, \overline{\Gamma}\rangle$$

where for each i,

$$\langle J^i, \Gamma^i\rangle = \mathcal{T}^i(I^i, \Delta^i \cup \Delta^{-i})$$

where $\Delta^{-i} = \bigcup_{j\in\mathcal{A},\ j\neq i} \Delta^j(j)$.

It is not difficult to show that the operator $\mathcal{T}_{\mathcal{A}}$ is monotonic.

Lemma 2 ($\mathcal{T}^{\mathcal{A}}$ **is monotonic).**
Let $\langle \overline{I}, \overline{\Delta}\rangle$ and $\langle \overline{J}, \overline{\Gamma}\rangle$ be such that $\langle \overline{I}, \overline{\Delta}\rangle \subseteq \langle \overline{J}, \overline{\Gamma}\rangle$. Then

$$\mathcal{T}_{\mathcal{A}}(\overline{I}, \overline{\Delta}) \subseteq \mathcal{T}_{\mathcal{A}}(\overline{J}, \overline{\Gamma}).$$

Proof. Let:

- $\langle \overline{I_1}, \overline{\Delta_1}\rangle = \mathcal{T}_{\mathcal{A}}(\overline{I}, \overline{\Delta})$
- $\langle \overline{J_1}, \overline{\Gamma_1}\rangle = \mathcal{T}_{\mathcal{A}}(\overline{J}, \overline{\Gamma})$

We need to show that, for each i, $\langle I_1^i, \Delta_1^i\rangle \subseteq \langle J_1^i, \Gamma_1^i\rangle$. By definition, for all i, $\langle I_1^i, \Delta_1^i\rangle = \mathcal{T}^i(I^i, \Delta^i \cup \Delta^{-i})$. By the hypothesis $\langle \overline{I}, \overline{\Delta}\rangle \subseteq \langle \overline{J}, \overline{\Gamma}\rangle$, it is clear that $\Delta^{-i} \subseteq \Gamma^{-i}$ and hence $\langle I_1^i, \Delta_1^i\rangle = \langle I^i, \Delta^i \cup \Delta^{-i}\rangle \subseteq \langle J^i, \Gamma^i \cup \Gamma^{-i}\rangle$. By the monotonicity of $\mathcal{T}^i$ it follows that $\mathcal{T}^i(I^i, \Delta^i \cup \Delta^{-i}) \subseteq \mathcal{T}^i(J^i, \Gamma^i \cup \Gamma^{-i}) = \langle J_1^i, \Gamma_1^i\rangle$. □

The monotonicity of $\mathcal{T}^{\mathcal{A}}$ allows us to give a bottom-up characterisation of the semantics of a multi-agent system as a whole, similarly to what we have done in Definition 12 for a single agent. In the next definition we denote by $\mathcal{T}_{\infty}^{\mathcal{A}}(\overline{\emptyset}, \overline{\Delta_0})$ the least fix-point of $\mathcal{T}^{\mathcal{A}}$, obtained by repeatedly applying it starting from the initial tuple $\langle \overline{\emptyset}, \overline{\Delta_0}\rangle$, where, for each i, Δ_0^i is a (possibly empty) initial plan for the agent i.

Definition 14. *Given a multi-agent system $\mathcal{A}$, and and initial plan Δ_0^i for each $i \in \mathcal{A}$, let $\langle \overline{I}, \overline{\Delta}\rangle = \mathcal{T}_{\infty}^{\mathcal{A}}(\overline{\emptyset}, \overline{\Delta_0})$. Then the* concrete semantics $\mathcal{S}^{\mathcal{A}}(\overline{\Delta_0})$ *of the system is defined as follows:*

$$\mathcal{S}^{\mathcal{A}}(\overline{\Delta_0}) = \langle \overline{I}, \overline{\Delta}\rangle$$

Notice that the semantics of the system as a whole is defined even if the semantics of some or all of the agents in the system is undefined. This is somewhat an arbitrary decision, that could be changed according to the needs of applications.

Example: A Fully Transparent Multi-agent System Concrete Semantics. We show how the operator $\mathcal{T}^{\mathcal{A}}$ behaves in the case of the multi-agent system of Section 5.2. The process is summed up by the following table, where rows represent the iteration steps and columns represent the agents. In the example, the initial plans are empty as far as agents 2 and 3 are concerned, whereas the initial plan of agent 1 consists of asking to agent 2 for the resource. We highlight in boldface the pairs $\langle I^i, \Delta^i \rangle$ which do not change in the future iterations. Hence the operator's fix-point is obtained by the tuple composed by the boldface pairs.

1	2	3
$\langle \{friend(2)\}, \{ask(1,2,r)\} \rangle$	$\langle \emptyset, \emptyset \rangle$	$\langle \{friend(2), have(r)\}, \{\} \rangle$
$\langle \{friend(2)\}, \{ask(1,2,r)\} \rangle$	$\langle \emptyset, \{ask(1,2,r)\} \rangle$	$\langle \{\mathbf{friend(2)}, \mathbf{have(r)}\}, \{\mathbf{offer(3,2,r)}\} \rangle$
$\langle \{friend(2)\}, \{ask(1,2,r)\} \rangle$	$\langle \{have(r)\}, \{ask(1,2,r), offer(3,2,r)\} \rangle$	
$\langle \{friend(2)\}, \{ask(1,2,r)\} \rangle$	$\langle \{\mathbf{have(r)}\}, \{\mathbf{ask(1,2,r)}, \mathbf{offer(3,2,r)}, \mathbf{give(2,1,r)}\} \rangle$	
$\langle \{friend(2), getfrom(2,r)\}, \{ask(1,2,r), give(2,1,r), thank(1,2)\} \rangle$		
$\langle \{\mathbf{friend(2)}, \mathbf{getfrom(2,r)}, \mathbf{g}\}, \{\mathbf{ask(1,2,r)}, \mathbf{give(2,1,r)}, \mathbf{thank(1,2)}, \mathbf{g}\} \rangle$		

From the fix-point, we can extract the set

$$\Delta = \{ask(1,2,r), give(2,1,r), thank(1,2), offer(3,2,r)\}$$

of the actions performed by each agent (and hence their single semantics). It is worth noting that this set coincides with the stable set shown in Section 5.2.

Indeed, we conjecture that a stable set can be constructed from the fix-points of the operator $\mathcal{T}^{\mathcal{A}}$. If this is the case, the latter can be seen as a way of incrementally building stable sets for the multi-agent system.

6 Related Work

Viroli and Omicini in [8] view a multi-agent system (MAS) as the composition of observable systems. The focus on observation is based, like in our framework, on the assumption that the hidden part of an agent manifests itself through interactions with the environment, and on how an agent makes its internal state perceivable in the outside. However, our work further distinguishes between different kinds of environment accessibility by agents through the use of visibility projection functions used by these

agents. In addition, we combine observable behaviour with the mental state of the agent, so as to permit to have partial access to the mental state of an agent in order to prove properties that are useful to a MAS, e.g. by allowing MAS designers to tests the desires against the mental state of an agent, without necessarily revealing/computing the full mental state.

Wooldridge and Lomuscio in [9] define a family of multi-modal logics for reasoning about the information properties of situated computational agents. They distinguish between what is objectively true in the environment, which in our approach is defined by what holds true in the world, the information that is visible, which our approach does not provide, information that an agent perceives, as with our observations, and finally information that the agent knows of the environment, which in our framework is defined by the mental state of an agent. Apart from the fact that we do not use a modal logic semantics, we also differ in the way we understand an environment. Wooldridge and Lomuscio's work is based on a definition often found in distributed systems [10], in that an environment does not contain the other agents (a bit like our notion of world). Instead in our approach the environment of an agent contains the state of the world and the other agents, and is closer to [11].

Another related approach to our work, presented by Ashri et al. in [12], is the identification and management of relationships in MASs. A formal model of the different kinds of relationships formed between interacting agents is presented and the way such relationships impact the overall system functioning is being investigated. If relationships between agents can be seen as properties, their work is similar to ours in that it attempts to identify properties in relation to observable parts of the environment in an application neutral manner. In this context, their way of managing relationships using control mechanisms can be thought in our terms as the required mechanisms that can be used to compute the semantics. However, Ashri et al. focus more on finding dependencies and influences between agent actions in the environment and less upon our concern of proving properties using the notion of stability.

Computational Logic approaches for formally describing and understanding MASs systems have been proposed in the past, e.g. [13, 14, 15], and are being pursued currently, possibly enhanced with other techniques, like Temporal Model Checking in [16]. Closer to our work is the work on the ALIAS system [17, 13], which relies on abductive logic programming to define a MAS. One major difference between ALIAS and our work is that agents in ALIAS have all the part of their mental states public, while in our approach part of the mental state needs to be public to the designer only.

7 Conclusions

We have proposed a semantics for multi-agent systems and a catalogue of properties for individual agents, multi-agent systems, and agents in multi-agent systems that we believe to be useful to aid the designers of concrete applications. Our semantics is fully declarative and abstract, and does not rely upon any concrete agent architecture or model, except for assuming that the semantics of individual agents is given in terms of their (public) observable behaviour and (private) mental state. We have illustrated the proposed notions for concrete abductive logic agents, whose beliefs are held within

an abductive logic program, and whose mental state and observable behaviour is given by adapting the T_p operator for logic programming. We have adopted a qualitative approach to the definition of success of agents, rather than assuming they are equipped with quantitative utility functions. The resulting model is not based upon game-theoretic concepts, but it would be interesting to compare/integrate our approach with that theory, e.g., comparing our notion of stable set with that of Nash equilibrium.

Other notions of individual welfare, different from the notion of individual success, would also be interesting. For example, we could consider maximising the number of achieved goals. Also, rather than having a "yes-no" kind success, we could compare multi-agent systems in terms of how close to success they are.

As future work, we plan to investigate the relationships between fix-points of the $\mathcal{T}^{\mathcal{A}}$ operator, i.e., the concrete semantics of a multi-agent system, and stable sets of $\mathcal{A}$, as described in the final example of Section 5.3. A further important problem for future studies is that of identifying means for the automatic verification of properties of multi-agent systems, in terms of properties of the individual agents composing them. This would aid the effective design of the such systems for the solution of concrete problems. Additional, less simplistic instances of our framework would also be interesting, e.g., 3APL agents [18]. In particular, we plan to adopt this framework for KGP agents, as defined in [19], and study the problem of properties verification in that context.

Acknowledgments

We would like to thank the anonymous referees for their valuable comments. This work has been supported by the SOCS project (IST-2001-32530), funded under the EU Global Computing initiative. The last two authors would also like to acknowledge support from the Italian programme "Rientro dei cervelli".

References

1. Sadri, F., Toni, F., Torroni, P.: Dialogues for negotiation: agent varieties and dialogue sequences. In: Intelligent Agents VIII: 8th International Workshop, ATAL 2001, LNAI 2333, Springer-Verlag (2002)
2. Apt, K.R.: Logic programming. In: Handbook of Theoretical Computer Science. Volume B. Elsevier Science Publishers (1990) 493–574
3. Kowalski, R.A., Sadri, F.: From logic programming towards multi-agent systems. Annals of Mathematics and Artificial Intelligence **25** (1999) 391–419
4. Sadri, F., Toni, F., Torroni, P.: An abductive logic programming architecture for negotiating agents. In Greco, S., Leone, N., eds.: Proceedings of the 8th European Conference on Logics in Artificial Intelligence (JELIA), LNCS 2424, Springer-Verlag (2002)
5. Toni, F., Stathis, K.: Access-as-you-need: a computational logic framework for flexible resource access in artificial societies. In: Proceedings of the Third International Workshop on Engineering Societies in the Agents World (ESAW'02), LNAI 2577, Springer-Verlag (2002)
6. Kakas, A., Kowalski, R.A., Toni, F.: Abductive Logic Programming. Journal of Logic and Computation **2** (1993) 719–770
7. Kakas, A., Mancarella, P.: Generalized stable models: a semantics for abduction. In: Proc. 9th European Conference on Artificial Intelligence, Pitman Pub. (1990)

8. Viroli, M., Omicini, A.: Multi-agent systems as composition of observable systems. In Omicini, A., Viroli, M., eds.: AI*IA/TABOO Workshop - Dagli oggetti agli agenti: tendenze evolutive dei sistemi software" (WOA 2001). (2001)
9. Wooldridge, M., Lomuscio, A.: A logic of visibility, perception, and knowledge: completeness and correspondence results. Journal of the IGPL **9** (2001)
10. Fagin, R., Halpern, J.Y., Moses, Y., Vardi, M.Y.: Reasoning About Knowledge. MIT Press (1995)
11. Abramsky, S.: Semantics of Interaction. (Technical report) Available at `http://www.dcs.ed.ac.uk/home/samson/coursenotes.ps.gz`.
12. Ashri, R., Luck, M., d'Inverno, M.: On identifying and managing relationships in multi-agent systems. In: Proc. of 18th International Joint Conference on Artificial Intelligence (IJCAI03), Acapulco, Mexico (2003)
13. Ciampolini, A., Lamma, E., Mello, P., Toni, F., Torroni, P.: Co-operation and competition in *ALIAS*: a logic framework for agents that negotiate. Computational Logic in Multi-Agent Systems. Annals of Mathematics and Artificial Intelligence **37** (2003) 65–91
14. Alferes, J., Brogi, A., Leite, J.A., Pereira, L.M.: Computing environment-aware agent behaviours with logic program updates. In Pettorossi, A., ed.: Logic Based Program Synthesis and Transformation, 11th International Workshop, (LOPSTR'01), LNCS 2372, Springer-Verlag (2002)
15. Alferes, J.J., Brogi, A., Leite, J.A., Pereira, L.M.: Evolving logic programs. In Flesca, S., Greco, S., Leone, N., Ianni, G., eds.: Proceedings of the 8th European Conference on Logics in Artificial Intelligence (JELIA'02), LNAI 2424, Springer-Verlag (2002)
16. Pokorny, L.R., Ramakrishnan, C.R.: Modeling and verification of distributed autonomous agents using logic programming. In: Proceedings of the Workshop on Declarative Agent Languages and Technologies (DALT'04), LNCS 3476, Springer-Verlag (2005). In this volume.
17. Ciampolini, A., Lamma, E., Mello, P., Torroni, P.: Rambling abductive agents in ALIAS. In: Proc. ICLP Workshop on Multi-Agent Sytems in Logic Programming (MAS'99), Las Cruces, New Mexico (1999)
18. Dastani, M., de Boer, F.S., Dignum, F., van der Hoek, W., Kroese, M., Meyer, J.C.: Programming the deliberation cycle of cognitive robots. In: Proc. of 3rd International Cognitive Robotics Workshop (CogRob) (2002)
19. Kakas, A., Mancarella, P., Sadri, F., Stathis, K., Toni, F.: The KGP model of agency. In: Proceedings of the 16th European Conference on Artificial Intelligence (ECAI), Valencia, Spain (2004)