Stable Multi-agent Systems

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

[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 present an abstract declarative semantics for multi-agent systems based on the idea of *stable set*, and argue that it can be suitably employed to describe, and to some extent verify, the dynamics of complex systems of autonomous and heterogeneous interacting agents. We view agents as black-boxes, whose semantics is abstractly understood as an input-output transformation from the agents' observations about their environment, to the actions they perform. Stable sets (of actions) characterise multi-agent systems able to reach an equilibrium point. Our semantics via stable sets takes into account the possibility that agents may fail. We illustrate how stability can characterise multi-agent systems by means of examples. We also draw considerations about how stable sets can be effectively approximated.

1 Introduction

The increasing complexity of software development calls for enhanced methods supporting the design, development and verification phases in the life-cycle of applications. Such methods are required to be formal, possibly supported by automated tools, and at an architectural level. Indeed, coding is no longer the main activity in building applications, but rather the emphasis is on the definition of the components which constitute an application and their relationships within an overall architecture. This approach requires models and verification tools, which are neither applicable, nor needed, when developing code. Moreover, the advent of a network-centric model of computation fosters the development of applications based on interacting components that may be *autonomous*, i.e. independent computational units with their own goals, possibly belonging to different domains, and *heterogeneous*, e.g. following different programming paradigm.
Many of the models and techniques developed within the field of Multi-agent systems (MAS) appear to be successfully applicable in the aforementioned context. Indeed, MAS feature architectures of autonomous and heterogeneous "intelligent components," which interact with one another in the environment where they are situated. There are competing models for agent programming (such as BDI[1], AgentSpeak(L)[2], 3APL[3], IMPACT[4], KGP[5], to cite but a few), and MAS design methodologies (such as Gaia [6] and Prometheus [7]), as well as paradigms to describe and reason about the way in which they can interact and coordinate their tasks, possibly in cooperative or competitive ways (such as

LAILA [8]). Moreover, a lot of research in MAS has been traditionally influenced by other disciplines, like economics, ecology, psychology, which have contributed to better understand the "organisational" aspects of such systems.

Taking into consideration the amount of different agent programming models proposed, we aim at developing a high-level description framework, which, by reasoning at an abstract semantical level, allows us to formally model the evolution of a MAS, by abstracting away from the peculiarities of a specific agent programming model, paradigm, or methodology. In this paper we show how our abstract approach, originally introduced in [9], can be adapted to the study of MAS, as well as distributed applications in general, where agents/components interact in order to fulfill their own goals, but may fail under certain conditions.

We view agents as "black-boxes", whose "semantics" is expressed as an input-output transformation describing the behaviour of agents in their environment. Given the *environment* of the agent, which may contain the "observable behaviour" of the other agents in the MAS, and an *initial plan* (i.e. a set of actions the agent intends to execute in order to accomplish its goals), the semantics of an agent determines (i) its *observable behaviour*, as an output set of actions from the pool of actions that the agent can perform, and (ii) its *mental state*, which is private and thus inaccessible to other agents. This consists of a representation of the knowledge of the agent, which may include its goal, plans, constraints, etc. In addition, the mental state of an agent and its beliefs about the environment in which it is situated may report a failure condition, for instance when the agent is not able to plan for its goal with respect to a dynamically changing and possibly partially accessible environment. The framework we propose is intended to be instantiated for any concrete agent architecture/theory that can be abstracted away in terms of the aforementioned "semantics".

Building on top of the above agent semantics, we define a notion of *stability* on the set of all actions performed by all agents in the system, and we characterise "good" MASs, as those reaching, by means of the "coordinated contribution" of their agents, an equilibrium point, where, intuitively speaking, no agent needs to further operate within the system and accepts the currently reached state. 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. This notion of stability is reminiscent of that of Nash equilibrium state in economics game-theory [10], where players accept a sort of "optimal compromise", and has been also inspired by the notion of stable model semantics for non-monotonic logic [11], according to which a model for a non-monotonic knowledge base exists if a "coherent compromise" between positive and negative knowledge can be reached (the detailed comparison with these works is out of the scope of this paper).

For the purpose of verification, in [9] we have shown how this kind of approach can be used to formalise properties, like individual success of agents and overall success, robustness and world-dependence of a MAS. Moreover, we have also shown how, in a specific case of "well-behaved" logic-based agents, stable sets can be constructively approximated (by adapting well known semantic approximation techniques of Computational Logic, viz. the T_P bottom-up operator [12]).

Here, we extend the approach of [9] by considering the possibility that agents may fail at some stage, a possibility which appears relevant from the viewpoint of engineering complex systems. An agent is in a failure state when it is not able to "properly" operate within its environment. The new approach is illustrated by examples in the context of the well-known Blocks World, chosen as a simple and paradigmatic scenario for the interaction of planning agents. Specifically for this context, an agent may fail when its planned course of actions would lead to a violation of some physical law, like the impossibility for two different blocks to be in the same position. This situation, which may be temporary, is represented by the agent semantics as a failure mental state ($\perp$) and an empty set of actions in response of the current environment and plan. It is worth noting that the agent metaphor we use in the rest of the paper, could be recast in terms of more generic *components* and adapted to different verification scenarios.

We first define the agent semantics and the notion of stable set, showing an example of successful cooperation for agents in a MAS, which corresponds to the existence of a stable set, and an example of failure, for which a stable set does not exists. Then we discuss how stable sets can be approximated by temporarily suspending failing agents, until they are able to reconcile their mental states with the current environment, and letting the successful ones operate. If all the agents are in a failure state we say that the MAS is *compromised*. Few simple examples show cases in which this way of operating may or may not lead to the construction of stable sets, according to which the MAS can evolve. To illustrate the generality of the approach in a simple way, we have abstracted away from modeling the details of time evolution and interleaving of actions. Finally, a brief comparison with similar approaches and some concluding remarks are reported.

2 Single Agent Abstract Semantics

The semantics of single agents is defined in an input-output style, by abstracting away from the agents' internals and, in particular, independently of any programming paradigm. We also refer informally to goals, plans and knowledge of agents, as well as failure and success of agents.

The input for an agent semantics consists of *(i1)* a representation of the world in which the agent is situated (referred to as its *environment*), which may encompass events occurred in the world and actions performed by other agents in the system, and *(i2)* an *initial plan*, namely a set of actions, that the agent has decided to execute in order to fulfill its goals. The output consists of *(o1)* the information that the agent is able to derive, (referred to as the *knowledge* or *mental state* of the agent), possibly encompassing both its goals and a representation of the world that it has observed, and *(o2)* the set of *actions* that

the agent decides to perform as a consequence of its inputs (typically, this set includes the actions of the original plan).
Moreover, each agent is equipped with a notion of *failure,* represented as $\perp$. This is used to represent any circumstance in which the agent is not able to cope with the environment, e.g. it is not able to devise any plan, or its observations are not coherent with some of its constraints. A failed agent is required not to commit to the execution of any action. For the sake of 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 items in $A(i)$) and executed in the "proper" order.
We indicate agents with $1, 2, \ldots i, \ldots n$, and with $A(i)$ and $O(i)$ the set of actions and observations of agent i, respectively. Given a set $\Delta \subseteq \cup_i(A(i) \cup O(i))$, $\Delta(j) = \Delta \cap A(j)$ is the set of actions in Δ pertaining to agent j.

Definition 1. *Given an agent i, its* input-output semantics *is indicated as*

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

where $\Delta_{in} \subseteq O(i)$ *and* $\Delta_0 \subseteq A(i)$ *are the* observations *and* initial plan *of the agent, respectively,* Δ_{out} *is the* set of actions *of the agent, and* M *is the* mental state *of the agent, such that either* 1) $M = \perp$ *and* $\Delta_{out} = \emptyset$*, and the agent is* failed*, or* 2) $M \neq \perp$ *and* $\Delta_0 \subseteq \Delta_{out} \subseteq A(i)$*, and the agent is* successful.

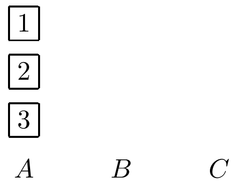
The mental state M, that can be logically understood as a *model* for the agent, is typically *private* to the agent itself, while the set of output actions Δ_{out} is the *public* side of the agent, observable by all the other agents in the system. The initial plan Δ_0 of the agent can be thought of as being determined by the agent itself, whereas the observations Δ_{in} might be about other agents. Notice that, given Δ_{in} and Δ_0, $\mathcal{S}^i(\Delta_{in}, \Delta_0)$ might not be unique (namely $\mathcal{S}^i$ may not be a function), since in general agents may exhibit non-deterministic behaviours. However, in all the examples in this paper, the agent semantics will always be uniquely determined. The following example illustrates Definition 1.

Example 1. In the well-known Blocks World, blocks are piled in stacks, with the usual constraints that a block can be moved only if it is on top of a stack, two blocks can not be in the same place, etc. Since our framework abstracts away from the specific paradigms used to build agents, we adopt an informal, self-explaining, language to describe the mental states and actions of the agents, while the world is represented in pictorial form. We also assume that actions are performed in sequence, abstracting away from any formal representation of temporal ordering. Consider the situation in the figure, with blocks $1, 2$ and 3 on stack A, and an agent i with the initial plan of moving block 2 to stack C, represented as $\Delta_0 = \{2toC\}$. This plan is unfeasible since the planned action $2toC$ violates a law of the physical world (that one cannot move a block which is not clear). Let $\mathcal{W}$ be an appropriate representation of this situation. Then:

(a) The agent may be able to extend its plan, e.g. by first moving block 1 to B. Hence, we will have $\mathcal{S}^i(\mathcal{W}, \{2toC\}) = < M, \{1toB, 2toC\} >$, where M is any

appropriate mental state. In this case, the agent is successful.

(b) On the other hand, the agent may not be able to suitably revise Δ_0 to render it feasible, and it may end up in what we consider a failure state, whereby the planned action $2toC$ violates some internal constraint of the agent intended to enforce coherence with the physical world. In this case, $\mathcal{S}^i(\mathcal{W}, \{2toC\}) = < \bot, \emptyset >$, and the agent is failed.



3 MAS Declarative Semantics and Stability

We consider a *multi-agent system* (MAS) as a collection of n agents, $n \geq 2$, situated in a world $\mathcal{W}$. The semantics of the MAS is given in terms of the semantics of the agents that constitute it. We indicate with $\mathcal{W}^i$ the set of observations that agent i is able to draw from $\mathcal{W}$, namely $\mathcal{W}^i = \mathcal{W} \cap O(i)$. In the following we will use the shorthand $< X >_{\mathcal{Y}}$ for the tuple $< X^{i_1}, \ldots, X^{i_k} >$, with $\mathcal{Y} = \{i_1, \ldots, i_k\} \subseteq \{1, .., n\}$. Given $< X >_{\mathcal{Y}}$, X^i is the i-th element of the tuple. The semantics of a MAS can then be defined on top of the single-agent semantics, as follows.

Definition 2. *A* multi-agent system $\mathcal{MAS} =< \mathcal{A}, \mathcal{W} >$ *consists of a set of agents* $\mathcal{A} = \{1, \ldots, n\}$*,* $n \geq 2$*, and a world* $\mathcal{W}$*. Given the tuple* $< \Delta_{in} \cup \mathcal{W}, \Delta_0 >_{\mathcal{A}}$ *of observations and initial plans for each agent, the semantics of* $\mathcal{MAS}$ *is*

$$< M, \Delta_{out} >_{\mathcal{A}},$$

where, for all $i \in \mathcal{A}$*,* $\mathcal{S}^i(\Delta^i_{in} \cup \mathcal{W}^i, \Delta^i_0) =< M^i, \Delta^i_{out} >$*. A multi-agent system is* compromised *if, for all* $i \in \mathcal{A}$*,* $M^i = \bot$*.*

In a compromised MAS, all the agents are failed. However an autonomously changing environment might allow some of the agents to recover from failure. The input observations of the agents, Δ^i_{in}, refer to events in the world as well as actions by other agents. It is legitimate to characterise these observations so that each agent is aware of what all the others are doing within the system. This can be done by recursively defining the input observations of each single agent as depending on the output actions of all the other agents and those performed by the agent itself. The existence of a solution of such recursive definition of the semantics represents a stability condition of the system. Indeed, all the agents have agreed in a coordinated manner on a course of actions. Such stability condition can be defined as follows.

Definition 3. *A multi-agent system* $\mathcal{MAS} =< \mathcal{A}, \mathcal{W} >$ *is* stable *if there exists* $\Delta = \bigcup_{i \in \mathcal{A}} \Delta^i_{out}$*, such that, for each* $i \in \mathcal{A}$*,*

$$\mathcal{S}^i(\Delta^{-i} \cup \mathcal{W}^i, \Delta^i_0) =< M^i, \Delta^i_{out} >$$

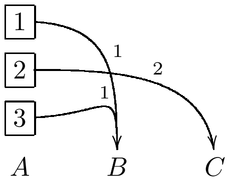
where $\Delta^{-i} = \bigcup_{\substack{j \in \mathcal{A} \\ j \neq i}} \Delta(j)$*. The set* Δ *is called a* stable *set for* $\mathcal{MAS}$*.*

Notice that Δ^{-i} is the set of all actions performed by all the agents except agent i. By the previous definition, the sets $\Delta^1_{out}, \ldots, \Delta^n_{out}$ are a solution for the set of mutually recursive equations

$$\mathcal{S}^1(\Delta^{-1} \cup \mathcal{W}^1, \Delta^1_0) =< M^1, \Delta^1_{out} >$$
$$\vdots$$
$$\mathcal{S}^n(\Delta^{-n} \cup \mathcal{W}^n, \Delta^n_0) =< M^n, \Delta^n_{out} >$$

where each Δ^{-i} occurring on the left-hand side of the $i - th$ equation is defined in terms of the Δ^j_{out} sets, occurring in all the other equations. Intuitively, 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 stability could consists in an infinite course of actions (e.g. when agents "steadily" keep on repeating their behaviour) and that stability does not imply the success of all the agents, indeed a failed agent might be part of a stable MAS. The following example illustrates a stable MAS.

Example 2. Consider again the Blocks World situation of Example 1, with the difference that now we have two agents operating according to the picture below. Agent 1 is responsible to move odd-numbered blocks in the stack B, while agent 2 is responsible to put even-numbered blocks in the stack C. Let us suppose that the agents have initially the goals $mvToB$ and $mvToC$, respectively. Trivially, the set:



$$\Delta = \{1toB_1, 2toC_2, 3toB_1\}$$

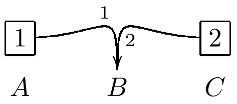
where each action is indexed according to its executor, is a stable set for the system. Indeed, being $\mathcal{W}$ an appropriate representation of the world

$$\mathcal{S}^1(\{2toC_2\} \cup \mathcal{W}^1, \{1toB_1, 3toB_1\}) = < M^1, \{1toB_1, 3toB_1\} >$$
$$\mathcal{S}^2(\{1toB_1, 3toB_1\} \cup \mathcal{W}^2, \{2toC_2\}) = < M^2, \{2toC_2\} >$$

where $M^1 \models mvToB$ and $M^2 \models mvToC$ (i.e. goals are satisfied). The previous equations can be read as "If agent 1 observes that agent 2 is moving block 2 to C, then it will move blocks 1 and 3 to B, while if agent 2 observes that blocks 1 and 3 are moved by agent 1, then it will move block 2 to C". Finally, note that we overlook here, as in the rest of the paper, issues concerning the treatment of time and ordering between actions.

The relevance of the existence of stable sets for a MAS is due to their interpretation as viable courses of actions that satisfy all the agents present in the system, given the current state of the world. The next example illustrates a case where the lack of a stable set corresponds to the impossibility for the agents in the system to coordinately accomplish their tasks, without resulting in a failure. Here, failure is due to the violation of a physical law of the world, which occurs as a consequence of the sum of the actions performed by the agents.

Example 3. Let us consider the different situation of the Blocks World illustrated below, where block 1 is on stack A and block 2 is on stack B. It is easy to agree on the fact that, being $\Delta^1_0 = \{1toB_1\}$, and $\Delta^2_0 = \{2toB_2\}$, there is no stable set for the system. Indeed both agents would like to place their block in the same position, ending up in a failure state. Differently from Example 2, where agent 2 resolves its initial failure state by coordinating its behaviour with that of agent 1, the MAS is here compromised.



4 Constructing Stable Sets by means of Successful Agents

In [9] we have shown how stable sets can be constructed for the case of a simple agent programming language based on Abductive Logic Programming. In that context, stable sets can be approximated by exploiting well known bottom-up techniques, traditionally used in Computational Logic. Here, we illustrate how stable set construction can be approached by means of a modification of the same technique, with respect to any agent architecture and language that can be understood abstractly in terms of the defined input-output semantics.

Informally speaking, starting from a current *partial* state of a system, the input for the single agent semantics is extracted, and, if possible, a more defined semantics for the system is returned, taking into consideration the actions executed by the agents according to their semantics and the current inputs. A bit of care is necessary in order to select a (maximal) subset of agents that can successfully operate within the system. This step can be (possibly infinitely) repeated to constructively approximate, if any, a stable set. Suitable assumptions on agent languages may guarantee the convergence of the method, as shown in [9].

The construction of the stable set in Example 2 relies upon agent 2 "waiting" until the state of the world has become consistent with its plans. Recall that failed agents, whose mental state can not deal with the current state of the world, could play a part in future states that the MAS can reach. In a sense, we impose that the set of executed actions at each step represents a stable set for the restricted system of currently successful agents. If the system is compromised, the semantics results in empty sets of actions and all the mental states are $\bot$. The step-wise semantic approximation is defined as follows in terms of the $\mathcal{T}^{\mathcal{A}}$ operator, which maps (tuples of) observations (actions by other agents and events in the world) onto (tuples of) mental states and new observations.

Definition 4 ($\mathcal{T}^{\mathcal{A}}$ operator). *Let $\mathcal{MAS} =< \mathcal{A}, \mathcal{W} >$ be a multi-agent system, and $< \Delta >_{\mathcal{A}}$ be a tuple of sets of actions. The $\mathcal{T}^{\mathcal{A}}$ operator is defined as follows:*

$$\mathcal{T}^{\mathcal{A}}(< \Delta \cup \mathcal{W} >_{\mathcal{A}}) = \begin{cases} < J, \Gamma >_{\mathcal{A}} & \textit{if } \mathcal{A}^+ = \mathcal{A} \\ \\ \mathcal{T}^{\mathcal{A}}(< \Delta \cup \mathcal{W} >_{\mathcal{A}^+}) \oplus < \bot, \Delta >_{\mathcal{A}^-} & \textit{otherwise} \end{cases}$$

where $\mathcal{A} = \mathcal{A}^+ \cup \mathcal{A}^-$ such that

$$\forall k \in \mathcal{A}^+.\ \mathcal{S}^k(\mathcal{W}^k \cup \Delta^{-k}, \Delta^k) = < J^k, \Gamma^k > \neq < \bot, \emptyset >$$

$$\forall k \in \mathcal{A}^-.\ \mathcal{S}^k(\mathcal{W}^k \cup \Delta^{-k}, \Delta^k) = < \bot, \emptyset >$$

and $\oplus$ merges tuples according to the order induced by agent names.

The previous definition can be read as follows. Given the observations $< \Delta >_{\mathcal{A}}$ of the agents and the environment $\mathcal{W}$ (which, for simplicity, we assume to be fixed and unchanging), the MAS is partitioned into:

1. the set $\mathcal{A}^+$ of agents that, taking into consideration the up-to-now plans of the other agents, the world and their own committed actions (at some previous step) $\mathcal{S}^k(\mathcal{W}^k \cup \Delta^{-k}, \Delta^k)$, are *successful* ($J^k \neq \bot$), and
2. the set $\mathcal{A}^-$ of agents that, taking into consideration the up-to-now plans of the other agents, the environment and their own committed actions (at some previous step) $\mathcal{S}^k(\mathcal{W}^k \cup \Delta^{-k}, \Delta^k)$, are *failed* ($J^k = \bot$).

The $\mathcal{T}^{\mathcal{A}}$ operator returns a "more defined" semantics for the overall MAS obtained by recursively seeking ($\mathcal{T}^{\mathcal{A}}(< \Delta \cup \mathcal{W} >_{\mathcal{A}^+})$) whether the restricted set of potentially successful agents $\mathcal{A}^+$ may successfully agree on a set of actions, without taking into consideration the actions of the currently failing agents in $\mathcal{A}^-$. The more defined semantics for the overall MAS is returned as soon as a (sub-)set of "reciprocally" successful agents is found (and in this case their contribution to the system is recombined with the previous one by the idle agents, $\oplus$), or all the agents are failed and the MAS is compromised ($\mathcal{A}^- = \mathcal{A}$).
The world is a parameter of $\mathcal{T}^{\mathcal{A}}$, which is supposed not to change while agents are coordinating themselves, while the mental states are recomputed at each successive application of $\mathcal{T}^{\mathcal{A}}$ for the successful (active) agents. In this way, recomputing at each iteration their mental states, agents are forced to check their consistency against the new situations. The performed actions are recorded in the output of the operator and then used as input at the next application of $\mathcal{T}^{\mathcal{A}}$, and hence, following the idea of stability, notified to all the agents. More precisely,

$$\mathcal{T}^{\mathcal{A}}_{i+1}(< \Delta \cup \mathcal{W} >_{\mathcal{A}}) = \mathcal{T}^{\mathcal{A}}(< \Gamma \cup \mathcal{W} >_{\mathcal{A}}),$$

where $\mathcal{T}^{\mathcal{A}}_i(< \Delta \cup \mathcal{W} >_{\mathcal{A}}) =< J, \Gamma >_{\mathcal{A}}$.
In [9] we have shown that, having chosen a specific agent language based on Abductive Logic Programming, the corresponding $\mathcal{T}^{\mathcal{A}}$ operator enjoys convergence properties to a minimal fix-point, from which a stable set can be extracted. However, notice that in [9] consistency issues were not taken into account.
Provided that, for a chosen agent language, the $\mathcal{T}^{\mathcal{A}}$ operator of Definition 4 does converge, we can give the following Definition 5, where $\mathcal{T}^{\mathcal{A}}_{\infty}$ denotes the fix-point of $\mathcal{T}^{\mathcal{A}}$, as a constructive way of approximating, if any, a stable set. This definition is relevant as a basis on which a verification methodology can be developed, according to the idea that the existence of stable sets guarantees the overall good engineering of the system. The study of general conditions for the convergence of $\mathcal{T}^{\mathcal{A}}$ to a stable set is scope for future work. However, we will give an example to show how Definition 5 captures the construction of a stable set for a Blocks World which has a stable set, and another one which shows how Definition 5 correctly fails to produce a stable set in a case where a stable set does not exist.

Definition 5. *Given a multi-agent system* $\mathcal{MAS} =< \{1,\ldots,n\}, \mathcal{W} >$ *and a set of initial plans* $< \Delta_0 >_{\mathcal{A}}$*, the* concrete semantics *of* $\mathcal{MAS}$ *is defined as*

$$< J, \Delta >_{\mathcal{A}} = \mathcal{T}^{\mathcal{A}}_{\infty}(< \Delta_0 \cup \mathcal{W} >_{\mathcal{A}}).$$

The next example shows how a stable set can be derived from $< J, \Delta >_{\mathcal{A}}$, the result of the application of $\mathcal{T}^{\mathcal{A}}_{\infty}$. More precisely, the iteration of the application of $\mathcal{T}^{\mathcal{A}}$ converges to a fix-point after few iterations.

Example 4. The stable set of the MAS in Example 2 can be approximated, and, actually, constructed, by few applications of the $\mathcal{T}^{\mathcal{A}}$ operator. Assuming that the agents have the goals of Example 2, and are not provided with an initial plan, the concrete semantics of the MAS is given by

$$\mathcal{T}^{\mathcal{A}}_{\infty}(<< \mathcal{W} >, < \mathcal{W} >>).$$

While calculating $\mathcal{T}^{\mathcal{A}}_1(<< \mathcal{W} >, < \mathcal{W} >>)$, the second agent can not find a partial plan, i.e. $\mathcal{S}^2(\mathcal{W}, \emptyset) =< \bot, \emptyset >$, indeed there is nothing it can do at present. Hence $\mathcal{A}^+ = \{1\}$ and $\mathcal{A}^- = \{2\}$:

$$\mathcal{T}^{\mathcal{A}}(<< \mathcal{W} >, < \mathcal{W} >>) \ = << M^1_1, \{1toB_1\} >, < \bot, \emptyset >> .$$

Analogously, at the next step agent 2 moves block 2 (accomplishing its goal), and agent 1 is suspended ($\mathcal{W}$ is updated with executed actions):

$$\mathcal{T}^{\mathcal{A}}(<< \{1toB_1\} \cup \mathcal{W} >, < \{1toB_1\} \cup \mathcal{W} >>) \ = << \bot, \emptyset >, < M^2_1, \{2toC_2\} >> .$$

Finally, also agent 1 completes its task *mvToB* (since $\mathcal{S}^1(\mathcal{W} \cup \{2toC_2\}, \{1toB_1\}) = < M^1_2, \{1toB_1, 3toB_1\} >$):

$$\begin{gathered}\mathcal{T}^{\mathcal{A}}(<< \{2toC_2, 1toB_1\} \cup \mathcal{W} >, < \{2toC_2, 1toB_1\} \cup \mathcal{W} >>) = \\ << M^1_2, \{1toB_1, 3toB_1\} >, < M^2_2, \{2toC_2\} >> .\end{gathered}$$

which is a fix-point of $\mathcal{T}^{\mathcal{A}}$, unless the world changes or agents introduce new plans. The union of the output actions of the agents is the stable set of Example 2.

It is also interesting to verify that a stable set cannot be constructed for the case of Example 3, where two agents are attempting to execute plans that make both the agents not consistent, when executed in the same environment.

Example 5. Let us try to construct a stable set for the Blocks World of Example 3, whose concrete semantics, given the initial plans of the agents, is:

$$\mathcal{T}^{\mathcal{A}}_{\infty}(<< \{1toB_1\} \cup \mathcal{W} >, < \{2toB_2\} \cup \mathcal{W} >>).$$

Let us assume that, given the current state of the world, neither agents need to generate further actions in order to accomplish their tasks, and that, without knowing what the other agent is doing, each one can reach a successful mental state encompassing the actions performed in this first step: $\mathcal{S}^1(\mathcal{W}, \{1toB_1\}\}) = < M^1, \{1toB_1\} >$, and $\mathcal{S}^2(\mathcal{W}, \{2toB_2\}) =< M^2, \{2toB_2\} >$. We have that in two steps, the agents, aware of each other's actions, end up in a failure state, and the MAS is compromised:

$$\begin{gathered}\mathcal{T}^{\mathcal{A}}(<< \{1toB_1\} \cup \mathcal{W} >, < \{2toB_2\} \cup \mathcal{W} >>) =<< M^1, \{1toB_1\} >, < M^2, \{2toB_2\} >> . \\ \mathcal{T}^{\mathcal{A}}(<< \{2toB_2, 1toB_1\} \cup \mathcal{W} >, < \{2toB_2, 1toB_1\} \cup \mathcal{W} >>) =<< \bot, \emptyset >, < \bot, \emptyset >> .\end{gathered}$$

5 Related Work

The work we present in this paper is close to several approaches, based on Computational Logic, whose aim has been to provide formal models to understand MAS environments, like [13–15]. While we share with many of these proposals the use of well known logic-based techniques, like bottom-up approximations and the idea itself of stability, the distinguishing aim of our work has been to devise a model for MAS, which, independently of specific agent paradigms, allows us to reason at an abstract and "declarative" level. Moreover, we also aim to define, in agreement with the Computational Logic tradition, an operational counterpart for the declarative settings. This will enable us to support forms of automated verifications.

Modal logic approaches whose aim has been to provide frameworks for proving properties of MAS are also well-documented, for example, see the framework of Lomuscio and Sergot [16] on *deontic interpreted systems*. Earlier work of Wooldridge and Lomuscio [17] define a family of multi-modal logics for reasoning about the information properties of computational agents situated in some environment. We differ from these approaches in the way we understand an environment. Their defininion of an environment is based on a definition often found in distributed systems [18], in that an environment does not contain the other agents. Instead in our approach the environment of an agent contains the state of the world and the other agents, and is closer to [19].

Other formal frameworks exist, for example, Viroli and Omicini in [20] view MAS as the composition of observable systems. These systems are 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, we differentiate ourselves from them by the kind of environment accessibility by agents, i.e. the way agents perceive other agents in terms of their performed actions.

6 Final Remarks

We have illustrated how a semantic characterisation of MAS can be used for checking whether the agents in the system can successfully cooperate, and hence the system can be considered well designed. In particular, in this paper, extending the approach previously introduced in [9], we have addressed the issue of dealing with agents that may temporarily fail, not being "consistent" with the environment in which they operate. We have shown how the notion of stable set, on which the approach is based, can be adapted to deal with this case. Both the semantics and the methodologies adopted are inspired mainly by the field of Computational Logic, but they also appear in other areas, like Economics and Game Theory. We have illustrated the proposed notions by applying them to the simple scenario of Blocks World.

Approaching the modeling of a complex system, like a MAS, at a semantic level allows us to define an abstract framework, which does not depend on the specific, possibly different, agent programming paradigms. However, further (semantical) characterisation of agents would be useful in order to better specify the properties that can be verified by means of the framework, and, also, the computational aspects of such verification. For instance, the choice of a specific agent language in [9] granted some preliminary results about the convergence of the bottom-up approximation of semantics. General conditions for the convergence of $\mathcal{T}^{\mathcal{A}}$, and more general properties of MAS and their stable sets, are scope for future work. A characterization of time at the agent language level, allowing for a more precise representation of plans as ordered sequence of actions, would add expressiveness to the framework, tightening the relation between agent and system semantics. The stable set represents an "ideal" course of actions, on which all the agents in the system agree, given the current state of the environment. This notion could be further exploited in order to characterise evolutions of MAS through stability conditions: the system evolves by means of agents aiming to stability, which may be compromised by either a variation in the environment or new plans/actions introduced by some of the agents in the system. Importantly, our declarative approach is provided with a computational counterpart, which seems amenable, under certain assumptions on agent semantics, to support automated verification. This issues, as well as its computational costs (for given classes of agent semantics) is currently under investigation.

Moreover, the relationships between our notion of stability and of that of Nash equilibrium [10], from the field of Economics, are worth being further studied.

Another interesting line of research is the study of the relations between stability and negotiation among agents. Actually, stable sets, when they exist, can be interpreted as a sort of shared agreement between agents. It would be interesting to study how agents can cooperate to the construction of a "preferred" stable set, by coordinating the course of actions they perform. Economics and Game Theory might also be applied.

Finally, we plan to extend our framework in order to incorporate social notions, such as social goals, joint goals amongst agents, social rules, conformance to them, and adoption of multi-agent system's expectations by individual agents. Also, we intend to adopt this extended framework for KGP agents, as defined in [5], and study the problem of properties verification in that context.

References

1. Rao, A.S., Georgeff, M.P.: An abstract architecture for rational agents. In: Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning (KRR92), Boston, MA (1992)
2. Rao, A.S.: AgentSpeak(L): BDI agents speak out in a logical computable language. In van Hoe, R., ed.: MAAMAW96. Volume 1038 of LNCS., Springer-Verlag (1996) 42–55
3. Hindriks, K.V., de Boer, F.S., van der Hoek, W., Meyer, J.C.: Agent programming in 3APL. Autonomous Agents and Multi-Agent Systems **2(4)** (1999) 357–401
4. Arisha, K.A., Ozcan, F., Ross, R., Subrahmanian, V.S., Eiter, T., Kraus, S.: IMPACT: a Platform for Collaborating Agents. IEEE Intelligent Systems **14** (1999) 64–72

5. 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). (2004)
6. Wooldridge, M., Jennings, N.R., Kinny, D.: The gaia methodology for agent-oriented analysis and design. Autonomous Agents and Multi-Agent Systems **3** (2000) 285–312
7. Padgham, L., Winikoff, M.: Prometheus: A methodology for developing intelligent agents. In: Proceedings of the Third International Workshop on AgentOriented Software Engineering at AAMAS 2002, (2002)
8. Ciampolini, A., Lamma, E., Mello, P., Torroni, P.: LAILA: A language for coordinating abductive reasoning among logic agents. Computer Languages **27** (2002) 137–161
9. Bracciali, A., Mancarella, P., Stathis, K., Toni, F.: On modelling declaratively multi-agent systems. In: Proc. of Declarative Agent Languages and Technologies (DALT 2004). LNCS, To appear (2004)
10. Nash, J.: Equilibrium points in n-person games. Proceedings of the National Accademy of Science (1950)
11. Gelfond, M., Lifschitz, V.: The stable model semantics for logic programming. In Kowalski, R., Bowen, K.A., eds.: Proceedings of the 5th International Conference on Logic Programming, MIT Press (1988) 1070–1080
12. Apt, K.R.: Logic programming. In: Handbook of Theoretical Computer Science. Volume B. Elsevier Science Publishers (1990) 493–574
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.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), Selected Papers, Springer-Verlag (2002) 216–232
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). Volume 2424 of LNCS, Springer-Verlag (2002) 50–61
16. Lomuscio, A., Sergot, M.: Deontic interpreted systems. In van der Hoek, W., Wooldridge, M., eds.: Studia Logica **75** (Special Issue on The Dynamics of Knowledge). Kluwer Academic Publishers (2003)
17. Wooldridge, M., Lomuscio, A.: A logic of visibility, perception, and knowledge: completeness and correspondence results. Journal of the IGPL **9** (2001)
18. Fagin, R., Halpern, J.Y., Moses, Y., Vardi, M.Y.: Reasoning About Knowledge. MIT Press (1995)
19. Abramsky, S.: Semantics of Interaction. (Technical report) Available at http://www.dcs.ed.ac.uk/home/samson/coursenotes.ps.gz.
20. Viroli, M., Omicini, A.: Multi-agent systems as composition of observable systems. In Omicini, A., Viroli, M., eds.: AI*IA/TABOO Joint Workshop - Dagli oggetti agli agenti: tendenze evolutive dei sistemi software" (WOA). (2001)