# A Model for Systems of Situated Autonomous Agents: an Application to Automated Deduction

**Jean-Pierre Müller**[1], **Paolo Pecchiari**[2]
[1]IIIA - University of Neuchâtel, rue Emile Argand 11, 2007 Neuchâtel, Switzerland
muller@info.unine.ch
[2]DIST - University of Genova, viale Causa 15, 16146 Genova, Italy
peck@dist.unige.it*

## Abstract

The goal of this paper is to present a model for systems of situated autonomous agents and to illustrate its application to (distributed) automated deduction. Our model is based on a sharp distinction between the environment and agent dynamics. In particular, it allows to express how the environment is modified by actions performed simultaneously; to formalize in a very general way the internal activity of an agent in terms of its decision processes; and, finally, to describe the interactions between the agents and the environment, and among themselves. In the application to automated deduction, we integrate our model with the concept of OMRS (Open Mechanized Reasoning System) (Giunchiglia, Pecchiari, & Talcott 1994). This concept intends to be a first small step towards complex reasoning systems built by composing existing modules, or extended with new modules in a "plug and play" (open) manner. In this setting, the environment consists of the proof structure, represented as a labelled graph, built when trying to prove a conjecture. The agents, described as OMRSs, are situated on the nodes of this graph and act on its nodes and edges (e.g. they may add new nodes). Some experimental results based on a simple example are provided in order to illustrate the potentialities of our approach.

## Introduction

The goal of this paper is to present a model for systems of situated autonomous agents and to illustrate its application to (distributed) automated deduction. Ferber (Ferber 1994b) stresses the importance of modelling how the environment changes when agents are situated on it and when, more important, they interact (only) by means of the environment (as in the case of ants (Drogoul 1995) or robots picking up samples (Steels 1990)). It is also important to modelize the agents themselves as autonomous entities. Notice that the term "autonomous" is not only related to the autonomy of the goals of each agent (Ferber 1995) [1], but also of the interpretation of its sensory-motor experience (cf. (Müller & Rodriguez 1995)). Finally, while describing the environment and agent dynamics in an independent way, it is important to connect these two aspects introducing the interactions between the agents and the environment and between themselves. In particular, the individual and collective behaviours are essentially the emergent effect of these interactions. Therefore, for studying multiagent issues like cooperation, we have to describe precisely the nature of these interactions. This paper extends the work presented in (Müller & Pecchiari 1996) in this respect.

Most of the current models for multiagent systems are based on logical formalisms. An exception is the approach presented in (Demazeau 1993). Our model differentiates from this one by the sharp distinction between the point of view of the external observer, the point of view of the agent (this is a consequence of the hypothesis of autonomy mentioned above), and the point of view of the designer who is able to describe the relationship between these two previous points of view. This distinction allows us to describe appropriately the difference between a cooperation from an external point of view and a cooperation that is explicit for the agent (cf. (Ferber 1994a)). In this paper we will focus on an external account of cooperation. Namely we will say that a set of agents are cooperating if their interactions produce a qualitatively new or quantitatively better property. This paper demonstrates the gain of efficiency when using interacting agents on the case of automated deduction.

An important problem in the domain of automated deduction is the development of mechanisms for the interconnection and integration of different theorem provers. Currently, if we need a prover, there are only two choices: implement a new one from scratch, or adapt an existing prover to our needs, or more prag-

* Current address: IIIA - University of Neuchâtel

[1]When we say that an agent is an entity, we mean, among other things, "that is moved by a collection of tendencies (under the form of individual goals or a satisfaction function, as well as a survival function, that this entity tries to optimize)".

matically, adapt our needs to the capabilities of this prover. None of these options is satisfactory considering the current state-of-the-art technology for building mechanized reasoning systems.

The ultimate goal of the work presented in (Giunchiglia, Pecchiari, & Talcott 1994) is to be able to specify mechanized reasoning systems that are "open", i.e. systems that may be easily extended with new inference procedures in a "plug and play" manner or combined with other systems without the need to re-implement them. In order to realize this ultimate goal, (Giunchiglia, Pecchiari, & Talcott 1994) proposes to specify complex reasoning systems as *OMRSs* (Open Mechanized Reasoning Systems). An OMRS consists of three components: a *logical component*, formed by a *reasoning theory*, which corresponds in this framework to the classical notion of formal system and has been completely formalized in (Giunchiglia, Pecchiari, & Talcott 1994), a *control component* which consists of a set of inference strategies, and an *interaction component* which provides an OMRS with the capability of interacting with other systems, including OMRSs and human users. The development of the second and third components of the architecture is one of the long-term goals of the work presented in this paper (but cf. (Giunchiglia, Pecchiari, & Armando 1996) to see a specification of these two levels very different from the one presented in this paper). Towards this goal, we propose to apply our multiagent model considering each OMRS as an agent acting in a reactive way. This means that an agent does not communicate with or perceive explicitly other agents (this is to guarantee the openness of the architecture) and acts in an environment consisting in the proof being built. Notice that the importance of this application is twofold: (1) the application allows to show the applicability of our model to non trivial and abstract problems (completely different from the examples given before in the robotic domain); (2) multiagent systems provide a paradigm that is very appealing for the implementation of open reasoning systems.

In this paper, we describe a model for a multiagent system that aims to be general, based on the distinction among the observer/agent/designer points of view and consisting in the model of the environment (from the observer point of view), of the agent (from the agent point of view) and of the interactions inbetween (from the designer point of view). We present the formalization of cooperative theorem proving problems in this multiagent model and show some simple experimental results on the effectiveness of using multiagent systems for such tasks.

## Formalization

The formalization we propose has to take into account the distinction we want to make between the environment and the agents on one hand, and to carefully describe the interactions between the agents and the environment and among themselves on the other hand. Therefore we have to describe two models: 1) the *model of the environment* taking into account its topological structure that allows us to "situate" the agents and to represent the simultaneous actions of the agents seen as black boxes, 2) the *model of the agents* describing their decision process influenced by the perception and producing commands. Finally we have to describe the interactions between the agent and the environment relating the two models.

The resulting points of view are illustrated in figure 1 in which one can see the clear distinction between the state of the environment and the perception of the agent, as well as the distinction between the commands of the agent and the actual effects on the environment. This distinction parallels the influence-reaction distinction presented in (Ferber 1994b). The formalization will use the notation of control theory (Dean & Wellman 1991).

### Environment

In order to formalize the environment, we need to take into account its *state* $x$, which contains the agents (as they are situated and hence part of the environment), and its *dynamics* under the influence of the agent actions $\vec{u}$. $\vec{u}$ is the array of actions of each agent and allows to take into account simultaneous actions (Ferber 1994b). The dynamics is a function: $x_{t+1} = F(x_t, \vec{u}_t)$. We assume a dynamics which is both stationary, i.e. $F$ does not change over time and discrete, hopefully without loss of generality.

In control theory, this formulation is sufficient. In order to customize our framework to multiagent systems, we have to situate agents in the space. For this reason, we define a state $x$ as a set of positions ($L$), with a metrics ($M : L \times L \rightarrow R$), a set of objects ($O$) including the agents ($A \subset O$), and a function returning the positions on which objects are situated ($P : O \rightarrow L$). The topology, the set of objects and their positions change over time, hence we put indexes on the state and its components in order to make explicit this dependence over time: $x_t = \{L_t, M_t, O_t, P_t\}$.

This environment model is accessible from the *observer's point of view.* In particular, the observer may categorize the courses of actions performed by the agents into *behaviours.*

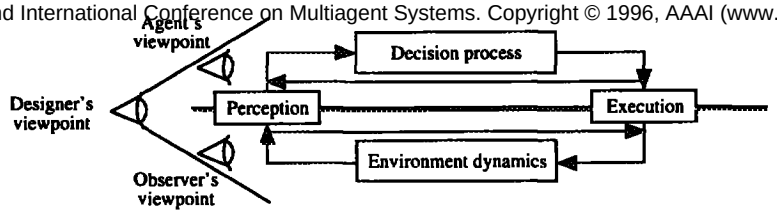


Figure 1: The different points of view

## Agents

An agent is characterized by its perception, commands and decision process. We have to distinguish clearly between the *environment state* ($x$) and the agent *perception* (indicated with $p \in P$ in the following). The agent perception is not only local, but it only consists of measures. For example, an agent may have an obstacle in front of him (from an observer point of view), but an agent equipped with a distance sensor is only able to get some measure and it may not know (contrarily to the observer) that this measure is correlated to its distance from an obstacle.

In the same way, one must distinguish between the *command* executed by the agent ($c \in C$) and the *action* that affects the environment ($u \in U$). For instance an agent may decide to change the speed of its motors or turn its wheels faster (the command), producing a movement in the environment (the action), i.e. a change of place. It is important to make this distinction because the correlation between them depends on the state of the ground and the wheels, or may even be unrelated (e.g. if the agent is pushed by another agent).

In the general case "immediate" perception is not enough to decide what command to execute and/or what the next perception will be (markovian hypothesis). As the markovian hypothesis is seldom satisfied, we introduce the notion of internal state $s \in S$ that is assumed to satisfy the markovian hypothesis and a *revision* process: $P \times S \rightarrow S$. Notice that the revision process of the internal state may vary from the identity function to the most sophisticated process of knowledge revision. This formalization spans the space between reactive agents where $S = P$ to cognitive ones.

The possible commands depend on the current internal state. Therefore we have for each internal state $s$, the set $C(s)$ of possible commands. We call *planification* the process for the construction of $C(s)$. Notice that the complexity of this process may vary from a very simple algorithm looking up data stored in a table (production rules mechanisms) up to a complex action planner.

The *decision process* is usually formalized as a function: $\pi : S \rightarrow C$. In our formalization, the function

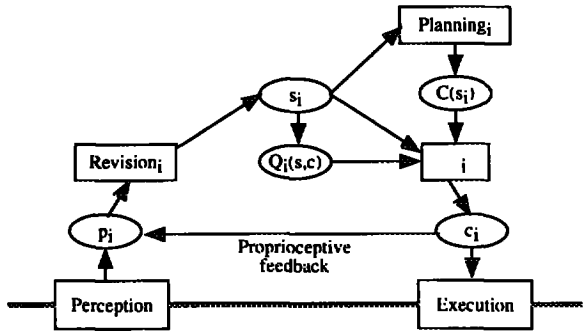


Figure 2: The model of the agents

$\pi$ is further decomposed into an evaluation function, $Q(s,c)$, that gives the long term utility of performing $c$ when the internal state of the agent is $s$. Notice that $Q$ may be given or calculated by different decision theories. Finally, the existence of multiagents (possibly heterogeneous) forces us to introduce indexes for each set defined above. The model of the agents so obtained is illustrated in figure 2 and formalizes the *sensory-motor loops* available from the *agent's point of view*.

## Interactions

Only the *designer* can actually describe both the environment dynamics (taking the place of the observer) and the agent dynamics (taking the place of the agent) but most importantly he is able to relate these two dynamics by means of interactions. Three types of interactions are coming up from the proposed model: the environment-agent interactions, the agent-environment interactions and the agent-agent interactions which occur as a composition of the previous interactions by the environment dynamics.

The environment-agent interaction implements the perception process. Perception can depend on the interaction of the agent with the environment. For this reason we propose to introduce a function depending on the environment state and the commands of the agent: $Perc : X \times C \rightarrow P$. This allows in particular to take into account active perception.

Regarding the agent-environment interaction, the distinction we have done between commands ($c \in C$)

and actions ($u \in U$) is instantiated by a function producing $u$ not only as a function of $c$, but also of the environment state. Therefore we have: $Exec : X \times C \rightarrow U$.

Finally, the agent-agent interactions can be deduced from the interactions mentioned above by means of the environment dynamics. In particular, the perception of an agent $i$ at time $t$ is a function of the state of the environment resulting from the effects of the commands of the other agents (including himself) at time $t-1$. If we isolate the contribution of a particular agent $j$, we get:

$$\begin{aligned} p_i(t) \;=\; & Perc( \\ & F(\; x(t-1), \\ & \quad (\ldots, Exec(x(t-1), c_j(t-1)), \ldots)), \\ & c_i(t-1)). \end{aligned}$$

It would be possible to introduce a specific notation for agent-agent interactions, but it is not necessary as the focus of this paper is on interaction through the environment.

## Application

As we mentioned in the introduction, an OMRS consists of a logic component (i.e. a reasoning theory), a control component, and an interaction component. We describe in the next subsection the first component [2]. The other components are then formalized using our agent's model.

### Logical Component of an OMRS

We start introducing the notion of reasoning theory and then show how a reasoning theory determines a set of reasoning structures (proof fragments occurring during the construction of a proof).

A *reasoning theory Rth* consists of a *sequent system* and a set of (inference) *rules*. A *sequent system Ssys* is a structure containing a set of *sequents*, $S$, assertions or judgements for consideration, and a set of *constraints*, $C$, that allow construction of provisional derivations (structures that are valid deductions when certain constraints are met). There is a constraint solving mechanism, $\models \subseteq \mathrm{P}_\omega(C) \times C$ (where $\mathrm{P}_\omega(C)$ is the set of finite subsets of $C$), represented abstractly as a consequence relation. Both sequents and constraints can be schematic and a sequent system also contains a set of *instantiation maps*, $I$, and an *application operation*, $\_[\_]$, for filling in schemata, that is $\_[\_] : [S \times I \rightarrow S]$ and $\_[\_] : [C \times I \rightarrow C]$.

A *rule* over a sequent system *Ssys* is a relation on tuples (called *instances*) consisting of a (possibly-empty) sequence of sequents (representing the premises or subgoals of the instance), a sequent (the conclusion or goal) and a finite set of constraints (the applicability conditions); that is, a rule is a subset of $S^* \times S \times \mathrm{P}_\omega(C)$. Furthermore, we pose the requirement that rules must be closed under instantiation.[3]

A *reasoning structure rs* is a directed labelled graph. The nodes of *rs* are partitioned into two (finite) sets: *sequent nodes* and *link nodes*. The edges of *rs* go from link nodes to sequent nodes or from sequent nodes to link nodes. Each link node has a unique incoming node (the *conclusion* or *goal*) and the outgoing nodes are ordered as a sequence (the *premises* or *subgoals*). Sequent nodes are labelled by sequents and link nodes are labelled by *justifications*. In this paper we consider only justifications for expressing rule applications (cf. (Giunchiglia, Pecchiari, & Talcott 1994) for other kinds of justifications), represented by a rule identifier and a set of constraints. We call link nodes with such kind of justifications *rule application links*. If *ln* is a rule application link of a reasoning structure *rs* we require that there exists an instance of the rule whose identifier is in the label of *ln*, such that its premises and conclusion are equal to corresponding sequents labelling the premises and conclusion of *ln*, and its constraints may be satisfied (under $\models$) assuming the constraints in the label of *ln*. For the sake of simplicity, in the following we do not consider rule application links with unsatisfied constraints. In this simplified framework, we may consider a reasoning structure as a graph whose nodes are sequent nodes and whose edges link premises of an inference step to its conclusion.

A reasoning structure is a *derivation*, of a conclusion sequent from a set of assumption sequents, if it represents a traditional proof figure.

### Multiagent Formalization

The multiagent formalization requires the definition of the environment, the agents and the interactions. We consider each OMRS as a reactive agent which contributes to develop the proof of the theorem submitted to the OMRS community. Therefore it is natural to define the environment as the reasoning structure being constructed. Intuitively, the sequents to manipulate are objects situated in the environment, i.e. in the sequent nodes of the reasoning structure considered, and the agents walk on sequent nodes choosing whether to produce new sequent nodes (and the corresponding se-

[2] The material presented in this subsection summarizes notions and results presented in (Giunchiglia, Pecchiari, & Talcott 1994).

[3] In (Giunchiglia, Pecchiari, & Talcott 1994) it is shown how particular rules, called *bridge rules*, may be used to interconnect different reasoning theories.

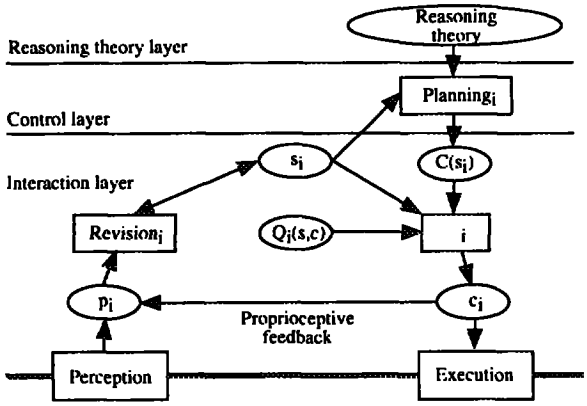


Figure 3: An OMRS seen as an agent

quents) or to continue to walk on other places.

**Environment** At each instant of time, the environment state $x_t = \{L_t, M_t, O_t, P_t\}$ is represented as a reasoning structure of the given reasoning theory. In this context, the set of positions $L$ corresponds to the set of sequent nodes. We define the distance between two nodes $d(l_i, l_j)$ as the minimal number of edges needed to reach $l_j$ from $l_i$ (but of course we could provide many other definitions for the distance between two nodes). The metrics $M$ is trivially defined from this notion of distance. The objects $O$ are sequents labelling nodes, for which the position does not change, and the OMRSs which move on the nodes. The function $P$ providing the position of each object is determined in each instant.

We have now to describe the array $\vec{u}$ of the actions performed by each agent. From the observer's point of view, an OMRS may: do nothing (but notice that its internal structure may change); move to other nodes (following a path determined by the edges of the reasoning structure); create new nodes or new edges (starting from the node where it is positioned).

**Agents** The general structure of an agent $a_i \in A$ described in the previous section may be naturally integrated with the three OMRS layers (see figure 3): the *reasoning theory component* provides the logical capabilities of the agent, i.e. the sequents and the rules that the agent is able to manipulate, the *control component* defines the inference strategies that the agent uses for deciding what the possible commands are and the *interaction component* provides the perception of the environment and decides which command to perform. W.r.t. the last component, we have to define the perception, internal state, commands, and decision process of an agent.

The perception $p \in P$ and internal state $s \in S$ of an agent depend on the problem to solve. In our case, the perception consists of a set of objects corresponding to sequents and fragments of the proof, possibly filtered according to the nature of the agent (i.e. according to the objects in its reasoning theory), and a measure related to the metrics defined on the environment (e.g. the distance of the agent from the objects perceived). Only the objects up to a certain distance are perceived. Finally, the internal state is the same as the perception.

The internal state and control component of the agent define the set of possible commands $C(s)$: the commands whose effect is to modify the perceived objects, and commands for moving towards a certain perceived object when the relative distance is greater than 0. The agent associates to each possible command a number $Q(s, c)$ that depends on the utility of the command in the current context. In general, if we try to obtain a certain quantitative performance, e.g. we try to minimize the length of the proof or to produce the less complicate proof, the evaluation function has to take into account these criteria according to the local perspective of the agent (as it is mentioned in (Ghedira 1995) about the distributed optimization and in the comments in the conclusion).

Finally, the agent has to choose a command on the basis of this evaluation. If the agent is strictly rational, it chooses one of the commands with the best evaluation. However, if each agent always executes the best command, the global result may not be effective. In such situations it is better to exploit the synergy of actions performed by all the agents (where the global result may be very good, but this effectiveness is not perceived locally by the agents). A possible solution is to choose a command with a probability proportional to its evaluation. In the following we use a fixed distribution.

**Interactions** Regarding the environment-agent interactions, notice that an agent does not directly perceive the modification of the proof structure, but only the effects on its perception (e.g. the modification of relative distances and the set of perceived sequents). We assume that an agent does not perceive the other agents but an agent may hide or modify sequents to the other agents. Regarding the agent-environment interactions, notice that the execution of a command produces the obvious effects on the environment. For instance the commands modifying the objects possibly introduce new nodes or new arcs in the reasoning structure. The conflicts arising from simultaneous actions are solved by the environment and not by some form of negotiation. For example, if two agents are moving simultaneously to the same place, the environment determines which of them will actually move.

Agent-agent interactions occur naturally from the composition of environment-agent interactions and agent-environment interactions.

## Some Experimentations

In this section we consider the theoretical model proposed by Colmerauer in (Colmerauer 1982). The goal of this model is to provide a formal account of the logic underlying Prolog II interpreters when they do not perform the occur-check test. We have chosen this example because it allows us to represent in a natural and open way the combination of agents performing different forms of reasoning. In particular, it is easy to combine equational reasoning (and inequations) with resolution reasoning as in Prolog II, and to also add treatment of numerical inequations and boolean equations as in Prolog III (Colmerauer 1990).

In order to present the structure of an OMRS, it is sufficient to specify the internal structure of an agent, i.e. its reasoning theory and its reasoning strategy. We conclude this section applying our multiagent formalization to a simple generate and test example.

### Prolog II as a Reasoning Theory

Before introducing the reasoning theory for Prolog II, $Rth_{\mathrm{pr}}$, we need to introduce some technical preliminaries. In the following we assume that $F$ is a set of function symbols and $V$ is an (enumerable) set of variables, and we define the set of terms $T$ from $F$ and $V$ in the usual way. A substitution *as* is a partial function from $V$ to $T$. An equation is a pair of terms $\langle s, t\rangle$ written $s = t$. We say that a substitution *as* is a solution of the equation $s = t$ if applying *as* to $s$ and $t$, we obtain the same term. This definition is naturally extended to a set of equations $\widetilde{eq}$.

Let us now define the sequent system $Ssys_{\mathrm{pr}}$. The sequents of $Ssys_{\mathrm{pr}}$, called r-sequents, have the form:

$$th \vdash_{\mathrm{r}} \langle \widetilde{t}, \widetilde{eq} \rangle,$$

where *th* is a set of objects $\langle t', \widetilde{t}' \rangle$ (representing the formula $(\bigwedge_{t \in \widetilde{t}'} t) \rightarrow t'$), and $\widetilde{t}$ and $\widetilde{eq}$ are respectively a finite set of terms and equations.

There is only one kind of constraint in $Ssys_{\mathrm{pr}}$, i.e. sol($\widetilde{eq}$) This constraint is satisfied when there is a solution of $\widetilde{eq}$.

We introduce below the $Rth_{\mathrm{pr}}$ rules. The rule *res* expresses in our setting the application of a resolution step:

$$\frac{th \vdash_{\mathrm{r}} \langle \{s_1, .., s_p, t_2, .., t_n\}, \widetilde{eq} \cup \{t_1 = s_0\}\rangle}{th \vdash_{\mathrm{r}} \langle \{t_1, .., t_n\}, \widetilde{eq} \rangle} \quad \langle res, \widetilde{c} \rangle$$

where $\widetilde{c}$ is the set of constraints $\{\mathrm{sol}(\widetilde{eq} \cup \{t_1 = s_0\})\}$ and $\langle s_0, \{s_1, \ldots, s_p\}\rangle$ is an element of *th* (to be precise, we should make explicit the fact that the variables $s_0, \ldots s_p$ have to be renamed to avoid name clashing). The rule *qed* is used to conclude the proof when there are no more terms in the current sequent and its equations are solvable:

$$\frac{}{th \vdash_{\mathrm{r}} \langle \emptyset, \widetilde{eq} \rangle} \quad \langle qed, \{\mathrm{sol}(\widetilde{eq})\}\rangle$$

Notice that these rules are presented in the classical forward form, i.e. from premises to conclusion. However, our notion of rule is adirectional and in the following this rules are applied in a backward way.

In the experiments showed below, we have introduced an *observing agent* which decides whether the proof attempt is terminated.

### A Simple Generate and Test Example

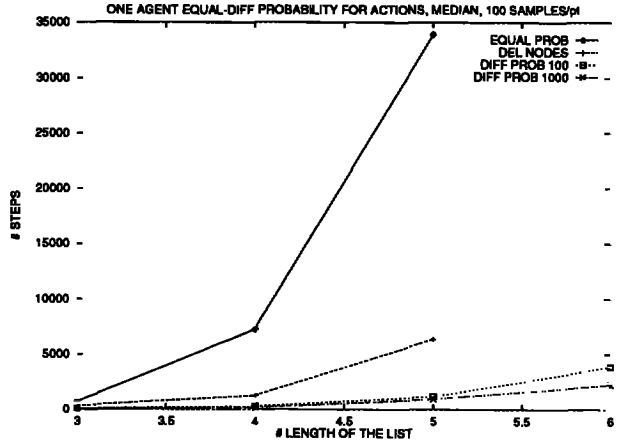


Figure 4: Experiments with one agent

We apply the above formalization to the classical problem of sorting the elements of a list (containing distinct natural numbers). We have to define an appropriate theory *th*. The elements of *th* are (using the syntax of Edimbourgh Prolog):

$$\langle select(X, [X|L], L), \emptyset\rangle$$

$$\langle select(X, L1, L2), \{select(X, [Y|L1], [Y|L2])\}\rangle$$

$$\langle perm(L1, [X|L2]), \{select(X, L1, L3), perm(L3, L2)\}\rangle$$

$$\langle perm(nil, nil), \emptyset\rangle$$

$$\langle ordered([X]), \emptyset\rangle$$

$$\langle ordered([X|[Y|L]]), \{less(X, Y), ordered([Y|L])\}\rangle$$

$$\langle sort(L, L1), \{perm(L, L1), ordered(L1)\}\rangle$$

At the beginning, the environment will contain only a sequent node labelled by a sequent whose form is

$th \vdash_r \langle\{sorted(L, L1)\}, \emptyset\rangle$, where $L$ is a ground term representing a list and $L1$ is a variable.

Notice that *th* represents, when interpreted in a Prolog-like fashion, a generate and test algorithm that "randomly" generates permutations of the list and then checks whether they are ordered. When a list has length $n$, there are $n!$ permutations of this list to consider using this method. The combinatorial complexity of this formulation is therefore well-suited to test the performance of a multiagent system with different strategies.

## Experiments with a Single r-agent

We start with the simple case of a single r-agent. At each instant, the r-agent has two possibilities: move towards a perceived sequent or apply a rule (adding the corresponding edges and nodes). We assume that an agent has no internal state, so there are two things we may vary: the perception of the agent and the evaluation function $Q(s, c)$. We have measured the performance of the agent w.r.t. the number of executed commands performed to obtain a proof. We have chosen the following experiments.

1. The agent perceives the immediate neighbor sequents (the subgoals generated from the current sequent and the goal from which the current sequent was generated) and the sequent on which the agent is situated, and $Q(s, c)$ is the same for all $c$ (case EQUAL PROB).

2. The agent may see only nodes which are not completely expanded and $Q(s, c)$ is the same for all $c$ (case DEL NODES). This is obtained with marks added by the r-agent when he is situated on a node for which there are no more rules to apply and its descendants (subgoals) have been completely expanded. Notice that this means that the agent may distinguish its descendants from the other nodes, e.g. its parent node.

3. The agent may see only nodes not completely expanded and $Q(s, c)$ is $\alpha$ times greater for a node expansion (application of a rule) than for a movement, where $\alpha$ is 100 and 1000 (case DIFF PROB 100 and DIFF PROB 1000, respectively).

The results are reported in figure 4 where the $x$-axis corresponds to the length of the list and the $y$-axis corresponds to the median of the number of commands executed by the agent in 100 samples. Notice that the global performance is very bad in the EQUAL PROB case. For instance for a list of length 5, the median value is more than 30000. A simple modification of the agent perception produces a very relevant effect (case DEL NODES). Finally, it is interesting to observe that, when $\alpha$ tends to infinite, the performance tends to be the same of a PROLOG-like depth-first approach (case DIFF PROB 100 and DIFF PROB 1000).

## Experiments with Multi r-agents

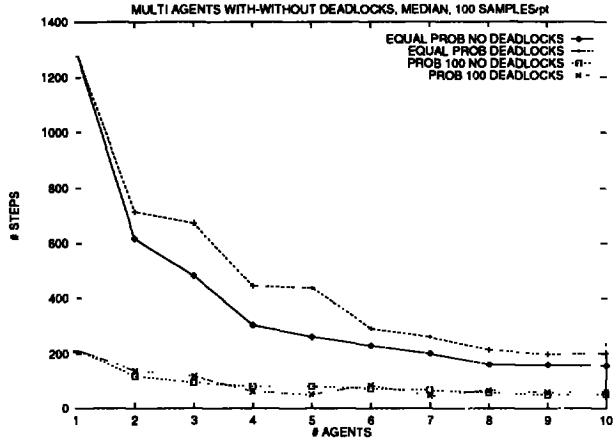


Figure 5: Experiments with multiagents

Let us now increase the number of agents and see what happens. We want to study the total number of actions, that is the total effort of a multiagent system according to the number of agents. Looking at the first experiments we have observed that there are agents which are blocked (situation with DEADLOCK). An agent is blocked when there are no rules to apply to the current sequent and all the neighborhoods are occupied by other agents. Therefore we have considered the following situations:

1. all the actions have equal priority with blocking (case EQUAL PROB DEADLOCKS);

2. all the actions have equal priority with correction of blocking (case EQUAL PROB NO DEADLOCKS);

3. the actions for adding new nodes have an evaluation 100 times greater that the others, with correction of blockings (case PROB 100 NO DEADLOCKS) and without correction of blocking (case PROB 100 DEADLOCKS).

The correction of blocking is obtained as follows. When an agent is blocked it is put on the root node of the proof.

The results are shown in figure 5. The number of steps takes into account the possibility of simultaneous actions. Therefore the number of steps corresponds to the number of instants in the formula:

| Agents | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Median | 216.5 | 117.5 | 94.0 | 80.5 | 80.5 | 72.0 | 66.5 | 57.5 | 50.0 | 52.0 |

Figure 6: Number of steps with multiagents

$x_{t+1} = F(x_t, \vec{u_t})$. Notice that in general the performance improves as the number of agents increases. Intuitively, this happens because occupied nodes are invisible and so there is a natural tendency to divide the work among the agents.

Figure 5 shows that the correction of blockages improves quite a lot the global performance. Nevertheless this is not sufficient to reach the performance of the strategy privileging node expansions. Finally, notice that table 6 shows that we do not obtain the optimal performance, i.e. division of number of steps for the number of agents, even in case case PROB 100 NO DEADLOCKS.

## Conclusions and future work

In this paper we have presented a model for systems of situated autonomous agents and we have showed the importance of distinguishing between the agents' point of view and the observer's point of view, of formalizing the interaction between the agents and the environment and, of situating the agents in the space in order to enrich the possibilities of interaction and therefore of cooperation. The first two points have been used to formalize properly a multiagent system for automated deduction. The third point has been used to situate agents in the proof. This allowed us to obtain a substantial improvement of performance by augmenting the number of agents without changing the implementation of the agents. This model opens up a lot of possibilities for the study of interactions and for discovering the limits we may go without an explicit communication between agents.

The link between OMRS and multiagent systems really seems appealing towards the definition of open reasoning systems. Actually, our approach allows flexibility in many directions: in the strategy utilized, in the number of agents, in the kind of agents, and in the problems considered. In our simple case, we have started exploring the first two kinds of flexibility. The other points are left for future work.

## References

Colmerauer, A. 1982. Prolog and infinite trees. In Clark, K., and Tärnlund, S.-A., eds., *Logic Programming*, number 16 in A.P.I.C Studies in Data Processing. Academic Press. 231–251.

Colmerauer, A. 1990. An introduction to prolog iii. *Communications of the ACM* 33(7):69–90.

Dean, and Wellman. 1991. *Planning and Control.* Morgan Kaufmann.

Demazeau, Y. 1993. La plate-forme paco et ses applications. In *2ème journées francophones sur l'Intelligence Artificielle distribuée et les Systèmes Multi-Agents.* Montpellier: PRC-IA.

Drogoul, A. 1995. When ants play chess. In Castelfranchi, C., and Müller, J., eds., *From Reaction to Cognition*, number 957 in LNAI. Springer Verlag.

Ferber, J. 1994a. Coopération réactive et émergence. *Intellectica* 19(2):19–52.

Ferber, J. 1994b. Un modèle de l'action pour les systèmes multi-agents. In *2ème Journées francophones IAD et SMA.*

Ferber, J. 1995. *Les systèmes multi-agents.* InterEditions.

Ghedira, K. 1995. Eco-optimisation combinatoire: fondements et exemples. In *3ème Journées francophones IAD et SMA.*

Giunchiglia, F.; Pecchiari, P.; and Armando, A. 1996. Towards provably correct system synthesis and extension. To be published in Journal of Future Generation Computer Systems.

Giunchiglia, F.; Pecchiari, P.; and Talcott, C. 1994. Reasoning Theories: Towards an Architecture for Open Mechanized Reasoning Systems. Technical Report STAN-CS-TN-94-15, Stanford Computer Science Department.

Müller, J., and Pecchiari, P. 1996. Un modèle pour des agents autonomes situés: une application à la déduction automatique. In J.P.Müller, and J.Quinqueton., eds., *JFIADSMA'96.* Hermes.

Müller, J., and Rodriguez, M. 1995. Representation and planning for behavior-based robots. In H.Bunke, T., ed., *Environment Modelling and Motion Planning for Autonomous Robots.* World scientific Pub.

Steels, L. 1990. Cooperation between distributed agents through self-organization. In Demazeau, Y., and Müller, J., eds., *Decentralized AI.* Elsevier.