Influences and Reaction : a Model of Situated Multiagent Systems

Jacques Ferber[1], Jean-Pierre Müller[2]
[1] LAFORIA - Université Paris 6, 4 place Jussieu, 75252 Paris cedex 05, France
ferber@laforia.ibp.fr
[2]IIIA - University of Neuchâtel, rue Emile Argand 11, 2007 Neuchâtel, Switzerland
muller@info.unine.ch

Abstract

This paper presents a general theory of action in multiagent systems which rely on a clear distinction between influences, which are produced by agents' behaviour, and the reaction of the environment. This theory allows us to rigorously describe complex interactions between situated agents. Two kinds of interactions are investigated: interactions between an agent and its environment, and interactions between agents through the environment. In this respect, we will show that the second kind is a special case of the first one. Although many theories about action are based on mental issues, we rely mainly on "what happens" in the world, i.e. the transformations that take place during interaction. Therefore, as shown in this paper, this theory is general enough to deal with both tropistic (reactive agents) and hysteretic (agents with memory) agents as special cases of general multiagent systems.

Introduction

As Jennings and Wooldridge have pointed out (Wooldridge & Jennings 1995), agent theories are essentially based on specifications about what properties agents should have and not on the definition of their behaviour from a more computational point of view.

A very popular approach in multi-agent systems is to use a knowledge level theory based on mental states such as beliefs, desires and intentions, to be used as specifications of what knowledge based agents should be. But these theories suffer from a major drawback which is related to the specification issue. The way these theories could be derived into tractable and implementable systems is not clear. Even if attempts have been made to describe implementable systems based on mental states (see for instance AGENT-0 (Shoham 1993), they are not formally related to the specifications. Wooldridge (Wooldridge 1996) on one hand and Rao (Rao 1996) on the other hand have tried to solve this issue using a synthesis process for the former (which is still a proposition at this time) and a dual logical/algebraic semantics (which makes the whole theoretical framework particularly complex) for the latter.

One of the major difficulties of these theories is their inability to take into account "reactive agents", i.e. agents that are not characterized by mental states but by their perception and action capabilities with respect to the environment. This is due to the fact that action per se is not represented: an action is seen only through the events that result from it. Thus, these theories cannot deal with simultaneous actions when they interact.

In order to represent actions in a multiagent system, a new trend in DAI is to use a variant of the situation calculus (Lespérance *et al.* 1996; Soutchansky & Ternovskaia 1995). In the situation calculus, the world is assumed to be in a certain state which changes only as a result of an agent performing an action. The state is reified in predicates and function terms. For instance the formula $on(C_1, C_2, s)$ means that the cube C_1 is on C_2 in state s.

There are two problems with the situation calculus. The first one is due to the well known frame problem (McCarthy & Hayes 1979). When describing an action, one has to specify also what remains unchanged. This limitation can be overcome by a special treatment (for instance see (Reiter 1991)). An action is split into two parts : a *precondition axiom* which states the conditions under which an action can be performed, and an *effect axiom* which expresses how an action affect the world's state. A solution to the frame problem is to automatically define a successor state axiom for each proposition changing its truth value over time in order to describe what effectively changes when an action is performed.

The other problem comes from concurrency. Basic situation calculus assumes that actions are performed sequentially. A concurrent extension has been proposed on different basis in (Weber 1990; Lespérance *et al.* 1996) which allows for interleaving actions, i.e. con-

current actions which do not interfere with each other. Still, this model suffers from the difficulty to handle simultaneous interacting actions. It cannot be applied to situated agents where actions can interfere in such a way that an action can prevent another action from being performed. For instance when a robot tries to push a cube from L_1 to L_2 and another robot tries to push the same cube from L_2 to L_1, actions cancel each other out. Moreover it is not clear how the situation calculus, even extended by concurrency, can express the dynamic laws of mechanics. Thus, the situation calculus is difficult to handle and suffers from limitations. We will show that our theory is able to solve these issues using a simple formalism.

The remainder of this paper is structured as follows. Firstly we present a general theory of action based on influences and reaction which allow for simultaneous interacting actions. Secondly we show how multiagent systems based on different kinds of agents may be described in this theory.

Action as Response to Influences

General Overview

We have seen that it is difficult to model both joint actions performed by several agents and the consequences of their actions. In particular, all the possible consequences of actions have to be described (this is called the ramification problem). The problem of these theories comes from the confusion between what an agent does and what is actually produced, the gesture and the result of the gesture (Ferber 1995).

For example, stretching out the arm (the gesture) can result in opening a door or not depending on the agents situation, the physics of the environment or even the gestures of other agents (for example, if another agent makes an opposite gesture on the other side of the door). But even stretching out the arm can be seen as a result of muscle contractions depending on external factors such as not carrying a heavy load or not having the arm tightly bound to the body. We call *influences* what come from inside the agents and are attempts to modify a course of events that would have taken place otherwise. *Reactions*, which result in state changes, are produced by combining influences of all agents, given the local state of the environment and the world laws.

Our theory is based on a different ontological basis from that of situation calculus. An action is divided into two phases (Ferber 1994): the first phase concerns influence production, and the second phase, the reaction of the environment to these influences given its previous state (see Figure 1).

By making as few hypotheses as possible on the

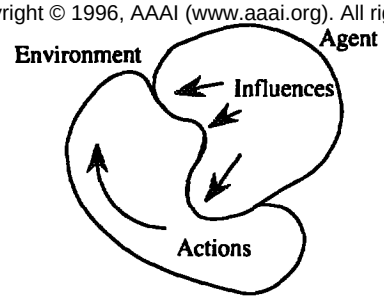

Figure 1: The influence-action model

structure of the states, we will be able to describe motions in a physical space as well as actions in other abstract spaces (Müller & Pecchiari 1996). Hence, this can be used as a general theory of action which integrate results from various disciplines such as classical mechanics, local propagation by cellular automata or even classical planning as particular cases. For instance, in classical mechanics, the state of the environment is given by the position and the speed of a set of bodies. Influences represent the different forces that act on bodies at a given time t. Applying the laws of the world will consist in summing all influences in order to deduce positions and speeds at time $t + \Delta t$.

This model is more general than the situation calculus, because it resolves into one of its variants when there is only one agent and there is a one to one correspondence between influences and results of actions. Our model of action relies on three main concepts:

1. A distinction between influences and reaction, in order to deal with simultaneous actions.

2. A decomposition of the whole system dynamics into two parts: the dynamics of the environment and the dynamics of the agents situated into the environment.

3. A description of the different dynamics by abstract state machines.

Dynamical States

In order to take into account both the state of the environment and the agents' influences, we introduce the notion of *dynamical state* (or *dstate*) $\delta \in \Delta$ as a pair $< \sigma, \gamma >$ where σ of Σ is the state of the environment and $\gamma \in \Gamma$ describes the influences.

The *state of the environment* σ of Σ can be represented by any structure. For the presentation we choose sets of logical formulas as in the STRIPS formalism, i.e. as a set of ground atomic formulas of the form $p(c_1, \ldots, c_n)$ where p is a predicate and a_i are constants (or functional terms without variables).

Whereas classical theories use only environment states, we introduce a new structure called an *influence set*, which represents the set of influences simultaneously produced by all agents and the environment. Each influence element γ can also be described by a set of atomic formulas which do not refer to the world state but to the dynamical component of the world as a set of tendencies for the environment to change. Γ is the set of all influence sets which can be described by such a language.

Actions and Reactions

The dynamics of the environment is simply a function mapping Δ into Δ. For demonstration purposes we split it into two functions: $Exec : \Sigma \times \Gamma \rightarrow \Gamma$ which produces the influences in the next dynamical state and: $React : \Sigma \times \Gamma \rightarrow \Sigma$ which produces the next environment state. Together they transform a dynamical state $\delta =< \sigma, \gamma >$ into a new dynamical state $\delta' =< \sigma', \gamma' >$ such that:

$$\sigma' = React(\sigma, \gamma) \; and \; \gamma' = Exec(\sigma', \gamma)$$

It is easy to see that, in classical theories, an action is a restriction of this model to the state transition: $\Sigma \rightarrow \Sigma$. This is the reason why it is necessary, in those theories, to describe complete transitions, whereas we only have to describe influences, which allow for a more economical description of simultaneous actions.

To describe the functions *Exec* and *React*, it is easier to introduce *operators* and *laws* respectively. An *operator* produces influences. We use the word *operator* to prevent any confusion with the concept of *action* that is usually defined as a function from world state to world state. An operator *op* of *Op* is simply a function from environment states to influence sets, and *Op* is the domain of such functions: $Op = \Sigma \rightarrow \Gamma$

Any language could be used to represent an operator. But for the sake of presentation we will use an operator language similar to STRIPS. An operator is then defined as a 3-tuple $< name, pre, post >$ where *name* is an expression of the form $f(x_1, \ldots, x_k)$ and where x_i are variables which can appear in both *pre* and *post* expressions: *pre* is a set of positive atomic formulas $p(a_1, \ldots, a_n)$ where p is a predicate of arity n and where a_i are either constants or variables; *post* is a set of influence terms. Notice that these operators, unlike STRIPS operators, do not transform a world state but produce influences. The function *Exec* is extended in order to take an operator as its argument:

$$\begin{array}{rccc} Exec: & Op \times \Sigma \times \Gamma & \rightarrow & \Gamma \\ & Exec(op, \sigma, \gamma) & \mapsto & \gamma' \end{array}$$

Exec is such that:

$$Exec \quad (< name, pre, post >, \sigma, \gamma) = \begin{cases} post & if \; Pre(\sigma) \; is \; verified \\ \{\} & otherwise \end{cases}$$

Operators can be composed to define the influence set resulting from simultaneously executed actions in a certain environment state. The parallel composition of operators, written $||$, simply produces the union of influences produced by the different operators. The function *Exec*, becomes a morphism from the space of operators into the set of influence sets Γ:

$$Exec : (Op, ||) \times \Sigma \times \Gamma \rightarrow \Sigma$$
$$Exec(a||b, \sigma, \gamma) = Exec(a, \sigma, \gamma) \cup Exec(b, \sigma, \gamma)$$

Similarly the laws of the world, called *laws* to simplify, describe how a next state is computed given a previous state and an influence set. Each law $\lambda \in Laws$ can be represented as a 4-tuple $< name, prestate, preinfluence, post >$ which describes how the next state is computed given the previous state and an influence set. We also extend the function *React* in order to cope with laws as follows:

$$\begin{array}{rccc} React: & Laws \times \Sigma \times \Gamma & \rightarrow & \Sigma \\ & React(\lambda, \sigma, \gamma) & \mapsto & \sigma' \end{array}$$

This function can be extended to the case of multiple laws in the following way:

$$React : (Law, ||) \times \Sigma \times \Gamma \rightarrow \Sigma$$
$$React(a||b, \sigma, \gamma) = React(b, React(a, \sigma, \gamma), \gamma)$$

The operator $||$ is a composition which must be commutative for laws to allow for a flexible description of state changes by an unordered set of laws.

We can define a *dynamical system* as a 6-tuple: $< \Sigma, \Gamma, Op, Laws, Exec, React >$ which allows for simultaneous interacting actions. At any time the *dynamical state* of the system is described as a pair $< \sigma \in \Sigma, \gamma \in \Gamma >$. Evolution of such a system can be defined as an infinite recursive function, called *Evolution*, which takes as its argument a state of the world but does not return any result due to its infinite loop (represented here as a function which returns a result in a domain containing only errors or impossible values, noted τ).

$$Evolution : \Sigma \times \Gamma \rightarrow \tau$$
$$Evolution(\sigma, \gamma) = Evolution(Cycle(\sigma, \gamma))$$

In a simple dynamical world without any agents, the cycle function can be defined as follows:

$$Cycle : \Sigma \times \Gamma \rightarrow \Sigma \times \Gamma$$
$$Cycle(\sigma, \gamma) =< \sigma', Exec(op_1 || \cdots || op_m, \sigma', \gamma) >$$

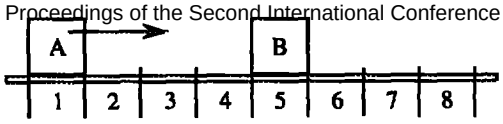

Figure 2: An initial situation (δ_1) of a cube world

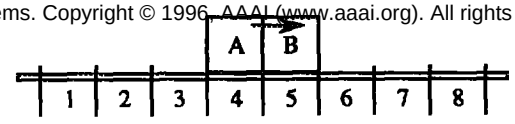

Figure 3: The environment state σ_2 of the cube world

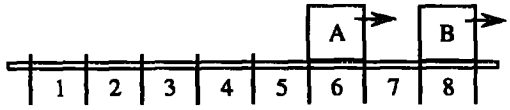

Figure 4: The environment state σ_3 of the cube world

where $\sigma' = React(\lambda_1 || \cdots || \lambda_n, \sigma, \gamma)$

Let us show the applicability of this formalism to model non-trivial physical phenomena and to account for simultaneous actions. For example the situation illustrated on Figure 2 can be described by the dynamical state δ_1:

$$< \{speed(A,4), speed(B,0), ontable(A), ontable(B), loc(A,1), loc(B,5), frictionidx(Table,1)\}, \{friction(A,Table,1)\} >$$

The influences are produced by the motion of A. The operators composing *Exec* are:

$$< produce friction(x,y), \{ontable(x), speed(x,v), v > 0, frictionidx(y,n)\}, \{friction(x,y,n)\} >$$

$$< push(x,y), \{ontable(x), speed(x,v), ontable(y), v > 0, loc(x,lx), loc(y,ly), |ly - lx| = 1\}, \{pressure(x,y,v)\} >$$

and the laws composing *React* are:

$$< reducespeed(x), \{speed(x,v), loc(x,l)\}, \{friction(x,z,n)\}, \{\neg speed(x,v), speed(x,v-n), \neg loc(x,l), loc(x,l+v-n)\} >$$

$$< producespeed(x), \{speed(y,v)\}, loc(y,l)\}, \{pressure(x,y,v)\}, \{\neg speed(y,v), \neg loc(y,l), loc(y,l+v+p), speed(y,v+p)\}$$

where $p = +/\{n | pressure(x,y,n)\}$ and / represents a reduction operator used to apply a binary operator to a set of values (ex: $+/\{1\ 2\ 3\ 4\} = 10$). It is easy to see that the cube A will slow down because of the friction and will come against the cube B. The environment state $s2$, will be (Figure 3):

$$\sigma_2 = \{ontable(A), loc(A,4), speed(A,3), ontable(B), loc(B,5), speed(B,0)\}$$

The cubes A and B are now in contact. Thus, the operator *push* will produce a pressure against the cube B which begins to move, while the cube A slows down even further. Because the *reducespeed* operator is only concerned by friction (a better representation would consider some kind of energy loss and the conservation of momentum, as in mechanics), the influence set γ_2 is equal to $\{friction(A,Table,1), pressure(A,B,3)\}$. The state δ_2, which is simply the pair $< \delta_2, \gamma_2 >$, is:

$$\gamma 2 =< \{ontable(A), loc(A,4), speed(A,3), ontable(B), loc(B,5), speed(B,0)\}, \{friction(A,Table,1), pressure(A,B,3)\} >$$

which will yield the following environment state σ_3 (Figure 5):

$$\sigma_3 = \{ontable(A), loc(A,6), speed(A,2), ontable(B), loc(B,8), speed(B,3)\}.$$

A Theory of Influence/Reaction for Multiagent Systems

It is now possible to extend the definition of a dynamical system in order to explicitely take agents into account. When action cannot be reduced to message passing, it is necessary to describe actions in their dynamics and not only by their results as in the situation calculus.

This theory of action based on influences and reactions gives a better understanding of multiagent systems and their implementation than previous formalizations because it distinguishes between agent behaviours and the consequences of those behaviours in the environment.

However, the above model, which is able to describe complex dynamics resulting from simultaneous interactions, does not make explicit the internal dynamics of the agents. Agents produce influences which can be described as their output to the environment and receives influences from the environment which can be described as their input. In order to go further we have the following choice: (1) to introduce new kinds of operators making explicit the computation which results into the modeled influences and state changes, (2) to

make the agents behave as complex environment states (and even dynamical states), the operators and laws describing at the same level the agent and the environment dynamics, and (3) to describe the agents separately from the environment dynamics but connected to it by computing at each step which influences they receive (perception) and which influences they produce given their situation in the environment.

The first solution is similar to the third but introduces notational complexity we want to avoid. The second solution makes the agent fully embedded in the environment. This can be very interesting if we are able to map high-level descriptions of agents into an equivalent dynamical system, but this direction has not been pursued in this paper. Finally, we will develop in this paper the third alternative in which the level of agent description is made independant from the formulation of the environment dynamics.

Models of agents and multiagents systems in an influence/reaction model

We are now able to elaborate how agent architectures and multiagent systems can be described in this model of action. Agents are entities which can perceive their environment and act accordingly, possibly deliberating about their actions. Genesereth and Nilsson (G&N for short) proposed a very simple architectural model of agents where actions are performed in a single agent world (Genesereth & Nilsson 1987). They have introduced a set of architectures ranging from pure reactive agents, called tropistic agents in their terminology, to higher level agents called hysteretic agents, i.e. agents with memory capabilities, and knowledge level agents. Their model has been used as a working basis for several formal theories about architectures and interactions. For instance Corrêa and Coelho (Corrêa & Coelho 1995) introduced conversational agents as extensions of knowledge level agents. In this section we will build on their architectural model and extend our theory of action to cope with both tropistic and hysteretic agents in a multiagent world. We will not present knowledge level agents, because, from an action point of view, they do not differ from hysteretic agents.

Perception is usually seen as a function which maps the set of world states Σ to a set of percepts P_a (in G&N theory, P_a is a partition of Σ) for an agent A: $Perception_a : \Sigma \rightarrow P_a$. This definition is related to a realistic point of view, in which we suppose that light and sound waves act directly on us, and that our internal processing is only based on a filtering process which classifies input situations. This philosophical idea which stems from Aristotle, and is still the basis of situation semantics (Barwise & Perry 1983), supposes that we are directly concerned by the whole state of the environment. While we could have used it to build tropistic and hysteretic agents as in G&N, we follow our principle of separation between influences and reaction and consider that agents are only influenced by the world. In this respect, the perception process is represented as a perception function which maps influences into a percept: $Perception_a : \Gamma \rightarrow P_a$.

This model includes automatically the locality of perception. Agents perceive what influences them and are not influenced by the whole state of the environment. Their reactions depend only on their internal state, which is the mark of their autonomy. This model could be seen as a formalization of the use of perturbations to describe system changes in the philosophical theory of Maturana and Varela (Maturana & Varela 1992). Computationaly speaking, we do not have to compute which part of the environment state is perceived by an agent. The environment computes it directly. Symetrically, agents produce influences in the environment in the same way operators do.

Deliberation is the process that takes place between perception and action and what differenciates tropistic agents, which are memoryless, from hysteretic agents, which have internal states and are able to take past situations into account.

Tropistic Agents

Agent Description Tropistic agents act directly on what they sense in the world. They do not have internal states nor explict goals. In this case, the behaviour of an agent is directly linked to its perceptions. Deliberation can be represented as a simple reflex function (that G&N call *action*): $Reflex_a : P_a \rightarrow \Gamma$

A tropistic agent, immersed into a dynamical system $< \Sigma, \Gamma, Op, Laws, Exec, React >$, can be defined as a 3-tuple: $a =< P_a, Perception_a, Reflex_a >$ i.e. as a set of perceptions, a perception function and a deliberation function which is limited to a pure reflex activity. Then the behaviour of an agent can be described as a mere production of influences, i.e. as a function which maps influences into influences:

$$Behaviour_a : \Gamma \rightarrow \Gamma$$
$$Behaviour_a(\gamma) = Reflex_a(Perception_a(\gamma))$$

A Multiagent Tropistic System A multiagent tropistic system (MATS for short), is defined as a set of tropistic agents which can *simultaneously* perform their action in a shared environment. A multiagent system can be described as an abstract machine which repetitively executes a *Cycle* function. Thus, a MATS can be formally described as a 7-tuple:

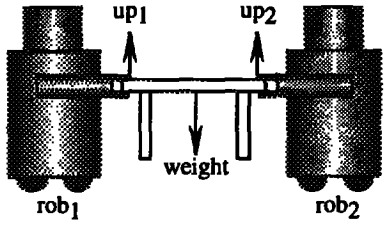

Figure 5: Lifting a table

$< A_t, \Sigma, \Gamma, Op, Laws, Exec, React, Cycle >$ where Σ, Γ, *Op*, *Laws*, *Exec* and *React* are defined as above, and where A_t is a set of tropistic agents.

In a tropistic system, the *Cycle* function gathers together all influences produced by the agents and the dynamics of the environment and lets the world reacts to them according to its reaction's laws. Simultaneously, the influences of the environment are passed to the tropistic agents to produce the agents' influences for the next step:

$$Cycle : \Sigma \times \Gamma \rightarrow \Sigma \times \Gamma$$
$$Cycle(\sigma, \gamma) =$$
$$< \sigma', Exec(Op_1 || \cdots || Op_m, \sigma', \gamma) \cup \bigcup_i Behaviour_i(\gamma) >$$

where $\sigma' = React(\lambda_1 || \cdots || \lambda_n, \sigma, \gamma)$.

An Example of Multiagent World with Tropistic Agents The Figure 5 illustrates our example of two agents lifting a table. In this case, the table will exert a torque on the robot hand depending on the angle of the table. This influence can be readily translated into a perception of the torque being zero (the table is horizontal), positive (the table is lower on his side) or negative (the table is higher on the robot's side): $P_a = \{zero, positive, negative\}$.

The influences generated by the robot are the possibility to exert a force upward (*up*) or to not exert any force (*quiet*).

To lift a table, we propose this simple decision function:

$$\begin{aligned} zero &\rightarrow up \\ positive &\rightarrow up \\ negative &\rightarrow quiet \end{aligned}$$

Firstly, we have to specify the laws of the environment in response to the robots influences. The first law expresses that the displacement of an object position y by the robot x is modified by the difference between the force applied upward (*up*) and the gravity going downward (*weight*). The second law describes that the respective height of the two extremities of the same object defines its tilt. Note that a law with an empty influence set expresses naturally inferences on the environment state:

$$\begin{aligned} < \; & displace(x,y), \\ & \{hold(x,y), height(y,h)\}, \\ & \{apply(x,up)\}, \\ & \{\neg height(y,h), height(y, h + (up + weight))\} > \end{aligned}$$

$$\begin{aligned} < \; & tilt(y), \\ & \{height(left(y), h_1), height(right(y), h_2), \\ & \; tilt(y, old))\}, \\ & \{\}, \\ & \{\neg tilt(y, old), \\ & \; tilt(y, arcsin((h_1 h_2)/d(left(y), right(y))))\} > \end{aligned}$$

The operators are limited to the production of a torque on each robot x depending on the position of the robot and proportional to the angle of the object y.

$$\begin{aligned} < \; & producelefttorque(y,x), \\ & \{hold(x, left(y)), tilt(y,a)\} \\ & \{torque(x, c * a)\} > \end{aligned}$$

$$\begin{aligned} < \; & producerighttorque(y,x), \\ & \{hold(x, right(y)), tilt(y,a)\} \\ & \{torque(x, c * a)\} > \end{aligned}$$

Several scenarii can be imagined for illustrating our model:

1. in a first case, the robot is alone at the left extremity of the table which initially is on the floor:

$$\begin{aligned} < \; & \{hold(rob_1, left(table)), tilt(table, 0), \\ & \; height(left(table), 10), \\ & \; height(right(table), 10)\}, \\ & \{torque(rob_1, 0)\} > \end{aligned}$$

   As the torque is null, the robot will produce the *up* influence ($apply(rob_1, up_1)$). Depending on the environment two outcomes can happen: (1) the table is too heavy and nothing happens, in which case the robot continues to try lifting; (2) the robot is able to lift the table ($up_1 > weight$), which results in the dynamical state:

$$\begin{aligned} < \; & \{hold(rob_1, left(table)), tilt(table, +a), \\ & \; height(left(table), 11), \\ & \; height(right(table), 10)\}, \\ & \{torque(rob_1, -(c * a))\} > \end{aligned}$$

   where the robot will wait for the torque to be zero again.

2. in a second case, we have a robot at each extremity of the table:

$$\begin{aligned} < \; & \{hold(rob_1, left(table)), height(left(table), 10), \\ & \; hold(rob_2, right(table)), tilt(table, 0), \\ & \; height(right(table), 10)\}, \\ & \{torque(rob_1, 0), torque(rob_2, 0)\} > \end{aligned}$$

Once again several outcomes can occur depending on the weight of the table and the strength of the two robots. If they exert the same force and the table is not too heavy ($up_1 = up_2 > weight$), the table will raise while remaining horizontal. If the forces are different, the weaker robot will wait for the other:

$$< \{hold(rob_1, left(table)), height(left(table), 12), hold(rob_2, right(table)), tilt(table, a), height(right(table), 11)\}, \{torque(rob_1, -(c * a)), torque(rob_2, (c * a))\} >$$

in which case rob_1 (with negative torque) will wait for rob_2 (or vice versa depending on the strength of the two robots).

This model shows that a behaviour is not encoded into the agent architecture but happens through interaction with the environment allowing sophisticated behaviours with simple agents. Our model particularly stresses this point.

Hysteretic Agents

Agent Description Hysteretic agents add internal states to tropistic agents. We will introduce a set of states S_a for each agent a. Memorizing information such as that of acquiring experience consists in transforming an internal state s of S_a into another internal state s' of S_a. Deliberation cannot be represented as one simple reflex function, but as two functions: one for decision and one for the memorization process. The memorization function takes two arguments, a percept and an internal state, and returns a new internal state: $Memorization_a : P_a \times S_a \rightarrow S_a$

The decision function takes an internal state as its argument and dictates which operation to execute: $Decision_a : S_a \rightarrow \Gamma$

Hysteretic agent immersed in a dynamical system $< \Sigma, \Gamma, Op.Laws, Exec, React >$ may be defined as a 5-tuple:
$< P_a, S_a, Perception_a, Memorization_a, Decision_a >$ where P_a and $Perception_a$ are defined as for tropistic agents above. Defining the behaviour of hysteretic agents is more complex than for tropistic agents, because it transforms internal states as well as producing influences. The behaviour function of an agent a can then be defined as follows:

$$Behaviour_a : S_a \times \Gamma \rightarrow S_a \times \Gamma$$
$$Behaviour_a(s, \sigma) =< s', Decision_a(s') >$$

where $s' = Memorization_a(Perception_a(\sigma), s)$.

Designing an agent consists in determining both the set of operations and the set of internal states and in describing the three functions $Perception_a$, $Decision_a$ and $Memorization_a$, such that the desired collective phenomena takes place.

A Multiagent Hysteretic System A multiagent hysteretic system (MAHS for short), is defined as for tropistic agents as a set of hysteretic agents which can simultaneously perform their action in a shared environment. A MAHS is defined as a 8-tuple: $< A_h, \Sigma, \Gamma, Op, Laws, Exec, React, Cycle >$ where Σ, Γ, Op, $Laws$, $Exec$ and $React$ are defined as above, and where A_h is a set of hysteretic agents. The set S of internal states, that represents the "mental state" of the multiagent system, is defined as a vector $< s_1, \ldots, s_n >$ of all agents $< a_1, \ldots, a_n >$ of A_h, i.e. as an element of the cartesian product $S_1 \times \cdots \times S_n$. Thus a MAHS state is composed of a MATS state and of a mental state, i.e. a 3-tuple of type $S \times \Sigma \times \Gamma$. The cycle function which describes the dynamics of the system, is defined as follows:

$$Cycle : S \times \Sigma \times \Gamma \rightarrow S \times \Sigma \times \Gamma$$
$$Cycle(< s_1 \times \cdots \times s_n >, \sigma, \gamma) =$$
$$< s'_1 \times \cdots \times s'_n, \sigma', Exec(Op_1 || \cdots || Op_m, \sigma', \gamma) \cup \bigcup_i \gamma_i >$$

where $\sigma' = React(\lambda_1 || \cdots || \lambda_n, \sigma, \gamma)$ and $< s'_i, \gamma_i >= Behaviour_{a_i}(s_i, \gamma)$.

In this formalization, the perception-deliberation-act cycle must be performed in one step. For a sophisticated hysteretic agent, this hypothesis looks like being unrealistic. There are several possible ways to solve this problem:

1. The first solution is to consider that an agent must act at each cycle, even when it has no time to fully integrate new perceptions into its internal representations. We could say that it cannot keep from acting. It is easy to modify our formalism by applying the decision on the current internal state (and not on the result of the memorization) allowing the memorization to occur concurrently on several steps: $Behaviour_a(s, \sigma) =< s', Decision_a(s) >$.

2. The second solution is to effectively consider that an agent takes several cycles to produce its influences. In this option, it is necessary to separate between the dynamics of the agent computations from the dynamics of the environment adding synchronisation points where influences are passed back and forth.

These solutions will not be elaborated further here for lack of place.

Conclusion

The theory presented here has been developed to deal with complex interactions both in the environment and between agents through the environment. We have seen that it is applicable for both tropistic (reactive agents) and hysteretic agents (agents with memory) in

the same formalism. This theory is based on a sophisticated state machine without commitment to the way state transitions are described as long as influences are separated from the reaction of the environment to these influences. In this paper, we are using a STRIPS-like description for states and transitions to cope with our conceptual requirements. But this theory is not restricted to this kind of language. We could have used any functional language to describe operators, laws and behaviours.

However, our approach is limited to synchronous descriptions of the multiagent systems evolutions. Extension to asynchronous models could come from the sophistication of the hysteretic agents. Such an extension would handle this problem in an elegant way. The perspective is to derive a complete multiagent abstract machine (MAAM) from the extensions of our formalism.

In our examples, we did not cope with knowledge level agents (i.e. with beliefs, desires and intentions). Two issues are of interest in this case, agent architectures and communications using speech acts. These issues are still to be further elaborated but in principle nothing interferes with the specification or implementation of our hysteretic agents in knowledge level terms. A mapping from knowledge level specifications into machine states and dynamics could be found in the line of (Rosenschein & Kaelbling 1986) and this line of research is being pursued. Finally, speech acts could be seen as influences either diffused around in natural environment or through communication links in communicating networks.

References

Barwise, J., and Perry, J. 1983. *Situations and Attitudes.* Situations and Attitudes.

Corrêa, M., and Coelho, H. 1995. Around the architectural agent approach to model conversations. In Castelfranchi, C., and Müller, J.-P., eds., *From Reaction to Cognition*, number 957 in LNAI. Springer Verlag. 172–185.

Ferber, J. 1994. Un modèle de l'action pour les systèmes multi-agents. In *Journées francophones sur les systèmes multi-agents et l'intelligence artificielle distribue.*

Ferber, J. 1995. *Les systèmes multi-agents: vers une intelligence collective.* InterEditions.

Genesereth, M., and Nilsson, N. 1987. *Logicial Foundations of Artificial Intelligence.* Morgan Kaufman.

Lespérance, Y.; Levesque, H.; Lin, F.; Marcu, D.; Reiter, R.; and Scherl, R. 1996. Foundations of a logical approach to agent programming. In Wooldridge, M.; Müller, J. P.; and Tambe, M., eds., *Intelligent Agents II*, number 1037 in LNAI. Springer Verlag. 331–346.

Maturana, H., and Varela, F. 1992. *The Tree of Knowledge, The Biological Roots of Human Understanding.* Scherz Verlag.

McCarthy, J., and Hayes, P. 1979. Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence* 4:463–502.

Müller, J.-P., and Pecchiari, P. 1996. Un modèle de systèmes d'agents autonomes situés: application à la déduction automatique. In Müller, J.-P., and Quinqueton, J., eds., *IA distribuée et systèmes multi-agents.* Hermès.

Rao, A. 1996. Agentspeak(l): Bdi agents speak out in a logical computable language. In de Velde, W. V., and Perram, J., eds., *Agents Breaking Away*, number 1038 in LNAI. Springer Verlag. 42–55.

Reiter, R. 1991. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In Lifschitz, V., ed., *AI and Math. Theory of Comp.: Papers in Honor of John McCarthy.* Academic Press. 359–380.

Rosenschein, S., and Kaelbling, K. 1986. The synthesis of digital machines with provable epistemic properties. In Halpern, J. Y., ed., *Conf. on Theoretical Aspects of Reasoning About Knowledge*, 83–98. Morgan Kaufman.

Shoham, Y. 1993. Agent-oriented programming. *Artificial Intelligence* 60(1):51–92.

Soutchansky, M., and Ternovskaia, E. 1995. Logical formalization of concurrent actions for multiagent systems. In Wooldridge, M., and Jennings, N., eds., *Intelligent Agents*, number 890 in LNAI. Springer Verlag. 129–144.

Weber, J. 1990. On the representation of concurrent actions in the situational calculus. In *8th Conf. of Canadian Society of Computat. Study of Int.*

Wooldridge, M., and Jennings, N. 1995. Agent theories, architectures and languages: A survey. In Wooldridge, M., and Jennings, N., eds., *Intelligent Agents*, number 890 in LNAI. Springer Verlag. 1–39.

Wooldridge, M. 1996. Time, knowledge, and choice. In Wooldridge, M.; Müller, J. P.; and Tambe, M., eds., *Intelligent Agents II*, number 1037 in LNAI. Springer Verlag. 79–96.