Sixth Annual Meeting of the European Chaos and Complexity in Organisations Network ECCON
Hotel Nassau, Bergen aan Zee, The Netherlands, 20-21 October 2006
Session Convenor: Dr. Frans M. van Eijnatten

Capturing Intentionality, Competence and Capabilities in Generic Agent Models using Boolean Rule Sets

Ton G.M. van Asseldonk & Ole Elstrup Rasmussen

TVA Developments, Veldhoven, The Netherlands,
Voice: +31 40 230 0100;Fax: +31 40 230 0200; E-mail: tva@tva.nl

Department of Psychology, University of Copenhagen, Copenhagen, Denmark,
Voice: +45 (353) 24810; Fax+45 (353) 24 802; E-mail: ole.elstrup@psy.ku.dk

---

Extended Abstract

Simulation techniques[1] are a powerful tool to study the nature and behaviour of complex interactive systems, whether they are weather systems, mechanical structures, etc, or systems that involve humans e.g. social systems, organizations. As interactions between agents presuppose the existence of events that can be observed by these agents and actions that can be executed by such agents, describing these agents in Boolean mathematics as sets of 'IF {event} THEN {action}' type of rules might be an adequate way to express agent behaviour in mathematical terms, in order to understand their micro behaviour, and simulate their behaviour in system models. However, there is a clear difference between a simple doorbell ("IF {button} THEN {ring}) and 'higher-level' agents (ultimately animals and humans) that can only be described in much more complicated rule sets.

This paper explores the possibilities of capturing this wide range of agents into one descriptive model. As a first step we prove that all possible event-action repertoires can be described in Boolean mathematics. Having done that, we develop the concept of generic agent classes by adding capability classes. The resulting five agent classes aim to cover the full range of possible agents from the simple doorbell up till and including the organisational behaviour of humans in interactive systems.

In the last section of the paper we aim to understand learning, individual as well as collective, in the context of these agent.

---

[1] Simulation techniques are becoming more and more important in studying the behaviour of interactive dynamic systems. In building understanding the often complex behavioural phenomena statistical methods, nor formal mathematical formulations prove, in many cases to be inadequate tools. Statistical methods fall short as the aggregated individual agents behaviour is not necessarily based on underlying grouping of agents and their characteristics, making (in such cases) notions as "mean", "standard-deviation", etc irrelevant expression. On the other hand (time dynamic, non-lineair) mathematics rapidly lead to a level of formula complexity that they are only useful in very stylized and simplified systems. Let, as in interactive systems, system behaviour is linked to, highly non-linear, interactions between system agents (Lorenz' butterfly causing hurricanes), such simplifications might well (and unknowingly) destroy the relevance of the simplified model for the more complicate real world.

Introduction and Scope

There are different ways to describe a system. A system being a collection of entities (called 'agents') of some kind, interacting with each other and with some type of environment. The environment encompasses everything that is not part of the system. For many of such systems one can describe the behaviour in e.g. mathematical terms: A function that relates the system output (that's what is 'given to' or can be observed by the environment) to the system input (that's what the system 'takes' or observes from the environment). Whereas such mathematical description of behaviour might prove to be highly practical, it *is* not the system, nor does it say much about how the system actually works. Even in case the agents comprising the system would have some form of consciousness whatever that may be, it is highly likely that they might not even be aware of the existence of such input/output relation of the system as a whole.

Let's consider a collection of birds flying in a flock. There are several mathematical descriptions that can turn a simulated collection of birds on a computer screen into a flock. Yet it is highly unlikely that birds know anything about flocks, or that they intent to create one. This brings us to another issue. Whereas we can define the flock as a system, we could also see the individual bird as a system, the brain of this bird, or the nerve-cell in this brain, etc. Describing systems in agent terms inevitably leads us into a seemingly endless nested structure, almost fractal, of systems and subsystems. At any level we could define agents and interactions. It is there-fore required to choose a level of observation. For this article we will consider systems to be networked agents at the level of 'things' or species e.g. birds, ants, computers, people, cars, etc. Anything below that level of granulation is considered to be part of the agent, not the agent itself.

Another way of describing system behaviour is to model the behaviour of the individual entities and their interaction. By simulating the system e.g. in a computer model where these agents interact, the system behaviour will emerge from this interaction, and the set of agent- and interaction characteristics can be seen as a description of the system. Yet again, it is very speculative whether such a description of the system *is* the system. If we were for example to describe the interaction and cooperation between ants in an ant-nest in their quest for food, we know that three basic rules describing agent behaviour will convincingly reproduce the observed behaviour. That doesn't mean that ant's 'think' according to the rules of the simulation. We just have no way of knowing that. The rules describe the system just like a mathematical formulation of the system function does.

However, such agent based descriptions might tell us a lot more about the working of the system than a mathematical formulation of the system function. Yet, as system behaviour is an emergent property of the interactions of the system, understanding: Why does the system what is does, in many cases might prove to be highly complex. And that's exactly the strength of agent based descriptions. It enables us to understand interactions between entities in a system and the behaviour of the system as a whole, even when this behaviour is highly non-linear, complex and dynamic. By understanding such systems we might be able to design purposeful systems that can be deployed to make useful contributions to its environment.

This article pursues the 'agent' component of such systems. It aims to classify agents in classes that have distinct properties with respect to the 'capabilities' of

such agents to relate input and output.

Rules and Rule-Based Systems

Above, we described agents as entities that can 'observe' some kind of input, and can 'create' some kind of output. For the scope of this article we will call the input 'event' and the output 'action'. This means that entities that can neither are not considered to be agents. But what about agents that can only do one of both (single side agents)? Agents that can observe events, but cannot create actions are clearly not of any use as they cannot contribute to the behaviour of the system. They might be there, but are of no interest. The other type: Output without input is an interesting one. Let's for the moment assume that such agent could exist. Clearly, it can affect the system behaviour by its actions, so it should be part of our considerations. The question arises whether such an agent is conceivable for any useful system (it could obviously be programmed). We will come back to this issue later on.

Let's for the moment limit our attention to agents that can both observe events and create actions. That also implies some relation between events and actions. In case this relation doesn't exist we have the equivalent of a single side agent described above. We can describe relations between events and actions in Boolean structures of the nature:

$$\text{IF } \{f[event_i]\} \text{ THEN } \{g[action_j]\}$$

In which f is a Boolean function of the set of possible events and g is a Boolean function of the set of possible actions. A large set of such relations could exist for a single agent. If we presume that event and action are binary and that an agent has just one input, only two such rules could exist:

Input: 0 -> output: 1 (if event is *not* there, the agent undertakes the action)
Input: 1 -> output: 1 (if event *is* there, the agent undertakes the action)

If the agent takes no action then the rule is meaningless for the system as such. With more inputs and outputs, the set of possible rules expands rapidly. Again, based on a binary input and output structure: with 2 inputs and outputs we have 4 separate input conditions, and 3 possible output conditions (1,1; 1,0, 0,1), which potentially gives us 12 rules. Three inputs and three outputs sums up to 8 x 7 = 56 possible rules, etc.

Now the question arises whether we can consider all inputs and outputs to be binary. This is obviously not necessarily the case for real systems. However, we can always describe non binary inputs and outputs in a binary way (e.g. 15 = 1111). Hence by increasing the number of inputs and outputs to 4 we can take care of all numerical inputs between 0 and 15. In other words, all non binary inputs and outputs can be expressed in Boolean terms, and hence the behaviour can be expressed in Boolean rule sets.

From the above, one important conclusions can be drawn: All agents defined in terms of observable events (inputs) and possible actions (outputs) can be described in Boolean terms. This implies that all interactive agent systems can potentially be described as a (Boolean) rule-based system, and all inputs and outputs as collections of binary events and actions. Hence by choosing this descriptive method no potential system is excluded.

Agent Classes

Having established an adequate tool for the description of agents, the question arises whether classes of agents that share certain characteristics in terms of the Boolean structure can be defined.

For the purpose of illustration we will use a rather peculiar version of a vacuum cleaner. As a starting point let's consider the type of automatic vacuum cleaners currently produced by the likes of Philips, Electrolux, etc. The machine moves itself around while sucking up dust on its path. For this movement, the machine has a set of wheels, an electromotor that drives the wheels, and (here we change the industrial product a bit) a mechanism that picks up batteries to power the electromotor, and some device to change direction of movement. For now, we use a very simple device: if the machine bumps into an object, it will move in a new, non-specific direction,

IF {obstacle} THEN {change direction}

and by that scan the surface and clean it. In order to sustain its existence as a working vacuum cleaner, the machine will require a new battery every once in a while.

Let's assume that the vacuum cleaner has the capability (action) to pick up batteries scattered in the machine's working area (input). Now and then the machine will bounce into one of the batteries while moving around. We could ad a rule like:

IF {battery} THEN {pick up}

This set of rules would constitute a viable agent, interacting with its environment. It would work and keep working under certain conditions with respect to its environment. The statistical chance of hitting a battery must be big enough to prevent exhaustion of energy, hence requiring a sufficient number of these batteries in the space. But when this condition is satisfied, our vacuum cleaner would not require any knowledge of its environment, nor of its own internal needs, to perform its function indefinitely.

We define this type of agent, with just inputs and outputs, as a Level-1 agent.

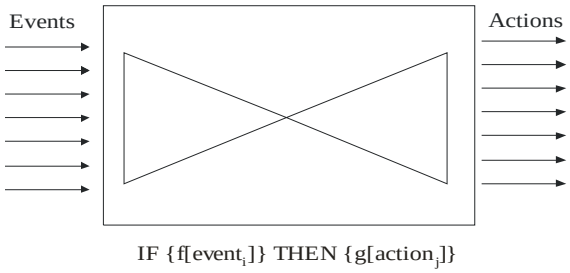

Figure 1: Level-1 Agent

The interesting thing is that already quite complicated systems can be modelled by means of such simple agents. The ant-nest referred to above, as well as models for birds flying in a flock can be adequately described using only Level-1 agents. E.g., if we allow the machine to store excess batteries and discharge them in front of inactive machines that our machine bumps into, we could model

entire ecological systems. We could for example model the oscillations of a fish stock in a lake given a specific amount of food. For our vacuum cleaners it would require the expansion of the 'Bump' rule with:

- a new rule: IF {bump AND object = stalled machine} THEN {offload spare battery}
- a device that can signal "Stalled" to the outside world (e.g. a defined sound)
- a sensor identifying the obstacle as a stalled machine (pick up the "stalled" sound
- a device that can dump the spare battery

The problem, however, with our LEVEL-1 machine is that it changes battery each time it hits a new battery, also when it has already a fully charged one. In our multi-machine setup this might be useful in order to provide batteries to 'stalled' colleagues, but if they all do that repeatedly, time and energy is wasted on changing good batteries.

To solve this problem we could add a timer. Let's assume that a battery lasts at least T_t minutes. This means that every battery it hits within this time could be neglected. So hitting a new battery after T_0+T_t (T_0 being the last battery change) causes a battery change. This implies the existence of some form of memory. The simple Level-1 agent doesn't have this capability. So let's define a Level-2 agent. This class of agent has all the capabilities of the Level-1 agent plus the capability of memorizing inputs and a notion of time, and with that a notion of history.

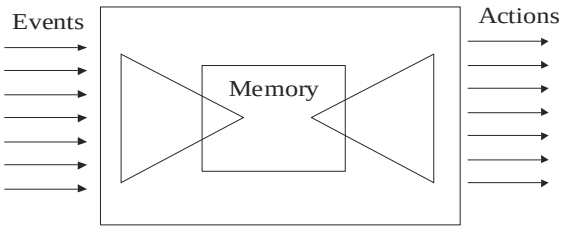

Figure 2: Level-2 Agent

The way to cope with this time delay in our Boolean notation is to add the time(shift) to the formula expression:

IF {f[event$_i$(t)]} THEN {g[(ection$_j$(t+ t)]}

With this new capability, we could e.g. add a rule like:

IF {new battery picked up} THEN {Pick up new battery in 1 hour}

The internal memory counts time-clicks (time is an external event) until 1 hour is passed and enables the pick up mechanism again. At any moment in time, the machine now knows how long it can drive before it needs a new battery by comparing T_t with the current time lapsed since battery change.

With that information, we can now enhance the probability of survival of the vacuum cleaner considerably by adding a sensory mechanism that can pick up the scent of environmental batteries. Normally the vacuum cleaner will change direction at certain intervals. This is, however, not particularly clever if the machine is in the vicinity of a battery. Thus, if the machine picks up the scent of a battery and it needs a new one, it should start steering towards the battery

detected. If more environmental batteries are available, it could steer towards the only one it can reach in time that is before his current battery passes out.

The sensory mechanism to do this is more complicated than the previous sensors with which our machine was fitted. A simple way of making such sensory system would be to have all new batteries in the environment fitted with a top light, radiating all around. If our vacuum cleaner has a rotating scanner on top that scans the environment for these lights it can work out relative direction (the angle of the scanner when it picks up the light) and the distance as the light intensity diminishes with the third power of distance. The rule to be added could then be something like:

IF {closest battery < 0.9 remaining distance}
THEN { Direction = angle(closest battery)}

leaving a 10% safety margin in distance[2].

Level-1 and Level-2 agents are passive, in the sense that they just respond to the outside events. They are, however, active in moving around. And in moving around in order to get energy from the environment, the machines already display intentionality, but they do not display a specific purpose, aspiration, or intention. They are just there to serve the environment. We just assume that all the ingredients to perform their function are there. In the case of our vacuum cleaner the availability of sufficient new batteries scattered over the space.

So let's make another step in sophistication. If we add an environmental map, remembering the inputs form or rotating scanner, the machine knows where the batteries are in general, and it also remembers what areas it has cleaned already. We could then have it, by some more or less clever algorithm which path through the space would create maximum cleaning for minimum time (or, equivalent, energy). The interesting thing is that now the machine will change direction, not in response to an external event, but as a result of an aspired performance. And if the environment changes e.g. batteries are appearing and disappearing randomly at a certain pace it can take action, preceding the 'need new battery event'. Rather than a notion of a past, this capability rests on a notion of a future.

So let's define another class of agents, which we will call Level-3 agents. What is required for such agents is not memory (representing the past) but a notion of future (knowing that it will die in the future if it doesn't recharge). The new element isn't just that notion of future. In other words it needs the ability to 'anticipate', and hence requires some form of model of the environment.
How could this 'anticipation' be treated in our Boolean system? If we were to treat the recharge as an event, which it is, then the Boolean taken the expression:

IF { f[$Event_i(t)$]} THEN {g[($Action_j$(t- t, t+ t)]}

expresses the fact that the event comes after the action. Mathematically this is a

---

[2] An interesting question now arises. If we were to see these machines act, it as if the display intent when they decide to head for the new battery. However, the intend is not a property of the machine, its merely the programmers hand that has put this necessity in place. The machine is hence executing an intention that is external to it. Phenomenologically however it will be difficult to impossible to distinct this from a situation in which the intend is internal to the machine (leaving aside what mechanism would yield this intent). At this level of agent, the machine does not have the capabilities yet to behave intentionally, and some other capabilities are required in order to achieve this. Below we will discuss intentionality extensively below.

nice mirroring of our expression for Level-2 agents.

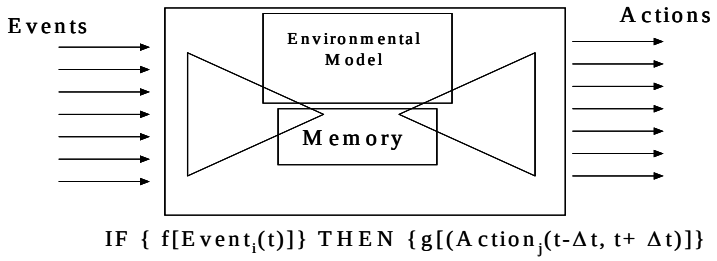

Figure 3: Level-3 agent

This type of agent gets close to behaviour of living agents and systems at a fairly high level of development, reptiles for example. Flowers growing towards the sun, trees growing roots, foxes searching for rabbits, they all anticipate (if they didn't they would have died before the action could take place). This is not surprising. As life is a continuous export of entropy from the system to the environment, it requires a lasting flux of energy into the system. That energy has to be found in the environment, and finding it requires some form of model of that environment, however simple.

Yet, as in our vacuum cleaner the needs are cast by the hands of the designer we still cannot say that our vacuum cleaner expresses "intention. Apparently, intention requires yet something else. Apart from the ability to anticipate and some model of the environment to link event en action, intention suggests some form of autonomy of the agents. And clearly this autonomy is lacking in our, by now rather clever, vacuum cleaner.

Authors' Full Coordinates

Ton van Asseldonk
TVA Developments BV
Postbus 387
5500 AJ Veldhoven
The Netherlands
Voice: +31 40 230 01 00
Fax: +31 40 230 02 00
E-mail: tva@tva.nl
Internet: http://www.tva.nl

&

Ole Elstrup Rasmussen
University of Copenhagen
Department of Psychology
Njalsgade 88
DK-2300 Copenhagen S
Denmark
Voice: +45 353 24 810
Fax: +45 353 24 802
E-mail: ole.elstrup@psy.ku.dk

****