Towards modelling of reactive, goal-oriented and hybrid intelligent agents using P Systems

Petros Kefalas and Ioanna Stamatopoulou

Department of Computer Science, CITY College, Thessaloniki, Greece,
International Faculty of the University of Sheffield
{kefalas, istamatopoulou}@city.academic.gr

Abstract. Intelligent agents are classified into various types depending on whether they just react to the stimuli they perceive (reactive) or they develop plans to solve their own goals (proactive or goal-oriented). In practice, agents are a mixture of two layers since they perform reactive or proactive tasks depending on what is the most appropriate at a given time (hybrid agents). Bearing in mind the dynamic organisation of a multi-agent system consisting of any of the above types, it is only natural to consider Population P Systems as a suitable candidate for modelling. In this paper, we describe preliminary work done towards modelling of MAS which include all types of agents. An initial attempt is made to tackle certain issues that have to do with the objects and rules that define each agent operation. Alongside the alternative solutions, we present a concrete example to demonstrate our findings and raise discussions.

1 Introduction

Intelligent agents are robotic or software entities which can exhibit autonomous, reactive, proactive and social behaviour [14]. Agents perceive their environment, react immediately if it is necessary, update their beliefs, revise their strategies, prioritise their goals and develop plans to achieve them. Multi-agent systems (MAS) are built upon the social behaviour of individual agents that can communicate, collaborate and negotiate in order to achieve their goals. Naturally, MAS are highly interactive, highly parallel and highly dynamic (change of organisation, change of roles, change in configuration etc). These dynamics make MAS specification, modelling and implementation a challenging activity. In our area of interest, formal modelling is particularly attractive as it raises many issues that cannot be tackled in a straightforward manner and leave many open challenges.

Intelligent agent architectures can be broadly categorised into reactive, goal-oriented and hybrid. In reactive agents, intelligent behaviour can be achieved without explicit symbolic representations or explicit abstract reasoning but it is an emergent property (e.g. ant colonies). The agents operation is based around a hierarchy of behaviours which resemble if *situation then action* rules.

On the other hand, goal-oriented agents and their most known representative Belief-Desire-Intention (BDI) agents [8], are based on: (a) Beliefs, i.e. the information an agent has about the environment, which may be false; (b) Desires,

i.e. the things that the agent would like to see achieved; and (c) Intentions, i.e. the goals that the agent is committed to. In principle, a BDI agent perceives its environment and updates its beliefs. Based on the current state of affairs, it may revise its options and prioritise its goals. Having picked up a current goal, generates a sequence of actions that achieve the goal and executes this plan. Finally, in most cases it is necessary for the agents to exhibit both reactive and proactive behaviour, hence the hybrid model.

Practically, BDI agents are not as complicated as the underlying theory dictates [3]. Desires play a strategic role to problem solving and in highly specialised agents desires are shrink down to one, the general raison d'etre of the agent. Plans are not generated but are ready made, residing in a library of plans that are brought into the play according the current goal. The current goal is the intention that is picked up for deliberation; if it is directly executable the agent performs an action; if not, the current intention is replaced by a more analytical list of new intentions (the plan).

There have been several attempts to formally model individual intelligent agents as well as MAS structure and change. Most of them were based on state machines and their variations, but were primarily concerned with simple reactive agents [2, 4]. Though such methods are adequate for the representation of the internal state of an agent, problems arose when having to deal with the dynamics of the structure of a system consisting of multiple agents. As a result other attempts used new computing paradigms, such as membrane computing. Such methods can efficiently address the aforementioned limitation of state-based methods (Population P Systems for example are very flexible in representing the dynamics of a population's structure), but were primarily concerned with biologically inspired or biological agents exhibiting emergent behaviour [10]. Finally, previous work has demonstrated that we can combine the above in order to take advantage of the complementary characteristics of the aforementioned formal methods [12]. For a complete review of this work, in terms of rationale and results, the interested reader is referred to [5].

With this paper, we initialise an effort towards modelling of goal-oriented agents using a variation of Population P Systems [1]. The following sections describe the modelling toolkit that should be available in order to formally model MAS that consists of reactive, goal-oriented or hybrid agents. We discuss the proposal along side with a MAS case in order to clarify our claims. Finally, we reach an initial definition of a Population P System suitable for modelling any type of MAS.

2 A MAS Scenario including goal-oriented agents

Assume a disaster area with civilians injured who are incapable of helping themselves in between obstacles and ruins [9]. A number of agents (rescue units or RU) are equipped with the necessary first aid kit and could provide help to injured civilians, thus temporarily rescuing victims from immediate danger. They can then broadcast the exact coordinates to the agents in their neighbourhood

and continue their rescue mission. Another set of agents (ambulance vehicles or AV) are capable of approaching the temporarily rescued civilians and carry them to a more secure establishment (e.g. emergency room or ER). Of course various parameters play an important role in this rescue scenario, such as number of agents, the amount of supplies, the capacity of the ambulances, etc. Also, one should take into account possible failures of agents as well as non-trivial interaction and mode of communication.

The development of such MAS would normally involve two kind of agents. RU would be reactive agents which would function under certain rules obeying a strict hierarchy such as for example:

if there is an obstacle then avoid obstacle $\succ$
if injured civilian is detected then
provide first aid to victim and inform nearby agents about location $\succ$
if empty space then move randomly

Injured civilians could also be modelled as reactive agents. On the other hand, AV would be goal-oriented agents which need to form plans to satisfy their goals, i.e. having updated their beliefs on where the victims are located based on incoming information and develop a sequence of actions to pick up their victims. In reality, AV agents should also have a reactive layer on top, which will respond to immediate threats, such as:

if there is an obstacle then avoid obstacle $\succ$
if at ER then upload the injured civilians $\succ$
if load reached the maximum capacity then move towards the ER $\succ$
if injured civilian is detected and not at ER then pick up victim

The above reactive layer deals with the simple behaviours, apart from *moving towards the victim* behaviour which requires planning. This is the main difference from agent RU that searches the space randomly for locating injured civilian.

Therefore the requirements for modelling the above is summarised in the following:

- modelling of individual separate agents of various types is necessary;
- the agent models should be developed with non-trivial data structures and their accompanying operations;
- there must be a way to code the rules for behaviours within an agent, including the communication behaviour;
- it is essential to set up priorities on these behaviours for the agent to perform the desired overall task;
- describing the change in communication links is desirable according to some "neighbouring" criteria;
- modelling of agents roles, generation and destruction must be possible in order to model the dynamic configuration of the system;
- agents in a MAS could operate in parallel exhibiting an asynchronous behaviour;

Most of the above naturally lead to considering elements of Tissue P Systems [7], Populations P Systems [1] and P colonies [6], equipped with some new features that could make them more focused to the modelling of goal-oriented and hybrid agents.

3 Formal Modelling of MAS

3.1 Agents as Cells

Each agent can be directly mapped to a cell. Cells are arranged in a graph (population) rather than a hierarchical tree structure. Cells must have types each corresponding to a different role of each agent in the MAS. For instance, in the rescue scenario there are four types of cells, namely a rescue unit (RU), an ambulance vehicle (AV), the civilian victim (CV) and the emergency room (ER). The latter could be an agent if it has certain characteristics (e.g. it is a mobile emergency unit, it can communicate with other agents etc.) or can be modelled as a simple entity otherwise. Instances of all these cell types make a MAS configuration. The graph denotes the communication between cells, e.g. neighbouring RUs and AVs have a direct communication, as well as a CV with a RU or AV on the same spot, the AV when it reaches the ER, the ERs between them etc.

3.2 Data Structures and Objects

Objects within cells should be more than multisets of symbols. Practically, more sets and mathematical structures are needed, such as naturals, reals, atoms, n-tuples, sequences/lists, sets, etc. as well as all operations applied on these. In addition, objects should be partitioned in various subsets, in a kind of annotated values of attributes. The absolutely necessary subsets of objects required for a goal-oriented agent are: (a) a set of Beliefs, (b) a set of Goals and (d) a set of internal agent States, and (e) a set of incoming Messages. Thus, for instance:

$B = \{(victim\_at\ X\ Y), (er\_at\ X\ Y), \ldots\}$ where $X,\ Y \in N$,
$G = \{(pickup\_victim\ X\ Y),\ (move\_towards\ X\ Y),\ (leave\_victim\_at\_er)\ldots\}$,
$States = \{doing\_nothing,\ rescuing,\ moving\_to\_er, \ldots\}$,
$IncomingMessage = \{(found\_victim\_at\ X\ Y), \ldots\}$ etc.

These in turn would be used in the list of goals, queue of incoming messages etc. Depending on the problem, some more sets might be necessary, such as the current position and direction is space, the capacity of an AV, the current load, the current supplies, fuel, etc. In practical modelling, we would need some sort of notation that differentiates objects according to the set they belong, for example:

$B : (victim\_at\ 3\ 8),\ State : rescuing,\ Pos : (4\ 5)$,
$ListOfGoals :\ \langle(move\_towards\ 6\ 7), (leave\_victim\_at\_er), \ldots\rangle$,
$IncomingQueue : \langle(found\_victim\_at\ 6\ 7),\ (found\_victim\_at\ 9\ 1), \ldots\rangle$, etc.

3.3 Behaviours and Rewrite/Communication Rules

Reactive behaviours can be modelled as a set of transformation rules for a specific agent type. For example, the rule:

$avoid\_obtacle : (State : moving\_to\_er\ Obstacle : (X\ Y)\ Pos : (X_1\ Y_1)$
$Direction : D$ if $(next\_to\ X\ Y\ X_1\ Y_1) \rightarrow$
$State : moving\_to\_er\ Obstacle : (X\ Y)\ Pos : (X_1\ Y_1)\ Direction : D'$ where $(random\ D'))_{AV}$

The objects on the left hand side are consumed and replaced by the objects of the right hand side within a cell of type AV. Checks and new values are performed and produced through the guards following the *if* delimiter and operations following the *where* delimiter. All rules could be identified by a unique identifier at the far left, in this case *avoid_obtacle*.

Proactive behaviours, such as updating beliefs, adding goals to the list or executing primitive goals (actions) can be modelled in a similar way.

Similarly, there exist communication rules that are used to pass messages to cells that are linked through the graph structure. For example, the rule:

$send\_victim\_position :$
$(B : (victim\_at\ 4\ 2);\ incoming\_message : (found\_victim\_at\ 4\ 2),\ broadcast)_{RU}$

means that in the presence of an object $B : (victim\_at\ 4\ 2)$ inside a cell of type RU an object $incoming\_message : (found\_victim\_at\ 4\ 2)$ can be obtained by all neighbouring cells, that is, those connected through links. The receiving agent can put the incoming message in the queue of incoming messages through a simple transformation rule. Guards and new values may also be required in the form of the rules.

Communication rules could also be used for perceiving the environment. For instance, the rule:

$perceive\_obstacle :$
$Pos : (X\ Y);\ Obstacle : (X_1\ Y_1)$ if $(within\_range\ X\ Y\ X_1\ Y_1),\ perceive)_{AV}$

is the same as the above with the exception that object $Obstacle : (X_1\ Y_1)$ is obtained by the environment. The other way round, i.e. an object is can be expelled out to the environment through an *output* rule. The performatives *broadcast*, *perceive* and *output* are equivalent to the commonly used in membrane computing *in*, *enter* and *exit*.

3.4 Priorities of Behaviours

In order to achieve the correct overall agent behaviour, individual behaviours including communication should be ordered. A top level ordering should determine which type of task behaviour should be tried first and whether communication (either *broadcast* or *perceive* or *output*) should precede or follow task behaviours. For example, a possible ordering might be:

broadbast rules $\preceq$ *perceive* rules $\prec$ *reactive* rules $\prec$
proactive rules $\preceq$ *output* rules

which implies that incoming message and perceptions should fire first followed by reactive rules followed by proactive rules and sending out messages.

At a lower level ordering between rules within the same type must exist. For instance, a possible ordering for the reactive behaviour should be:

$$avoid\_obstcle \prec upload\_victims \prec move\_towards\_er \prec pick\_up\_victim$$

It should be noted at this point that imposing such priorities in types of rules, or rules themselves, in combination to the use of lists for objects may have a restrictive effect on the maximal parallelism of the P system, which however for MAS modelling purposes is acceptable.

3.5 Communication Links and Bond Making

The graph connecting the cells is by no means fixed. Depending on the problem, the notion of "neighbourhood" between cells can be defined. This will allow cells to communicate directly through the existing links. Establishment of links is governed by a bond making rule that states the preconditions which must be true. For example, a bond making rule between an AV and a RU can be:

$$connect\_neighboring\_agents : (AV,\ Pos : (X_{av}\ Y_{av});\ Pos : (X_{ru}\ Y_{ru}), RU)$$
$$\text{if } (neighbours\ X_{av}\ Y_{av}\ X_{ru}\ Y_{ru})$$

meaning that in the presence of neighbouring $Pos : (X_{av}\ Y_{av})$ and $Pos : (X_{av}\ Y_{av})$ inside two cells of type AV and RU respectively, a bond is created between the two cells.

3.6 Dynamic Structure and Cell Differentiation/Division/Death

It is often the case that new agents should appear into the system, perhaps some change role and eventually some will disappear. Such situations can be handled effectively by cell division, cell differentiation and cell death rules. For instance, a RU which runs out of fuel is removed from the system:

$$out\_of\_order : (fuel : 0)_{RU} \rightarrow \dagger$$

4 Main Proposal

Bearing in mind the above, we propose that MAS consisting of any type of agents can be modelled using a variation of Population P Systems with Active Membranes, which can be defined as the construct:
${}^{\alpha}\mathcal{P} = (V, \Phi, T, \gamma, \alpha, w_E, A_1, A_2, \ldots, A_n, R_b, R_s, O_b, O_r, O_p)$ where:

- V is a finite set of structures and symbols called objects and $V = Beliefs \cup Goals \cup States \cup Messages \ldots$;
- Φ is a set of default and user-defined operations on items of V;
- T is a finite alphabet of symbols, which define different types of agents;
- $\gamma = (\{1, 2, \ldots n\}, G)$, with $G \subseteq \{\{i, j\} \,|\, 1 \leq i \neq j \leq n\}$, is a finite undirected graph;

- α is a finite set of bond-making rules;
- $w_E \in V^*$ is a finite multi-set of objects initially assigned to the environment;
- $A_i = (w_i, t_i)$, for each $1 \leq i \leq n$, with $w_i \in V^*$ a finite multi-set of objects, and $t_i \in T$ the type of agent/cell i;
- R_b is a finite set of behavioural rules and $R_b = BroadCast \cup Perceive \cup Output \cup Reactive \cup Proactive$.
- R_s is a finite set of structural rules and $R_s = Differentiation \cup Division \cup Death$.
- O_b is a partial order over behaviours R_b
- O_r is a partial order over the set of *Reactive* behaviour rules
- O_p is a partial order over the set of *Proactive* behaviour rules

The form of the rules is not formally defined here but can be fairly directly implied by the examples stated above.

5 Conclusions and Open Issues

We have made an initial attempt to define a new variation of Population P Systems with Active Membranes, namely ${}^{\alpha}\mathcal{P}$, which is suitable for modelling multi-agent systems including all types of agents, such as reactive, goal-oriented and hybrid. This is the main difference from previous work, in which we only dealt with simple biological reactive agents. We demonstrated the need through an example of a disaster scenario in which agents are trying to rescue civilians injured. A more concrete and precise definition, including the theoretical BDI model as well as the detailed computation steps are in our immediate intentions. Before that, however, we need to identify the other practical issues raised by such modelling. The main question is whether the constructs of the ${}^{\alpha}\mathcal{P}$ are adequate to map a MAS (including BDI agents) or there is a need to extend it with new ones. This will lead us to the design and implementation of a tool that animates the models, along the lines of previous work done both textually [11] and visually [13].

References

[1] Bernardini, F. and Gheorghe, M. (2004). Population P Systems. *Journal of Universal Computer Science*, 10(5):509–539.

[2] Coakley, S. (2007). *Formal Software Architecture for Agent-Based Modelling in Biology*. PhD thesis, Dept. of Comp. Science, Univ. of Sheffield, UK.

[3] Georgeff, M. P. and Lansky, A. L. (1987). Reactive reasoning and planning. In *Proc. of the 6th Conference on Artificial Intelligence*, pages 677–682.

[4] Kefalas, P., Holcombe, M., Eleftherakis, G., and Gheorghe, M. (2003). A formal method for the development of agent-based systems. In Plekhanova, V., editor, *Intelligent Agent Software Engineering*, pages 68–98. Idea Publishing Group Co.

[5] Kefalas, P. and Stamatopoulou, I. (2010). Modelling of multi-agent systems: Experiences with membrane computing and future challenges. In *Applications of Membrane computing, Concurrency and Agent-based modelling in POPulation biology (AMCA-POP), Satellite event of the 11th Conference on Membrane Computing*. To appear.

[6] Kelemen, J., Kelemenova, A., and Paun, G. (2004). Preview of P colonies: A biochemically inspired computing model. In Pollack, J. B., Bedau, M., Husbands, P., Ikegami, T., and Watson, R. A., editors, *Proceedings of the 9th Intern. Conference on the Simulation and Synthesis of Living Systems (Alife IX)*, pages 82–86. MIT Press.

[7] Martin-Vide, C., Păun, G., Pazos, J., and Rodriguez-Paton, A. (2003). Tissue P systems. *Theoretical Computer Science*, 296:295–326.

[8] Rao, A. S. and Georgeff, M. P. (1991). Modeling rational agents within a BDI-architecture. In Allen, J., Fikes, R., and Sandewall, E., editors, *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91)*, pages 473–484. Morgan Kaufmann.

[9] Sakellariou, I., Kefalas, P., and Stamatopoulou, I. (2008). Enhancing NetLogo to simulate BDI communicating agents. In *Artificial Intelligence: Theories, Models and Applications, Proceedings of the 5th Hellenic Conference on AI (SETN'08)*, volume 5138 of *Lecture Notes in Computer Science*, pages 263–275. Springer.

[10] Stamatopoulou, I., Gheorghe, M., and Kefalas, P. (2005a). Modelling dynamic configuration of biology-inspired multi-agent systems with Communicating X-machines and Population P Systems. In *Membrane Computing: 5th International Workshop*, volume 3365 of *Lecture Notes in Computer Science*, pages 389–401. Springer-Verlag, Berlin.

[11] Stamatopoulou, I., Kefalas, P., Eleftherakis, G., and Gheorghe, M. (2005b). A modelling language and tool for Population P Systems. In *Proceedings of the 10th Panhellenic Conference in Informatics (PCI'05)*.

[12] Stamatopoulou, I., Sakellariou, I., Kefalas, P., and Eleftherakis, G. (2008). *OPERAS* for social insects: Formal modelling and prototype simulation. Special Issue of *Romanian Journal of Information Science and Technology (ROMJIST)* on *Natural Computing — from biology to computer science and back to applications*, 11(3):267–280.

[13] Wilensky, U. (1999). Netlogo. http://ccl.northwestern.edu/netlogo. Center for Connected Learning and Computer-based Modelling. Northwestern University, Evanston, IL.

[14] Wooldridge, M. and Jennings, N. R. (1995). Intelligent agents: Theory and practice. *Knowledge Engineering Review*, 10(2):115–152.