PROCEEDINGS OF THE WORKSHOP ON APPLICATIONS OF SOFTWARE AGENTS
ISBN 978-86-7031-188-6, pp. 37 - 44, 2011

A Framework towards the Verification of Emergent Properties in Spatial Multi-Agent Systems

Isidora Petreska[1], Petros Kefalas[2], and Marian Gheorghe[3]

[1] South East European Research Centre (SEERC),
24 Proxenou Koromila Str., Thessaloniki 54622, Greece,
ispetreska@seerc.org

[2] CITY College, International Faculty of the University of Sheffield,
2 Leontos Sofou Str., Thessaloniki 54626, Greece,
kefalas@city.academic.gr

[3] University of Sheffield, Dept. of Computer Science
Regent Court, 211 Portobello Str., Sheffield S1 4DP, UK
m.gheorghe@dcs.shef.ac.uk

Abstract. Formal modelling of multi-agent systems (MAS) present many interesting challenges. In this extended abstract we present a framework of how formal modelling can lead towards identification and verification of emergent properties of spatial biology-inspired MAS. We discuss the problem in question as well as initial work done on the formal modelling side and the visual animation of these formal models.

Key words: Biology-inspired MAS, formal modelling, emergence, visual animation

1 Introduction

Verification of the emergent behaviour of multi-agent systems is an extremely complex task. It is not only the fact that the verification process, formal or model checking, leads to combinatorial explosion, but also the fact that emergent properties should be identified first before there is an attempt to be verified. The latter is not always straightforward. It is therefore desirable to combine several formal with informal techniques that would be able to join forces towards the verification of MAS.

In agents that operate in a 2 or 3-dimensional space, such as biology or biology-inspired agents, emergence is characterised by a pattern appearing in the agents configuration at some instance during the operation of the system. Trivial examples are colonies of social insects, like ants, birds, fish etc. The type of emergence observed is related to the positioning in space, for example line formation, flocks, schools, herds etc. Modelling such agents would require modelling of their position and verification would require the exploration of a state space developed by the combination of all agent positions evolved through time.

Someone could apply formal verification techniques, such as model checking, under the assumption that we know what emergent property we are looking for. With biology agents this is known in advance, since it has been observed in-vivo. With artificial agents it is not as simple, there is however an active research that proposes a framework for empirical exploration of emergent formations [1]. Consider the following example, known as aggressor-defender game.

In the aggressor-defender game [2] there exist two teams of agents randomly distributed in an environment: defenders (refer to them as *friends*) and aggressors (or *enemies*). There are three different sub-games involved:

– All the agents defend – at each turn everyone tends to position between a friend and an enemy (such as they were defending the friend against the enemy), Fig. 1 a).
– All the agents flee – at each turn everyone tries to position in a way that a friend is between themselves and an enemy (such as the friend protects them from an enemy), Fig. 1 b).
– Some agents defend while the rest of the agents flee.

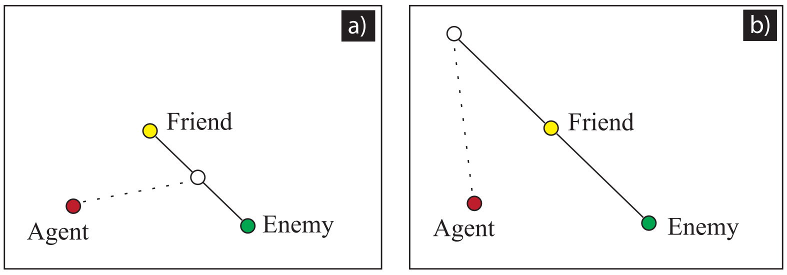

Fig. 1. Rules for playing the aggressor-defender game.

Assuming that we can develop a formal model to be used for model checking, it is interesting to consider what property to check for, that is, whether there is an emergent behaviour in all the above three cases, if this MAS is massively populated with similar agents.

This work aims to set up a framework of study concerning the above interesting problems and more in particular to demonstrate preliminary results in identifying emergent behaviour through the automatic transformation of a formal model to an executable visual simulation.

2 A Proposed Research Framework

The proposed research framework is depicted in Fig. 2. At the top, we start by formal modelling of agents. Such formal models should be able to clearly distinguish modelling of various types of behaviours, such as spatial or other behaviours, communication, dynamic organisation etc. By separating the various behaviours within the same formal model, it is possible to apply different transformations which will facilitate further processing. On one hand, the spatial behaviour determined by movement in space, can lead towards visual animation. The latter is a useful informal tool which will help observing potential emergent properties. On the other hand, suitable abstractions of spatial behaviour together with the rest of the behaviours can lead towards simulation and logging of time series data. These could be used to identify patterns of behaviours which combined with the visual animation produce a set of desired properties. Finally, the desired properties (including emergence) can be verified in the original spatial agent model by model checking, as long as there is a way to transform the original model into an equivalent, susceptible to formal verification, model.

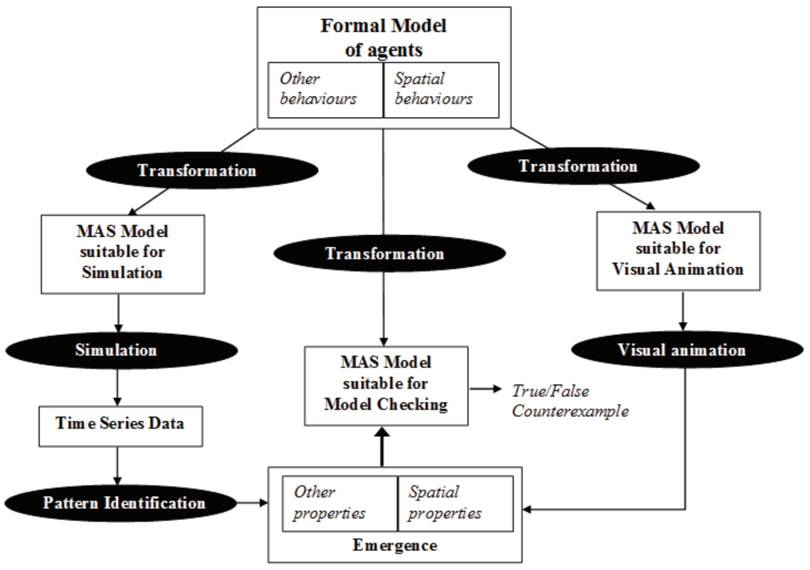

Fig. 2. A framework for validating emergent properties in spatial biology-inspired MAS.

3 Formal Modelling of Spatial Agents

We have been long experimenting with state-based modelling for agents and MAS [3–5]. The state-based modelling method we use is X-Machines (XM). XM are state machines with memory and instead of inputs triggering transitions, they trigger functions which label the transitions. XM are able to communicate through message exchange, thus forming Communicating X-Machines [6]. XM can also be wrapped around cells inspired by P-Systems [7] which are responsible for the dynamic configuration of the MAS. This idea was successfully introduced in the OPERAS framework (or more particularly OPERAS$_{XC}$) [8]. Although, XM can treat movement in space as any other behaviour, we have developed a number of arguments why this spatial behaviour must be separately modelled and treated [3, 5].

^{sp}XMs represent a variation of Stream XMs by defining additional components that allow specification of the current position and direction of an agent, as well as to formally specify a movement of an agent within its environment. Formally, a ^{sp}XM is a 13-tuple; ^{sp}XM = (Σ, Γ, Q, q_0, M, m_0, π, π_0, θ, θ_0, E, Φ, F) [3]), where:

- Σ is an input set of symbols,
- Γ is an output sets of symbols,
- Q is a finite set of states,
- q_0 is the initial state,
- M is an n-tuple called memory,
- m_0 is the initial memory,
- π is a tuple of the current position, i.e. (x, y) when a 2D representation is considered,
- π_0 is the initial position,
- θ is an integer in the range 0 to 360, that represents a direction,
- θ_0 is the initial direction,
- E is a set which contains elementary positioning operations: e_i such as e_i : $\Pi \times \Theta \longrightarrow \Pi \times \Theta$, such as direction, moving forward and moving to a specific position.
- Φ is a finite set of partial functions ϕ that map a memory state, position, direction and set of inputs to a new memory state, position, direction and set of outputs:
  ϕ: M $\times$ π $\times$ θ $\times$ Σ $\longrightarrow$ M $\times$ π $\times$ θ $\times$ Γ,
- F is a function that determines the next state, given a state and a function from the type Φ,
  F: Q $\times$ Φ $\rightarrow$Q, and

A ^{sp}XM model which demonstrates the third strategy is presented on Fig. 3. The model's states are Q={DEFENDING, STAYING_STILL, FLEEING}. There are three corresponding functions: to *defend*, to *stay* still and to *flee*. The memory stores the game strategy of the agent, an agent's friend and enemy, as well as its position and direction. The input consists of the friend's and the enemy's current position. Finally, the output is the new position of the agent, because every agent outputs its position to the other agents thus constructing a *communicating* ^{sp}*X-machine* system [6], [9].

^{sp}XMDL is the notation used to define ^{sp}XMs [3] and it is modified version of XMDL (see [10], [11]) used in the standard XM. The functions in ^{sp}XMDL are coded in the form:

```
 #fun functor (($input$), ($memory tuple$),
($position$), ($direction$)) =
      (($output$), ($memory tuple'$),
($position'$), ($direction'$)) =
      where
      $<list of operations including positioning>$
```

Considering Fig. 3, the function *defend* is:

```
#fun defend ( ((?x_fr, ?y_fr),(?x_en, ?y_en)),
   (?strategy, ?friend, ?enemy),
   (?my_xcor, ?my_ycor),
   (?curr_direction) ) =
   (("move to ?new_xcor ?new_ycor"),
   (?strategy, ?friend, ?enemy),
   (?new_xcor, ?new_ycor),
   (?curr_direction))
   where
   ?new_xcor <- (?x_fr + ?x_en)/2 and
   ?new_ycor <- (?y_fr + ?y_en)/2.
```

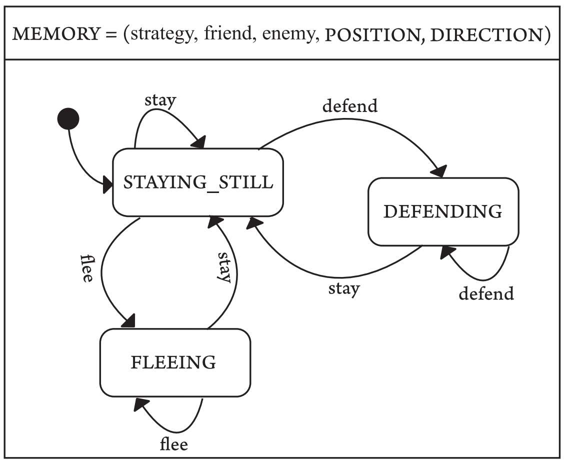

Fig. 3. ^{sp}XM model of the aggressor-defender game.

4 Visual Animation

As part of the first steps towards the achievement of formal verification of emergent properties, we have developed a tool for automatic translation of a ^{sp}XM model to NetLogo [3]. NetLogo is considered specialised into simulating natural and social phenomena, including modeling of complex systems [12], [13]. The platform supports hundreds of agents to operate independently, providing a clear picture of the micro-level behavior of the agents, as well as the macro-level patterns within the whole system. The translator is based on a set of mapping between formal constructs of XM and language primitives of NetLogo as well as a library supporting all the spatial behaviours.

For the aggressor-defender MAS, an executable counterpart is generated. The output shows a visual animation with which the emergent spatial behaviour is observable (Fig. 4), such as:

- The model in which all the agents defend, see Fig. 4 a), behaved as all the agents quickly collapsed into a tight knot,
- The model in which all the agents flee, see Fig. 4 b), behaved as a highly dynamic group that expands over time towards the ends of the environment, and
- The model in which the agents randomly choose whether to defend or to flee, see Fig. 4 c), exhibited there different behaviours. In some situations the agents were all collapsed into a tight knot (as the model from the defender game) with the difference that this knot was now oscillating around the environment (i), in others they were stationary, randomly distributed and oscillating (ii), and in the last case the agents would form a flocking (iii).

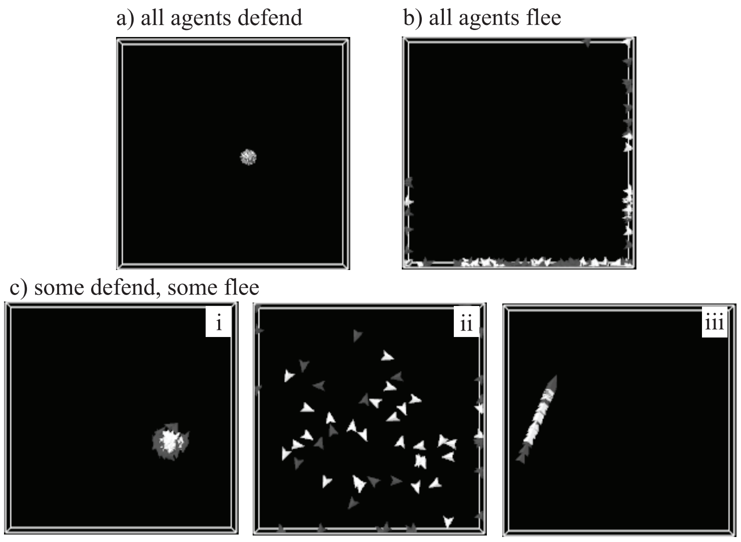

Fig. 4. NetLogo output of the aggressor-defender game.

The case of this game clearly demonstrated that visual animation aided in discovery of the system's emergence and properties that could be verified at a later stage, which in turn proved that even the small changes within the individual agent rules might cause a huge difference in behaviour of the system as a whole.

5 Discussion and Conclusions

The contribution of this paper is to present the overall picture of a framework towards the verification of emergent behaviour of spatial MAS. We have also reported progress so far, that is, a definition of ^{sp}XM and a tool for automatic transformation to NetLogo. Using this experience, the next steps in the framework are instantiated (Fig. 2) as follows:

- ^{sp}XM can be transformed into a simulation tool that can generate a time series data. Such tool may be FLAME [14, 15] which is used to animate XM models with thousands of agents. FLAME, however, does not deal with the spatial behaviour, which we have already covered by NetLogo.
- The logged time series data could be used as an input to a tool identifying patterns, such as DAIKON [16]. The output would be interesting properties that combined with the emergent properties from visual animation could aid us forming the logic temporal formulae to verify.
- The ^{sp}XM can be suitably transformed into an equivalent model in SPIN, PRISM or SMV [17–19], which given the temporal formulae will verify that all the desired properties hold in the original model.

Of course the above would assume that a correct transformation from the original model to equivalent models is possible, something which is an interesting problem by itself.

References

1. O.Paunovski, G.Eleftherakis, A.J.Cowling: Disciplined exploration of emergence using multi-agent simulation framework. Computing and Informatics **28**(3) (2009) 369–391
2. E.Bonabeau: Agent-based modeling: methods and techniques for simulating human systems. Proceedings of the National Academy of Sciences (2002) 7280–7287 Washington, United-States.
3. I.Petreska, P.Kefalas, I.Stamatopoulou: Extending x-machines to support representation of spatial agents. Work in progress (2011)
4. I.Petreska, P.Kefalas, M.Georghe: Population p systems with moving active cells. Twelfth International Conference on Membrane Computing (CMC12) (2011) In Print.
5. I.Petreska, P.Kefalas, M.Georghe: Informal verification by visualisation of state-based formal models of bio-agents. Proceedings of the 6th Annual SEERC Doctoral Student Conference (DSC 2011) (2011) In Print.
6. P.Kefalas, G.Eleftherakis, E.Kehris: Communicating x-machines: A practical approach for formal and modular specification of large systems. Information and Software Technology **45** (2003) 269–280

7. Gh.Păun: Membrane Computing: An Introduction. Springer, Berlin (2002)
8. I.Stamatopoulou, P.Kefalas, M.Gheorghe: Operas: A framework for the formal modelling of multi-agent systems and its application to swarm-based systems. In: ESAW, Berlin, Heidelberg, Springer-Verlag (2007) 158–174
9. I.Stamatopoulou, M.Gheorghe, P.Kefalas: Modelling dynamic configuration of biology-inspired multi-agent systems with Communicating X-machines and Population P Systems. Volume 3365:389-401 of LNCS. Springer-Verlag, Berlin (2005)
10. P.Kefalas, M.Holcombe, G.Eleftherakis, M.Gheorge: A formal method for the development of agent based systems. In V.Plekhanova, ed.: Intelligent Agent Software Engineering, Idea Group Publishing Co. (2003) 68–98
11. F.Ipate, M.Holcombe: Specification and testing using generalised machines: a presentation and a case study, Software Testing, Verification and Reliability (1998) 61–81
12. U.Wilensky: NetLogo Segregation model. Center for Connected Learning and Computer-Based Modeling, Northwestern Univ., Evanston, IL. (1997) http://ccl.northwestern.edu/netlogo/models/Segregation.
13. U.Wilensky: NetLogo. Center for Connected Learning and Computer-Based Modeling, Northwestern Univ., Evanston, IL. (1999) http://ccl.northwestern.edu/netlogo/.
14. M.Pogson, R.Smallwood, E.Qwarnstrom, M.Holcombe: Formal agent-based modelling of intracellular chemical interactions. Biosystems **85** (2006) 37–45
15. R.Smallwood, M.Holcombe, D.Walker: Development and validation of computational models of cellular interaction. Journal of Molecular Histology **35** (2004) 659–665
16. D.E.Michael, G.G.William, K.Yoshio, D.Notkin: Dynamically discovering pointer-based program invariants. Technical Report UW-CSE-99-11-02, University of Washington Department of Computer Science and Engineering, Seattle, WA (November 1999) Revised March 17, 2000.
17. G.J.Holzmann: The model checker spin. IEEE IFans. on Software Engineering (1997) 279–295
18. M.Kwiatkowska, G.Norman, D.Parker: Prism: Probabilistic symbolic model checker. In Proc. PAPM/PROBMIV'01 Tools Session (2001) 7–12
19. K.L.McMillan: Symbolic Model Checking. Kluwer Academic Publishers, Englewood Cliffs (1993)