Simulation Modelling Practice and Theory 40 (2014) 144–160

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat

Adaptive agent abstractions to speed up spatial agent-based simulations

Abbas Sarraf Shirazi [a], Timothy Davison [a], Sebastian von Mammen [b], Jörg Denzinger [a], Christian Jacob [a,c,*]

[a] Dept. of Computer Science, Faculty of Science, University of Calgary, Canada
[b] Institut für Informatik, University of Augsburg, Germany
[c] Dept. of Biochemistry & Molecular Biology, Faculty of Medicine, University of Calgary, Canada

ARTICLE INFO

Article history:
Received 9 February 2013
Received in revised form 3 September 2013
Accepted 7 September 2013
Available online 24 October 2013

Keywords:
Agent-based simulation
Abstraction
Optimization
Online learning

ABSTRACT

Simulating fine-grained agent-based models requires extensive computational resources. In this article, we present an approach that reduces the number of agents by adaptively abstracting groups of spatial agents into meta-agents that subsume individual behaviours and physical forms. Particularly, groups of agents that have been clustering together for a sufficiently long period of time are detected by *observer* agents and then abstracted into a single meta-agent. *Observers* periodically test meta-agents to ensure their validity, as the dynamics of the simulation may change to a point where the individual agents do not form a cluster any more. An invalid meta-agent is removed from the simulation and subsequently, its subsumed individual agents will be put back in the simulation. The same mechanism can be applied on meta-agents thus creating adaptive abstraction hierarchies during the course of a simulation. Experimental results on the simulation of the blood coagulation process show that the proposed abstraction mechanism results in the same system behaviour while speeding up the simulation.

1. Introduction

Agent Based Models (ABM) provide a natural means to describe complex systems, as agents and their properties have a convenient mapping from the entities in real world systems. The interaction of agents in ABM gives rise to an interesting concept in the study of complex systems: emergent phenomena, higher-level properties or behaviours that are not easily traceable in the lower-level entities [1]. Moreover, ABM capture discontinuity in individual behaviours, which is difficult when modelling with an alternative like differential equations [2].

The flexibility of ABM comes at a computational cost. As the granularity of a model increases, so do the computational resources needed to simulate all of the interactions among the agents, which directly translates into longer simulation times. Some researchers have restricted agent interactions to be only among neighbouring agents in a two or three-dimensional lattice [3,4]. However, changing the interaction topography among agents is a necessary feature in some models, e.g. developmental processes [5]. Others have utilised parallel computing to meet the computational demands of ABM [6,7]. Finally, many researchers have proposed super-individuals [8]: agents that encompass other agents, e.g. a single super red blood cell agent that subsumes and represents thousands of individual red blood cell agents.

* Corresponding author at: Department of Computer Science, University of Calgary, Calgary, AB T2N 1N4, Canada. Tel.: +1 4032207682.
E-mail addresses: asarrafs@ucalgary.ca (A. Sarraf Shirazi), tbdaviso@ucalgary.ca (T. Davison), sebastian.von.mammen@informatik.uni-augsburg.de (S. von Mammen), denzinge@ucalgary.ca (J. Denzinger), cjacob@ucalgary.ca (C. Jacob).

1569-190X/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.simpat.2013.09.001

In this paper, we extend our previous work by proposing another type of abstraction that aims to build adaptive hierarchies of spatial agents during the course of the simulations. To this end, *observer* agents are immersed in the simulation to monitor groups of agents. The *observers* try to detect a cluster of agents that have adhered to one another for a sufficiently long duration of time. Once an *observer* finds such a cluster, it abstracts the agents into a single meta-agent that subsumes both the behaviour and the structure of the individual agents in that cluster. As the dynamics of the simulation change, groups of agents may no longer stick together and therefore the *observer* needs to break down those meta-agents into their constituent individual agents. An unsupervised validation mechanism ensures the validity of meta-agents by periodically monitoring whether they should continue to subsume their agents. Since meta-agents have the same basic definition as the individual agents, the same abstraction process is applied on them, thus making adaptive abstraction hierarchies during the course of the simulation.

The remainder of this paper is organised as follows. Section 2 reviews related works both in solving the problem of scalability and in dealing with higher-order patterns in agent-based simulations. Section 3 gives a formal definition, along with a computational timing analysis of our component-based agent framework – *LINDSAY Composer*. Section 4 presents our abstraction framework with a detailed description of the involved steps and algorithms. We conclude this section with a computational timing analysis of our abstraction. In order to demonstrate the effectiveness of this approach, we apply it to an agent-based blood coagulation simulation and report the results in Section 5. Finally, Section 6 provides a comparison between this work and our previous work, and presents the concluding remarks.

2. Related work

Agent based models operate at the individual level and describe potentially numerous behaviours for all of their constituent units. Simulating all of the individual behaviours is therefore considered to be extremely computationally intensive [2,9–12]. It has been suggested that abstracting higher-order patterns could reduce the computational complexity of ABM without introducing much overhead [13,12,14]. In this section, we briefly describe the attempts made to address the problem of scalability and performance in ABM, then we review the works that motivated this research.

2.1. Scalability and performance in ABM

Bonabeau points out that despite increasing computational power, simulating all the individual behaviours in ABM still remains a problem when it comes to modelling large-scale systems [2]. Research in improving the scalability of ABM is roughly categorised into two groups: (1) parallel computing and (2) grouping similar agents into a single agent.

The first category, parallel computing, tries to concurrently simulate clusters of agents that interact primarily with one another without much intra-cluster communication. Efficiency is improved as long as the time spent on synchronisation is much less than the time spent on computation [6]. Scheutz and Schermerhorn developed a framework with two algorithms for the automatic parallelization of ABM [6]. Particularly, they developed a separate algorithm for spatial agents, as their location data can efficiently determine in what cluster they should be simulated.

Along the same line, Lysenko and D'Souza propose a framework to use Graphics Processing Units (GPU) to parallelize an agent-based simulation [7]. They utilise a technique in General Purpose Computing on GPUs called state textures [15] to map each agent to a pixel. A pixel is defined by its colour components: Red, Green, Blue, and Alpha (RGBA). Each numerical property of an agent is thus mapped to a colour component. If an agent cannot be squeezed into four floating point values, then extra colour buffers should be used, which in turn adds to the complexity of the problem.

The second category of grouping similar agents deals with the granularity of an agent. For example, super-individuals can represent groups of agents. Scheffer et al. suggest assigning an extra variable to each agent to denote how many agents it represents [8]. More advanced algorithms have been proposed to find super-individuals during the course of a simulation. Stage et al. propose an algorithm called COMPRESS to aggregate a cluster of agents into one agent [16]. They divide their algorithm into two stages to avoid applying a time-consuming clustering algorithm on the space of all the attributes in all the agents. In the first stage, they calculate a linear combination of attributes l_i for each agent i by applying principal component analysis (PCA) [17]. Then this list is sorted to find n clusters of agents with the largest gaps in l_i. In the next stage the clusters are further subdivided based upon their variance until the variance is within a given range. The first stage maintains overall system variations while the second stage reduces the intra-cluster variations.

COMPRESS is a static algorithm, in that once a cluster of agents is replaced by one agent, the original agents will not be released back into the simulation. Wendel and Dibble extend the static COMPRESS algorithm with the Dynamic Agent Compression (DAC) algorithm in which higher-order agents are created and destroyed based on the heterogeneity of agents in the system [18]. They define two special agents in their system: (1) *container agents* which are the higher-order agents and (2) a *compression manager* which handles all the queries to individual agents thus making the container agents invisible to the model. It also creates and destroys other agents. For example, upon receiving a create request from the model, the compression manager decides if it has to create a new individual or whether the create request can be ignored, as there already exists an agent with the same attributes. In DAC, a container agent monitors its encompassed agents, and