Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Ausarbeitung Anwendungen 2

Christian Hüning, christian.huening@haw-hamburg.de

Scalability and Usability of modern Multi-Agent Systems

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Christian Hüning, christian.huening@haw-hamburg.de

Scalability and Usability of modern Multi-Agent Systems

Ausarbeitung Anwendungen 2 eingereicht im Rahmen der Ausarbeitung Anwendungen 2

im Studiengang Bachelor of Science Angewandte Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Thiel-Clemen

Eingereicht am: 11. Juni 2014

Christian Hüning, christian.huening@haw-hamburg.de

Thema der Arbeit

Scalability and Usability of modern Multi-Agent Systems

Stichworte

Multiagentensystem, Usability, Skalierbarkeit Ökologie, Verteilte Systeme, Systemmodellierung, Lastverteilung

Kurzzusammenfassung

Christian Hüning, christian.huening@haw-hamburg.de

Title of the paper

Scalability and Usability of modern Multi-Agent Systems

Keywords

Multi-Agent-System, Usability, Scalability, Ecology, Distributed Systems, Systemmodelling, Load Balancing

Abstract

Contents

1 Introduction

A large number of simulation tools for MAS are available and been put to good use by many scientists in various fields (Amouroux *et al.*, 2007; Bellifemine *et al.*, 2008, 2007; Le *et al.*, 2008; North *et al.*, 2013; Ralha *et al.*, 2013). All these tools provide either excellent generality, usability and additional features like easy data import or comfortable visualization and analysis options or excel to produce great results in a specific domain.

However scientists demand for large-scale simulations is continously increasing and that is where the above mentioned systems fall short in terms of scalability. Some of them try to adapt to that by implementing distribution solutions on top of the original solutions, but usually that approach fails because of prior decisions in their architecture.

This is where other, more modern ideas and concepts come into play (Collier & North, 2012; Suryanarayanan *et al.*, 2013; Vigueras *et al.*, 2013). These approaches focus on scalability of MAS and thus attempt to answer the requirement for effiecient execution of larger scale models.

Since these approaches are very new and have a clear focus, they might fall short in terms of other features already delivered by their predecessors and thus introduce a gap between usability and scalability. To investigate how well these systems and concepts meet the requirements of modern multi-agent simulations, the Abdoulaye model (Pereki *et al.*, 2013) was analyzed amongst other models currently under development in the MARS group.

As a result this paper provides the requirements of a modern MAS (Chapter 2), discuss most recent work towards scalable and usable MAS frameworks (Chapter 3) and finally provide a conclusion on the matter as well as providing an outlook on my own work on the MARS system (Chapter 4).

2 Requirements of modern MAS

This section provides a summary of the most important requirements found for a modern simulation system. Findings from various corresponding work, as well, as the experience from the MARS project group are also featured. (http://www.mars-group.org), while designing a solution for the tasks at hand (e.g. Pereki *et al.* (2013)).

2.1 Modularity and Reusability

As shown (Liu *et al.*, 2007), almost every ecosystem today is tightly coupled with its neighboring economic or social systems and thus these need to be taken into account when watching the evolution of that ecosystem. Filatova *et al.* (2013) go even further by demanding that the corresponding aspects of ecological systems like economy, social systems and bio-physical dynamics need to be integrated into the representation of a heterogeneous landscape representation.

The integration of existing models is one of the most important requirements resulting from this circumstances. This can only be done if models or their parts, are designed in a modular and reusable manner. The idea is to connect and integrate domain specific models from domain specific experts to create a new super model of a certain domain or to reuse sub-models in completely different domains. If for example one would want to create a large scale model of a given ecosystem in south Africa, it would be very helpful, if one could use already existing models of certain components, such as animal behaviors, weather, land erosion and so on.

Comparison is another aspect that could profit from modular and reusable models. If it was easy to integrate most of the models available, models could be run directly next to each other, consuming the same data, allowing for example to perform real-time digression analyses.

Actually integrating models turns out to be extremely difficult, since each group of scientists working on a model, tends to use another, individual, paradigm, architecture, programming language or data format. A good solution should address this problem.

2.2 Information Integration

Of huge importance in simulation is data. It is needed for nearly all tasks from generation of hypotheses, over simulation initialization and calibration to validation. Unfortunately the data that is being collected, has a tremendous heterogeneity in terms of temporal and spatial resolution, reference formats, completeness and error margins. To be viable in a simulation, this data has to be integrated. It must be carefully corrected, the resolutions have to be aligned, the error must be treated.

Furthermore the relevant data of all the available must be singled out and connected. Since the MARS group focuses on spacially explicit simulations, a special point is also to link data without any further reference together to establish a common context. For example we might be designing a model for an animal species in a wildlife reserve somewhere in Africa. For one concrete simulation it could be necessary to include weather data for the whole region, topology data of the general landshape, as well as a rough overview of vegetation types and population metrics for certain species in that area.

A simulation framework should assist domain experts with all the steps involved: GIS imports, data collection, data analysis and possibly transformation. These tasks target the difficulty when technically connecting different models. A more functional view to the importance of information integration has been made by (Liu *et al.*, 2007) who take a look at the complexity of coupled human and natural systems. Their integration efforts aim at taking interdisciplinary research on a broader scale into account, as well as exceeding local and temporal boundaries when modelling certain ecological system.

2.3 Scalability

Although it should always be the goal of a modeler, to design everything as simple as possible, some things are inherently computationally intensive. There are several scenarios that, often in combination, prohibit simulation execution on a single computer within reasonable time frames. First of all the agents themselves are becoming more complex, in order to replicate natural behavior. This is especially true for animate objects, such as for example animals or humans. To come close to the real world, the modeler might need to use computationally expensive techniques, such as learning or planning algorithms, path-finding, collision avoidance and others, often even simultaneously. And the more models are integrated, the more of those techniques are likely to occur.

As the field of multi-agent systems research matures, the applications get also bigger, resulting in a larger number of agents. Imagine for example a continuous field with an average agent density of one agent per square meter accordingly; the system has to handle about 100 agents. Now, if the length of the field's sides is only doubled, the computational effort increases fourfold, in the three-dimensional case even eightfold .

The real world areas of interest are steadily growing larger, further intensifying this problem. This is especially true, when a model is used to forecast future developments of its real world counterpart. Initially mostly used for the understanding of dynamic systems, IBM is likely to be used increasingly for large scale prognosis as well. The area of interest may for example be something like the Abdoulaye forest (Pereki *et al.*, 2013) as in my current use case.

Of course it may sometimes be possible to avoid the problem by extrapolating from a sample set of agents to the bigger scenarios. But that would in return diminish the factor that sets apart IBM from other simulation techniques: the ability to track individual agent's actions and states. Also, depending on the system, some desirable emergent properties of the real system are only achievable with a realistic density of agents. For example Yamamoto *et al.*, 2008 found that massively increasing the amount of agents in an auction simulation, significantly changed the outcome of the simulation.

The most promising solution to really solve this problem, is to make the simulation system scalable across multiple computers. Research budgets are not limitless, so I think it is important to target commodity hardware or rentable compute clouds. Scalability by my definition means the computation speed of a single simulation run increases by a constant factor per added compute node.

2.4 Ease of Use

To be useful for and accepted by experts of other domains than computer science, a simulation system should also be as accessible as possible. There are two aspects that I want to emphasize in this context. One is the ever important question of usability of the general toolset. The other is the nature of the means provided by the simulation system to model the actual questions. Specifically a good solution should address and overcome the gap between the domain specific model and its corresponding technical representation in the simulation system.

2.5 Visualization

Since I am opting for large scale simulations with millions of agents, I need a visualization solution that copes with these numbers. To the best of my knowledge there is no affordable current graphics engine or hardware that allows to render these numbers in real-time at once to the screen.

Therefore my solution should be able to visualize only a specified section of the whole simulation space. I further require that section to be dynamically movable and resizable. As there might be a lot of data bound to the entities and patches of environment I am to visualize, a good solution should allow to enable and disable certain levels of information (e.g. weather, vegetation etc.). Lastly, the rendering should also be efficiently possible in 3D, if a scenario demands or would greatly benefit from it, like for example evacuation scenarious.

3 Scalable MAS Frameworks

This section provides a detailed discussion of the current state-of-the-art of scalable multi-agent system frameworks or concepts. I will specifically have a look at the aspect of scalability, how it is achived and what implications on modelling and usability these technological decisions imply.

3.1 PDES-MAS

3.1.1 Overview

PDES-MAS by Suryanarayanan *et al.* (2013) is short for Parallel Discrete Event Simulation. The whole multi-agent system is modelled from logical processes, which may be distributed across several compute nodes.

PDES differenciates two types of logical processes (LP). Agent Logical Processes (ALP) model the agents' behaviour, whereas Communication Logical Processes (CLP) represent communication and interaction betweend agents. The overall paradigm used to model the latter are Shared State Variables (SSV), which hold all information important to the simulation and are changed concurrently by the agent processes. SSVs reside in the CLPs.

The scalability problem is now solved by arranging the CLPs in a tree of predefined fixed size and with the ALPs as leafs. So each ALP is directly attached to a CLP which allows for a possible colocation of logic and data. At initialization all SSVs are placed in the root node of the CLP tree. As the ALPs start working, the SSVs are being repartitioned to CLPs residing closer to the accessing ALPs. This process is called State Migration by the authors.

All ALPs are executed by a round robin scheduler and manage their own local virtual time (LVT). This local virtual timestamp is used by the SSVs in the CLPs when there is need for a rollback. Also each SSV stores a history of recents changes, mapped by the LVT. Rollbacks are needed when the tree gets repartitioned and / or messages between nodes get lost or are delayed.

3.1.2 Discussion

Throughout the tests the authors conducted, it became apparent that there is an optimal number of CLPs for a given simulation and ALP count. If the ALP / CLP ratio is too low, e.g. there are too many CLPs, the overhead of reorganizing and initializing the whole tree becomes too large. If there are too less CLPs on the other hand, the benefit of distribution is lost.

It has to be noted that the concept of PDES-MAS looks rather similiar to that of TupleSpaces / Linda from Gelernter & Carriero (1992) as it implements the concept of a distributed shared memory. This system, though scaling very well, raises questions when it comes to usability. Suryanarayanan *et al.* (2013) don't present a solution for importing data, visalization or an easy enough way to implement a model. The aspect of model reusability could also become very complex as a sub-model is fully integrated with all other sub-models due to the usage of shared state variables. It can be expected to be very difficult to cut out a single sub-model out of the whole set of SSVs.

3.2 Repast HPC

3.2.1 Overview

Collier & North (2012) present Repast HPC as the distributable brother of RepastJ or Repast Symphony as the latest version of the famous simulation framework is called. Collier & North (2012) motivation to building a large scale MAS is very similiar to that of the MARS groups' . That is to allow large-scale model simulation instead of optimizing a smaller-scale model by running many parallel simulations of the same model.

Repast HPC translates models into working simulations through a concept of agents, contexts and projections. A context is a set of agents, whereas the term set corresponds to its mathematical definition. Projections at last use contexts to model the environment. This structure allows for multiple agents to take part in multiple environments, as well as to reuse certain projections.

To distribute a simulation Repast HPC uses a concept called Shared Projections. The environment created by a projection basically is a 2D grid due to the usage of the Logo language. This grid is sliced and then distributed across several processes. The slices are created by means of an influence sphere, which represents the space an agent is or may be active in. To optimze communication a shared grid buffer is attached to each slice. The buffer holds non-local agent stub objects from the neighbouring slices and thus allows for changes / interactions to be made locally at first. The system then distributes the changes to the corresponding home objects in the other processes and takes care of synchornization matters.

3.2.2 Discussion

Just like Suryanarayanan *et al.* (2013) the work of Collier & North (2012) provides a very scalable solution, which also allows for model reusability through its projections and contexts. However the communication and distribution algorithms rely on a strong localized behaviour of agents. It would be interesting to observe performance of the system with a simulation model lacking this feature.

From a usability point of view the communication and synchronisation mechanism do also not sound too friendly, since the user has to provide specfic pieces of code for each class he wants to take part in it. A more transparent solution would be highly desireable.

Also it must be noted that Collier & North (2012) used high-end super computing hardware (IBM BlueGene cluster with up to 65.536 cores and Infiniband network) which makes it questionable how the system will run and scale on commodity hardware. The latter would also allow smaller research teams to make use of the system.

The intense usage of the Logo language paired with RepastHPC only supporting 2D environments, mark clear restrictions towards the models, which can be implemented. It is rather complex and uncomfortable to map a 3D environment onto a 2D representation. This could be observed during the development of the WALK system (Thiel, 2013). In terms of development the authors provide the information that a skilled developer with good knowledge about both RepastJ and RepastHPC was able to translate an epidemiology model approximately within a week.

Collier & North (2012) do also not address the problem of data import, but it can be assumed that we will see that feature in the near future, since Repast Symphony is pretty strong in that field. The same may be true for the challenge of visualization. While GAMA (a fork of RepastJ by Amouroux *et al.* (2007)) has very good visualization features, RepastHPC currently only supports the creation of a global logfile with results from the simulation.

3.3 Vigueras

3.3.1 Overview

The architecture proposed by Vigueras *et al.* (2013) was developed to provide a solution for interactive simulations. In this case the interaction was meant to be a viewer process which allows to watch the simulation in near real-time. Later additions could include the possibility to interactively take part in the simulation like in a game or similiar features.

To achieve this goal Vigueras *et al.* (2013) also went with the paradigm of logical processes and in addition decided to have the simulation executed completely asynchronously. The environment is sliced into pieces and each piece is assigned to one dedicated ActionServer (AS) process. An ActionServer manages position of agents and similar manners in its slice. The agents are managed by ClientProcesses (CP), whereas one CP may manage agents living on different CPs. ASs and CPs each reside on their own physical hosts, which allows to increase the overall performance.

Since the whole system postulates to be asynchronous, agents and thus CPs only need to synchronize when they happen to operate near the borders of an environment slice. This approach again introduces the presumption of localized agent behaviour, which might not always be the case.

Finally to achieve the goal of real-time visualization Vigueras *et al.* (2013) consequently use their approach of a seperate process per simulation task and thus feature a VisualClientProcess (VCP), which again resides on its own hardware. A VCP defines a camera that looks onto the environment and by that only sees a part of the overall simulation space.

3.3.2 Discussion

Vigueras *et al.* (2013) present a scalable solution which prominently features the aspect of visualization. Therefore aspects like data integration, model definition, reusability etc. are not mentioned in their paper. It remains unclear how these tasks would be done in their architecture. Also the decision to support only asynchronous simulation execution implies a lack of temporal precision when it comes to validation and similar matters. Conclusively it can be stated that Vigueras *et al.* (2013) provide a good approach for efficient visualization which resolves the bottleneck of other systems like GAMA, that always visualize the whole environment at once.

4 Conclusion & Outlook

In this work the most important requirements for a modern MAS as found throughout the work on the Abdoulaye model (Pereki *et al.*, 2013) are described and three modern framework approaches to MAS from the recent past were discussed. As a result the existence of a gap between usability and scalability in the landscape of MAS frameworks was observed. A lot of very good and accessable tools like RepastJ, GAMA, NetLogo, JADE etc. are used by many researches in all domains, but their lack of scalability becomes more and more evident as an increasing number of researchers call for larg-scale simulation models.

The scalable architectures on the other hand are short on usability features and require the domain experts to learn a large amount about the system itself or to get help by somebody from the computer science domain. A good solution should at least allow the domain expert to start on his own and concentrate on the modelling process, the compilation of data and to run first tests including some sort of validation and visualization process.

As a first prototypic implementation of MARS is in finalization currently, the creation of a first simulation model of the Abdoulaye forest will be on focus. This first version will include a tree growth and seed dispersal model and will be used to simulate the development of biomass in Abdoulaye. The system will be tried to be scaled up to represent the full area of Abdoulaye which would result in roughly 20 million tree agents on an area of about 300 square kilometres. The results from these tests will then be compared to those of the other frameworks presented in this paper.

As of the writing of this paper, at least five other models from different domains are being created for the MARS system, including classic predator prey models and evacuation scenarious but also socio-ecological and climatic models as well as a simulation of the human immune system. The feedback and experience regarding the usability of MARS from theses works, is going to be used to eventually evaluate if MARS is capable of filling the gap mentioned above.

Bibliography

Amouroux, Edouard, Thanh-quang, C H U, Boucher, Alain, & Drogoul, Alexis. 2007. GAMA : an environment for implementing and running spatially explicit multi-agent simulations.

Bellifemine, Fabio, Caire, Giovanni, Poggi, Agostino, & Rimassa, Giovanni. 2008. JADE: A software framework for developing multi-agent applications. Lessons learned. *Information and Software Technology*, **50**(1-2), 10–21.

Bellifemine, FL, Caire, G, & Greenwood, D. 2007. *Developing multi-agent systems with JADE.*

Collier, N., & North, M. 2012. Parallel agent-based simulation with Repast for High Performance Computing. *Simulation*, **89**(10), 1215–1235.

Filatova, Tatiana, Verburg, Peter H., Parker, Dawn Cassandra, & Stannard, Carol Ann. 2013. Spatial agent-based models for socio-ecological systems: Challenges and prospects. *Environmental Modelling & Software*, Apr., 1–7.

Gelernter, David, & Carriero, Nicholas. 1992. Coordination languages and their significance. *Communications of the ACM*, **35**(2), 96.

Le, Quang Bao, Park, Soo Jin, Vlek, Paul L G, & Cremers, Armin B. 2008. Land-Use Dynamic Simulator (LUDAS): A multi-agent system model for simulating spatio-temporal dynamics of coupled human–landscape system. I. Structure and theoretical specification. *Ecological Informatics*, **3**(2), 135–153.

Liu, Jianguo, Dietz, Thomas, Carpenter, Stephen R, Alberti, Marina, Folke, Carl, Moran, Emilio, Pell, Alice N, Deadman, Peter, Kratz, Timothy, Lubchenco, Jane, Ostrom, Elinor, Ouyang, Zhiyun, Provencher, William, Redman, Charles L, Schneider, Stephen H, & Taylor, William W. 2007. Complexity of coupled human and natural systems. *Science (New York, N.Y.)*, **317**(5844), 1513–6.

North, Michael J, Collier, Nicholson T, Ozik, Jonathan, Tatara, Eric R, Macal, Charles M, Bragen, Mark, & Sydelko, Pam. 2013. Complex adaptive systems modeling with Repast Simphony. *Complex Adaptive Systems Modeling*, **1**(1), 3.

Pereki, Hodabalo, Wala, Kperkouma, & Thiel-Clemen, T. 2013. Woody species diversity and important value indices in dense dry forests in Abdoulaye Wildlife Reserve (Togo, West Africa). *International Journal*, **5**(June), 358–366.

Ralha, Célia G., Abreu, Carolina G., Coelho, Cássio G.C., Zaghetto, Alexandre, Macchiavello, Bruno, & Machado, Ricardo B. 2013. A multi-agent model system for land-use change simulation. *Environmental Modelling & Software*, **42**(Apr.), 30–46.

Suryanarayanan, Vinoth, Theodoropoulos, Georgios, & Lees, Michael. 2013. PDES-MAS: Distributed Simulation of Multi-agent Systems. *Procedia Computer Science*, **18**(Jan.), 671–681.

Thiel, Christian. 2013. *Analyse von Partitionierungen und partieller Synchronisation in stark verteilten multiagentenbasierten Fußgängersimulationen.* Master Thesis, Hamburg University of Applied Sciences.

Vigueras, Guillermo, Orduña, Juan M., Lozano, Miguel, & Jégou, Yvon. 2013. A scalable multi-agent system architecture for interactive applications. *Science of Computer Programming*, **78**(6), 715–724.

Yamamoto, G, Tai, H, & Mizuta, H. 2008. A platform for massive agent-based simulation and its evaluation. *Massively Multi-Agent Technology.*