Proceedings of the 38th Annual Conference on Computer Applications and Quantitative Methods in Archaeology, CAA2010
F. Contreras, M. Farjas and F.J. Melero (eds.)

Large-scale Agent-based Simulation in Archaeology: an Approach using High-performance Computing

Rubio, X., Cela, J.M.
Computer Applications in Science & Engineering, Barcelona Supercomputing Center, Spain
{xrubio, josem.cela}@bsc.es

Agent-Based Modelling is one of the techniques with more potential to develop useful simulations for archaeological research. On the other hand, existing ABM tools are not well suited to manage the high computational cost that some of the approaches taken by researchers can demand. This paper discusses possible pitfalls of the use of ABM in Archaeology, as well as a new platform developed to deal with them through the use of High-performance computing.

Keywords: Agent-Based Simulation, High-Performance Computing, GIS.

1. Introduction

Agent-Based Modelling is recognised as one of the techniques with more potential to develop useful simulations of complex social interactions (GILBERT, 2008). The basic methodology tries to simulate the relations inside a small-scale society in order to understand behavioural emergence, thus exploring an abstract and simplified model (EPSTEIN and AXELL, 1996).

Focusing on archaeological science, a growing number of researches have used ABM as a basic modelling system (DORAN, 1999), although the case studies are often more specific. Given the fact that this discipline has at its disposal the information provided by the archaeological record, ABM is useful to test hypothesis on behaviour agents, validating the results of the simulation against the known material culture (BARCELÓ, 2009:311-331; LAKE, 2000). One of the main problems of the validation of these facsimile models (GILBERT, 2008: 43-44) is that it will be impossible to replicate the past processes during the simulation, given the complex and chaotic nature of societies. Results provided by these simulations will be forcibly different than the real, so it will be necessary to run an important number of executions. It will be useful to combine different hypothesis and generate statistical data, more suitable to the validation, in order to understand the system dynamics.

The consequence is that a research that uses facsimile models will need to make a great effort executing as many simulations as possible to test different hypothesis, thus requiring a huge amount of computation.

Moreover the data used in these specific case studies is often spatially referenced (being collected in archaeological works), so we will need to track this sometimes huge volume of information, as well as the one coming from the simulation in order to validate results. If the project is big enough, a desktop computer or a small cluster can be insufficient to manage the amount of spatial information, resulting that some researches can be forced to decrease the quality and quantity of raw data managed by the simulation.

Finally, the number of agents, and interactions between them, can be extremely large in some of these specific cases, thus forcing the scientist to limit its number in order to execute the simulation on a standard computer.

Although some of the existing ABM platforms try to fix these problems through the use of distributed systems, none of them is specifically designed for its execution in distributed supercomputers, probably the hardware architecture more suited to execute large-scale simulations.

This paper is an approach to the use of High-performance computers as one of the solutions to the particular requirements given by the application of ABM simulation in archaeological research, specially in the case of large-scale projects that need a sizeable amount of computing power.

The next section is dedicated to discuss the computational costs of large-scale agent-based

simulations. The third section focuses on existing initiatives focused on using High-performance computing to solve ABM projects. In fourth section a new ABM software platform particularly designed to its use in supercomputers is presented, trying to deal with the requirements defined by this type of simulation and research field.

2. The challenges of large agent-based simulations

As we have stated, the limitations of high computational costs can derive a well-defined model on a poor simulation in some particular cases. Some of the issues that a large-scale simulation needs to solve are explained below.

2.1. Number of executions

The exploration of past social processes makes almost mandatory the use of statistics. The exact reproduction of past facts is impossible, and as a consequence every execution of a system designed to model a society will be different from the others, following its non-deterministic nature. The stochastic analysis of the results is one of the most important tools to understand human societies, and in the case of ABM we need to statistically study the results of our simulations. The problem is that the computational cost of some of these models is high, the result being a limited number of samples. An example of this issue is this discussion regarding the Stepping Out project, where the number of executions of a cellular automata was limited to 30 runs for the standard data set (see MITHEN and REED, 2002; NIKITAS and NIKITA, 2005).

2.2. Election of variable values and behaviours

A related problem is that, in any simulation of past societies, an important percentage of behaviours are determined a priori, as well as the values of some variables (i.e. reproduction rate, lifetime expectancy). Some disciplines (as anthropology and sociology) can be useful to define this information, but in any case the researcher will need to execute several simulations exploring the entire space of values to get a clear picture of the most feasible ranges and behaviours of the real system, in order to define a model as close to the former as possible.

2.3. Spatial resolution

As most of archaeological data can be referenced spatially, most agent-based simulations will need to address this issue (being the exception abstract models, or models where space is discarded as non-essential in the modelling process). Space is important in two different phases of the research; the first one is execution, as most agents will "live" in some determined location and landscape. The other one is hypothesis validation, as the results of the simulation will be tested against real data spatially referenced, coming from archaeological works.

This is the reason why the use of a Geographical Information System linked to the chosen simulation tool, in order to manage the spatial data generated by the simulation, is often necessary. On second term a GIS will be useful when a researcher needs to test the initial hypothesis against the archaeological record. Some of the most popular agent-based simulators are not suited for this task, while others provide just limited interaction with GIS software (i.e. location of agents, loaded at initialization time). Finally, in the case of large-scale simulations, the resolution of available data often will need to be decreased to avoid higher computational costs.

2.4. Time steps

ABM simulations are executed by running of a sequential number of discrete steps, each one using the resulting data of the preceding one (thus simulating time). The simulation will need to define a time scale for these steps, the final value being a compromise between computational cost and accuracy.

Moreover, it is important to analyse the relation between time and space resolutions. Some problems need a large time step because we are incapable of knowing the value of a given variable on higher resolutions, but, on the other hand, some spatial behaviours will be forcibly ill-defined at this time interval. For example, if we try to simulate a migration, we will be incapable of defining behaviour at a daily time step rate, equalling usually each step to some decades or centuries. On the other hand, the movement capabilities of a human would be restricted, as on the other case it would produce unrealistic results, because some of the agents could have walked through an entire continent in just one time step.

Higher computational assets could be a solution, as the researcher will be capable of executing an increased number of steps, as well as defining the different parameters through the execution of multiple runs.

3. ABM and High-performance computing

As we have seen, some of the problems of the use of ABM techniques in archaeological research that can be solved through the use of supercomputers. Two different approaches can be taken, depending on the complexity of the simulation.

The first one is easier to implement, and simply consists on the parallelization of different executions in different computer nodes of a distributed HPC architecture. This solution is straightforward, as it doesn't require any additional code to be developed, and any suitable existing ABM tool can be used. The drawback of this task distribution is that the computational cost of a given

simulation can't be divided amongst different computer nodes.

The second approach is precisely the parallelization of a single simulation between different computer nodes. This solution is particularly interesting for large-scale simulations, both in time and space scales. Since time can't be distributed, because every time step depends on the past one, the only solution is the distribution of the landscape between different computer nodes.

Unfortunately most of the existing tools are not well suited for this task, although there is some work in progress, particularly with the popular RePast toolkit (MINSON *et al*., 2008). Unfortunately is not an easy task, as common tools were not developed for being executed on a HPC environment.

Other prototypes have been developed in order to fix this issue, but they are not specially well suited for the requirements that an archaeological research project defines, specially regarding spatial reference (TAKAHASHI and MIZUTA., 2006). In this case a scheduler has been implemented to load-balance the different nodes without shared memory inside a supercomputer.

The important factor in distribution is the number of interactions between agents; agents closely related in terms of social interaction will be clustered in the same node, thus limiting the communication rate between nodes. This approach is extremely effective, but the need of spatial reference is a problem that is not solved. Agents will modify the landscape where they are living, thus having the necessity of passing these changes between nodes on an effective way, a problem not relevant for this approach.

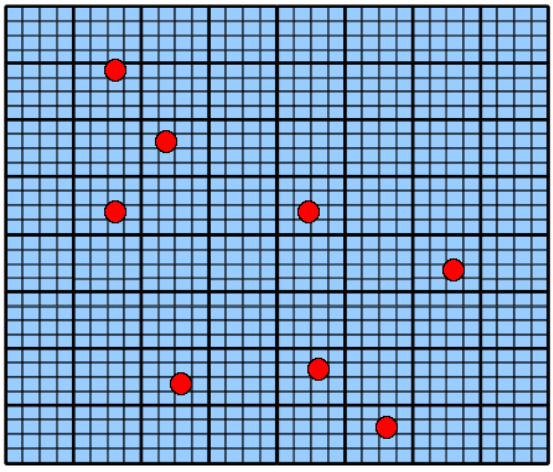

Figure 1: *Raster data is divided regularly, each submatrix being owned by a different node. Agents are managed by the node where they are spatially located, dictated by their position in landscape.*

4. A new tool to execute HPC Agent-Based simulations

The tool currently being developed at the BSC (Barcelona Supercomputing Center) tries to devise solutions to these issues. The landscape where the agents live is defined as a raster structure, a geographic model commonly used in GIS (LONGLEY, 2006:76). Complementing this basic landscape, other rasters can be created, in order register defined parameters that are spatially referenced. Each raster is a bi-dimensional matrix, that is divided in a regular way, being each computer node the owner of a part of the global matrix, as well as the agents located there (Figure 1).

This distributed execution would not be efficient, as two adjacent computer nodes could be modifying at the same time common data, thus creating conflicts in the simulation (i.e. two agents, each owned by different computer nodes, moving to the same location). In order to avoid these problems the landscape owned by each node is at the same time divided in four matrices, each of them being managed by an independent process numbered 0 to 3. The system only allows the execution of simultaneous processes if they have the same number, thus avoiding the execution of adjacent neighbours capable of modifying the same spatial region as Figure 2 shows. When the entire group of regions has executed a time step, the next one is ready to begin.

The management of communications between nodes has been developed using the Musik library, specifically designed to execute discrete-event simulations in a distributed environment (see PERUMALLA, 2005; PERUMALLA, 2006; STEPHAN, 2008).

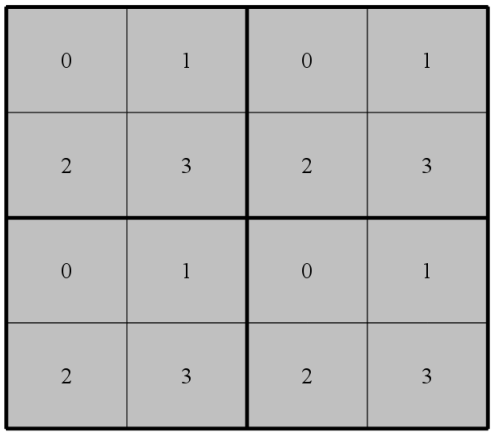

Figure 2: *Each node divides its section of the landscape between four processes. The group of processes with the same id number is executed simultaneously (0's, 1's, 2's and 3's), and no group will be executed until the end of the entire preceding set. This scheduling avoid conflicts between neighbours modifying the same data at the same time.*

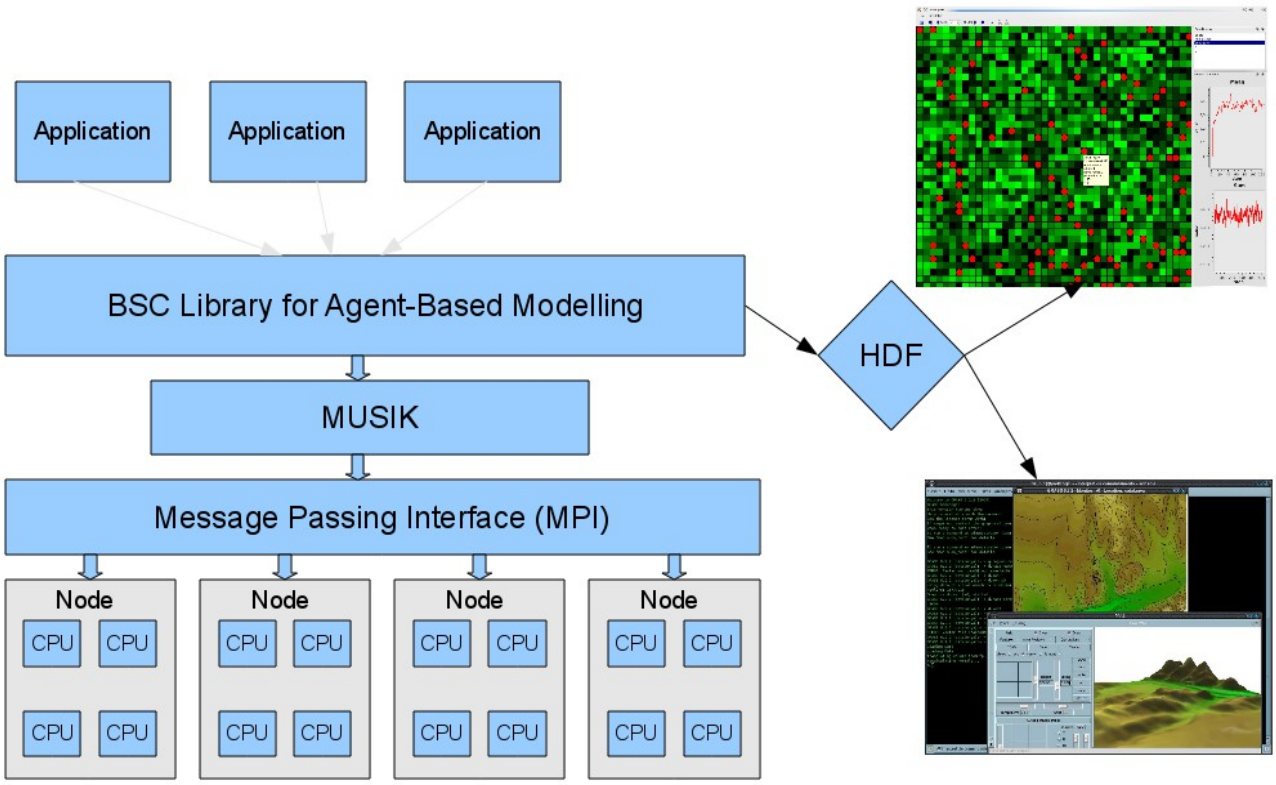

Figure 3: *General infrastructure of the ABM framework*

As this micro-kernel is not focused on agent-based simulation with spatial references, an additional layer has been added. The programming paradigm is object oriented (developed in C++), so a collection of classes has been implemented in order to make easier the reusability of the code. This is an important requirement of the system, because it is designed to be used in several different large-scale simulations.

Another important necessity of the project is the capability to analyse the results after the execution of the simulation, because the usual system of execution in a HPC architecture is the batch mode (applications are added to a queue of processes, executed at disposal of computational time). It requires a lot of data management, as the interest of the researcher will be the entire simulation, and not its final result; this requirement makes mandatory that raster data must be stored for every time step, as well as the state of all the agents.

The solution is provided by HDF5 (Hierarchical Data Format), a data model, library, and file format focused on storing and managing large volumes of data. HDF5 provides support for distributed systems, avoiding bottlenecks during the process of writing data. Moreover GRASS GIS provides tight interaction with HDF format, so the ABM simulator will be capable of transmitting data to this GIS system.

In Figure 3 the design of the entire framework is shown. The different applications the researchers will design use the utilities developed inside the ABM library, that at the same time uses MUSIK in order to manage distributed execution. The results, through the storage in HDF support, will be processed and analysed through the use of an analyser, as well as different Geographical Information Systems.

It is necessary to emphasize that the amount of data stored in a hard disk is extremely challenging, as for each we need to register two different types of entities:

- Raster data. Most simulations need to store data in more than one raster, so the volume of information will be equal to the size of the scenario multiplied by the number of rasters we are storing.

– Agent data. The state of each agent is valuable for the analysis of the simulation, as the variables and chosen actions that the entire group of agents have made is the main source of information for the researcher. This is the reason why we need to store, for each step, the position and state of all the agents. Moreover, agents can be removed or added through the execution time, so the storing mechanism must be flexible enough to deal with these simulations.

5. Performance

A small application was developed in order to test performance of the system. Tjhe main objective was to validate that an increase in the size of a node region and agents number was scaled on a suitable way by the system.

The test established a simulation where two different rasters were stored at every step. Inside this space a big amount of agents were located, specifically designed to expand through the entire virtual world while moving on a random way. The code executed by the agent was deliberately implemented to be time-consuming (searching a position through large regions of the space, etc.), with multiples messages between agents closely positioned in order to improve the validity of the test.

All the simulations were executed with an starting population of 500 agents, and were run through 1000 time steps. In every round the size of the rasters side was doubled, thus multiplying by 4 the number of cells (64x64, 128x128 and 256x256), and annotations were made about time execution, number of agents and number of cpu's. It is important to state that the number of agents was established after the equilibrium of the system (when all the space has been colonized).

Table 1 shows the execution performance with 1 computer node with just 1 cpu, and table 2 shows the same values for 1 computer node with 4 cpu's. In both cases the cost of execution grows exponentially with the number of agents, as the amount of messages between them increases in the same proportion.

Size (Cells)	Seconds	Agents	Steps in 1 Sec.
64x64 (4096)	399	423	2,51
128x128 (16384)	2451	1605	0,41
256x256 (65536)	74581	11500	0,01

Table 1: *Execution results with 1 node/1cpu.*

Size (Cells)	Seconds	Agents	Steps in 1 Sec.
64x64 (4096)	386	430	2,59
128x128 (16384)	2368	1653	0,42
256x256 (65536)	56559	11353	0,02

Table 2: *Execution results with 1 node/4 cpu's.*

Thr same test was made with two executions, where different computer nodes were used. Table 3 shows results for 4 computer nodes (totalling 16 cpu's), and Table 4 for 16 computer nodes (64 cpu's). As we can see the overhead of managing 16 different nodes (without shared memory) only is assumed with large number of agents per node.

Size (Cells)	Seconds	Agents	Steps in 1 Sec.
64x64 (4096)	1066	447	0,94
128x128 (16384)	2389	1640	0,42
256x256 (65536)	18740	11440	0,05

Table 3: *Execution results with 4 nodes/16 cpu's*

Size (Cells)	Seconds	Agents	Steps in 1 Sec.
64x64 (4096)	4652	441	0,21
128x128 (16384)	4738	1631	0,21
256x256 (65536)	6240	11503	0,16

Table 4: *Execution results with 16 nodes/64 cpu's.*

We can see a comparative display of these results at Figure 4.

Finally, it is important to state that main computing cost is related to writing time, as the storage of the entire set of agent states and raster cells generates around 10Gb of data for the larger simulation.

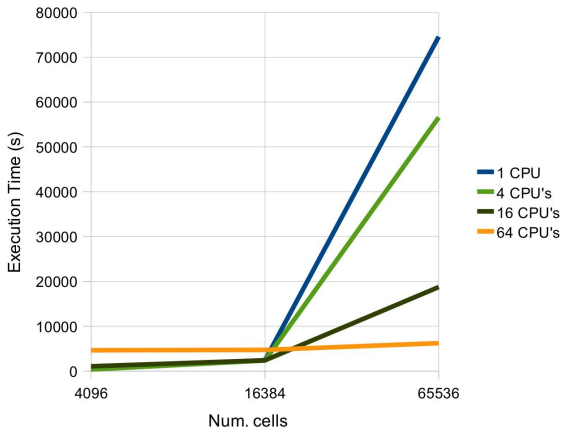

Figure 4: *Comparative analysis of performance test.*

6. Data analysis

An Agent-Based Simulation generates a large volume of data, and it is obvious that the phase where it is analysed and interpreted is the most important step of the entire process. Depending on the chosen methodology it is linked to the generation of new hypothesis (if we use the sources to construct the simulation) or the validation of the research (if, on the contrary, we use the archaeological record to validate a previous hypothesis).

In order to improve the process a different application has been developed, where the researcher will be capable of loading and analysing the information produced during execution.

In Figure 5 this applications is shown while analysing a version of the classic model Sugarscape (EPSTEIN and AXTELL, 1996) specifically designed using the developed framework.

The Graphical User Interface of the analyser consists of three sections. The most important one is the graphical representation of the space where the agents interact. The background of this zone is provided by one of the rasters stored through the simulation. On top of it the agents are shown, and the user can select anyone of them in order to check its position and state at a given moment (specified in the toolbar of the application). On the left side there are two different selection menus, where the researcher can choose the information that he or she needs or wants to see and analyse. The bottom menu displays available rasters, and selecting one of them will show it as the background of the graphical representation of space. On the other hand the top menu displays the different attributes of the agents that the

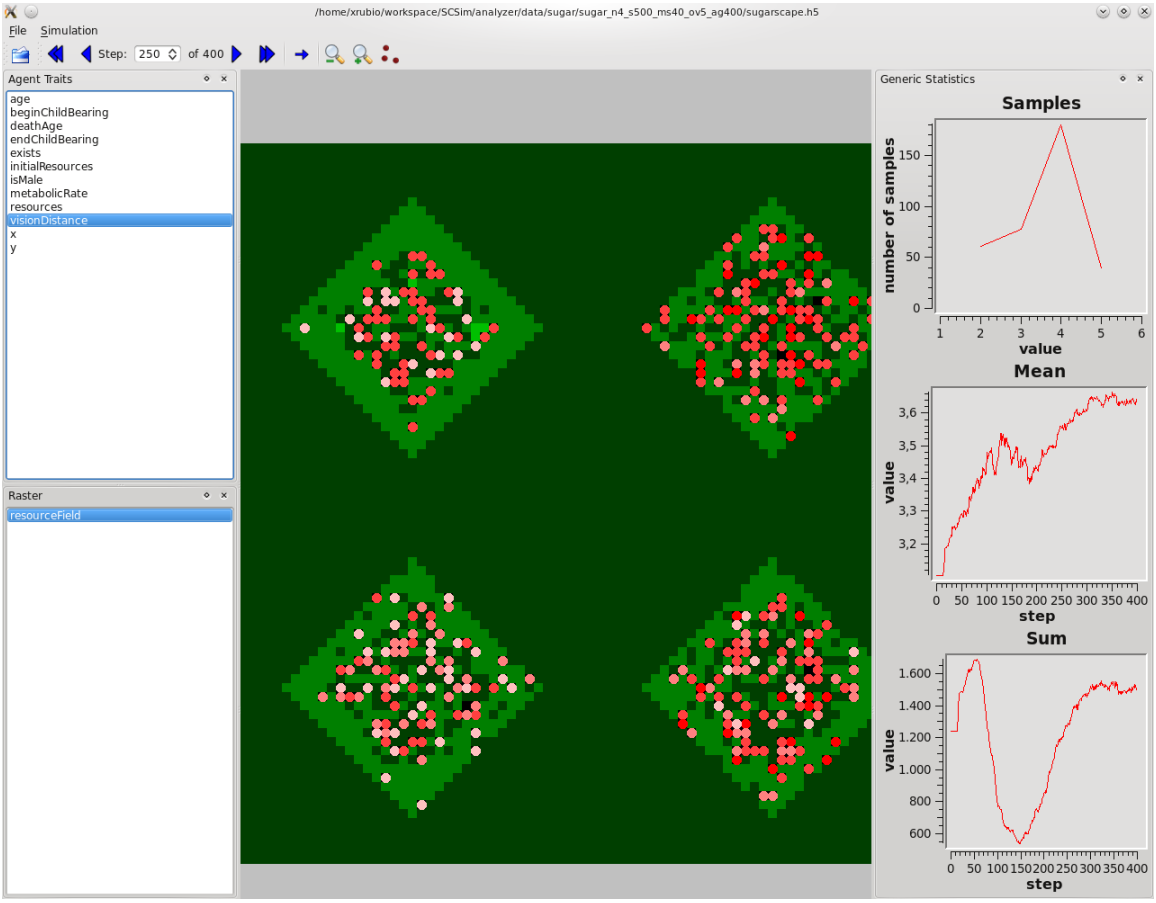

Figure 5: *The application designed to analyse simulation results.*

user stored through the simulation. The user is allowed to select one of them, in order to check the spatial distribution of the values; in the example the attribute containing maximum distance of vision was selected, so we can see in red the maximum value of the entire population. The rest of the agents are drawn choosing a colour between red and white, depending the value of its attribute.

In this way the user can detect spatial patterns of individual attributes, that could be difficult to check without this tool.

Finally, the right side of the application is a menu where basic statistical data is collected from the attribute chosen by the user. This section allows to detect variations on the global state of attributes, as well as possible equilibriums and tendencies affecting the entire agent population in real time while watching simulation execution.

Apart from this analyser, the information through the execution can be displayed on any Geographical Information System, as HDF is an open format that can be loaded by any GIS application. GRASS GIS, for example, is able to open the list of agents position and raster values using GDAL library (Geospatial Data Abstraction Library). On the other hand, GIS information can be loaded through the simulation using this same library.

Conclusions and further work

The combination of the execution framework and the application designed to analyse the results is a starting point from which new ways to use High-Performance computing in social simulation can be developed. Preliminary work is promising, but obviously the platform will need to develop case studies and projects, in order to test its usefulness and validity.

New tools will need to be developed, specially regarding the statistical analysis of the huge volumes of data that can be generated by a single simulation. On the other hand, the system doesn't have any capability to cross information generated by different execution runs. These problems can be address through the integration of a statistical package like R to the system, that will allow the researcher access to more elaborated analysis techniques, as well as the ability to compare multiexecution data and understand, in a more suitable way, the parameter space of the model we are trying to explore.

References

BARCELÓ, J.A., 2009. *Computational Intelligence in Archaeology*. Information Science Reference, London.

DORAN, J, 1999. Prospects for Agent-Based modelling in Archaeology. *Archeologia e Calcolatori* 10, pp.33-44.

EPSTEIN, J.M.; AXTELL, R.L., 1996. *Growing Artificial Societies: Social Science from the Bottom Up*. The MIT Press, 1996.

GILBERT, N., 2008. *Agent-Based Models*. SAGE Publications, California.

LAKE, M.W., 2000. MAGICAL Computer Simulation of Mesolithic Foraging. In Kohler, T.A.; Gumerman, G.J. (ed) *Dynamics in Human and Private Societies*, Oxford University Press, New York, pp.107-143.

LONGLEY, P.A.; GOODCHILD, M.F.; MAQUIRE, D.A.; RHIND, D.W., 2006. *Geographic Information Systems and Science*. John Wiley & Sons, USA.

MINSON, R; THEODOROPOULOS, G.K., 2008. Distributing RePast agent-based simulations with HLA. *Concurrency and Computation: Practice and Experience* 20, pp.1225-1256.

MITHEN, S.; REED, M., 2002. Stepping out: a computer simulation of hominid dispersal from Africa. *Journal of Human Evolution* 43, pp.433-462.

NIKITAS, P.; NIKITA, E., 2005. A study of hominin dispersal out of Africa using computer simulations. *Journal of Human Evolution* 49, pp.602-617.

PERUMALLA, K.S., 2005. Musik – A Micro-Kernel for Parallel/Distributed Simulation Systems. *Proceedings of the Workshop on Principles of Advanced and Distributed Simulation*, Monterey, California, USA.

PERUMALLA, K.S., 2006. Parallel and Distributed Simulation: traditional techniques and recent advances. *Proceedings of the 2006 Winter Simulation Conference*.

STEPHAN, B.S., 2008. Parallel Discrete-Event Simulation of Population Dynamics. *Proceedings of the 2008 Winter Simulation Conference*.

TAKAHASHI, T.; MIZUTA, H., 2006. Efficient Agent-Based Simulation Framework for Multi-Node Supercomputers. *Proceedings of the 2006 Winter Simulation Conference*.