Grid Computing Based Simulations of the Electrical Activity of the Heart

J.M. Alonso, V. Hernández, and G. Moltó

Departamento de Sistemas Informáticos y Computación.
Universidad Politécnica de Valencia. Camino de Vera s/n 46022 Valencia, Spain
{jmalonso,vhernand,gmolto}@dsic.upv.es
Tel. +34963877356, Fax +34963877359

Abstract. Simulation of the electrical activity of the heart by modelization of the action potential propagation is a computational and memory intensive process. In addition many studies, such as the investigation of the ischemia phenomenon, require the execution of lots of parametric simulations, what increases the computational problem by several orders of magnitude. This paper presents the integration of a parallel simulator for the action potential propagation on cardiac tissues into a Grid infrastructure under the Globus Toolkit and InnerGrid middlewares.

1 Introduction

Electrical activity is a key indicator of the state of the heart. Its modellization and simulation allows a better understanding of the electrical behaviour. Cardiac tissue simulations present high computational and memory requirements. Moreover, many cardiac research studies require the execution of a huge amount of parametric simulations. Studies of vulnerable window in ischemia require to vary the time interval between two consecutive stimulus in order to detect the range of values which provoke a reentry, a phenomenon that can derive into heart fibrillation. Besides, to study the effects of late ischemia it is necessary to vary the coupling resistances in all the dimensions of the tissue and observe the evolution of the electrical activity for different anisotropy ratios.

A MPI-based parallel simulation system has been already developed [1] in order to reduce the simulation time on beowulf architectures and allowing the study of larger tissues. Nevertheless, the integration of parallel concurrently executed simulations in a Grid infrastructure seems the key combination to offer a substantial increase in productivity.

2 Grid Computing System Developed

2.1 Portability and Interoperability

To enable portability, the simulation system has been statically linked, so that no external dependencies are required, creating one simulator for the 32-bit Intel

M. Bubak et al. (Eds.): ICCS 2004, LNCS 3036, pp. 482–485, 2004.

architectures and other one for the 64-bit Intel Itanium architectures. Even the MPI library has been introduced into the executable. Besides, all the platform-dependent optimizations have been switched off, such as optimized versions of BLAS and LAPACK, as they may result in potentially executing illegal instructions in the remote machine. This way, it is possible to achieve a self-contained parallel simulation system that can be executed on different Linux platforms.

2.2 Globus Toolkit Developments

We have designed a software layer [2], based upon the Globus Toolkit 2.4 [3] and results from the GridWay project [4]. Basic scheduling support has been added to allocate tasks to nodes on the grid. The total number of available processors is guessed querying the MDS (Monitoring and Discovery Service) server of the execution hosts, assigning a number of simulations to each host proportional to its available computational resources. No attempt is performed to investigate the workload of remote workstations, as they do not offer a local queue system to be queried for free nodes.

Once the scheduler has decided the best computational resource, the stage in phase takes place, compressing the executable and the input data and transferring them to the execution node. A temporary folder is created on the execution machine which will act as a container for the job execution. The data transfer is internally achieved via the Globus GASS (Global Access to Secondary Storage) service, launching a GASS server on the job submission machine. A decompression of the files is performed. Next, the simulation system is executed in parallel integrating, if configured, with the queue manager of the execution node (PBS, LoadLeveler, etc), thus respecting the execution policies of the remote organization. The binary file results obtained are compressed, transferred back to the submission node and saved on the appropriate local folder created for this simulation. Finally, all the temporary created files in the execution node are deleted.

2.3 InnerGrid Developments

InnerGrid [5] is a multi-platform commercial product, that comprises a set of tools that allow to manage an heterogeneous platform of computers. It consists of a server that distributes the pending tasks among the agents, which control each execution. This middleware offers a web-based single point of entry to the Grid, allowing the definition of new tasks, controlling the execution, managing the state of the Grid and accessing the file results of the simulations in a centralized manner. This software implements a fault-tolerance scheme that guarantees the finish of the tasks as long as there are living nodes in the Grid.

InnerGrid does not provide mechanisms for parallel execution of MPI-based applications and so it is restricted to the execution of sequential parametric simulations. In our case, a new module has been created, what represents a definition pattern of all the possible parametric tasks. This module defines the memory and storage space required for the parametric simulations, and allows to

specify a different executable file for each architecture supported by InnerGrid. In addition, the module specifies the command-line arguments of the simulation system that are going to be parametric.

A task is the instantiation of a module, where the user specifies the range of values for the varying parameters indicated in the module, as well as the priority level under which the simulations will be run. InnerGrid built-in scheduler is in charge of allocating the pending tasks to the idle nodes of the Grid.

3 Experimental Results

3.1 Case Study

To analyze the vulnerability to reentry of a cardiac tissue that has been locally affected by ischemia, it is necessary to perform different parametric simulations where the interval between two consecutive stimulus is changed. The vulnerable window for reentry represents the time interval between the two stimulus, in which reentrant activity on the tissue is detected. For a three-dimensional 60x60x60 cell cardiac tissue, a vulnerable window of up to 40 ms has been studied, varying the injection delay of the second stimulus from 1 to 40 ms with respect to the application of the first stimulus and analyzing whether it has resulted in reentry or not. 250 ms of time will be simulated in this example. This results in 40 independent and different parametric simulations of action potential propagation that can be performed simultaneously on a Grid infrastructure.

3.2 Execution Results

The available testbed is composed of two clusters and a workstation. Cluster A has 20 Pentium Xeon 2.0 Ghz biprocessors, with 1 GByte of RAM. Cluster B consists of 12 Pentium III 866 Mhz biprocessors, with 512 MBytes of RAM. An Intel Itanium 2 900 Mhz biprocessor, with 4 GBytes of RAM, has been introduced in this heterogeneous testbed.

InnerGrid does not have an Itanium version yet and therefore only both clusters have this middleware installed. The InnerGrid server was setup in a separate machine, while the agents were run on every node of both clusters. On the other hand, the Globus Toolkit 2.4 has been installed in all the machines.

For each machine, Table 1 shows the execution time and the number of simulations performed, for the case study presented. Parallel simulations are executed with a quarter of the total available processors, a polite policy that allows the execution of several simultaneous simulations. Global execution time corresponds to the slowest machine, 39.16 hours for Globus based executions. On the other hand, simulations ran through InnerGrid needed an 8.5% extra time (42.5 hours) to conclude, executing on nodes of each cluster of the testbed. Sequential execution of the case study in one node of Cluster A required over 563.3 hours, while 9-processor parallel executions in the same cluster, which allows two concurrent simulations, lasted for 45.6 hours.

Table 1. Execution times (in hours) for the simulations of the case study. Numbers in parentheses indicate the number of processors involved in each parallel simulation

		Cluster A	Cluster B	Itanium
Globus	Simulations	24 (9 proc.)	13 (5 proc.)	3 (1 proc.)
	Execution time	38.13	39.16	25.6
InnerGrid	Simulations	28 (1 proc.)	12 (1 proc.)	-
	Execution time	34.3	42.5	-

Large tissues enforce a serious memory requirement and thus, they may not be succesfully executed on sequential platforms, an important handicap for InnerGrid simulations. While InnerGrid seems appropiate to take advantage of idle computers in single-organizational Grids, the Globus Toolkit is focused on running on dedicated resources in different organizational Grids. Therefore, a Globus-based solution is much more appropiate for the cardiac electrical simulation problem, as it offers transparent access to distant computational resources.

4 Conclusions

This paper has presented the integration of a parallel system for the simulation of electrical activity on cardiac tissues into a Globus-based Grid infrastructure. The application features state-of-the-art capabilities such as data compression, self-contained executable and dependencies migration, cross-linux portability, and parallel execution of simulations on the multiprocessor machines. In addition, InnerGrid commercial product has been tested as an easy-to-use alternative middleware to create single-organizational Grids. A new module has been developed, allowing the user to vary several parameters and managing the execution of the different parametric tasks. Having available a parallel simulation system that can be integrated in a Grid infrastructure enables to focus both on speedup, running on a cluster, and productivity, taking advantage of the power of a Grid.

References

1. Alonso, J-M., Ferrero, J-M., Hernández, V., Moltó, G., Monserrat, M., Saiz, J.: High Performance Cardiac Tissue Electrical Activity Simulation on a Parallel Environment. Proc. of the First European HealthGrid Conf., January 16-17. 2003, 84-91
2. Alonso, J-M., Hernández, V., Moltó, G.: Grid Computing Based Simulations of Action Potential Propagation on Cardiac Tissues. Technical Report DSIC-II/05/04. Universidad Politécnica de Valencia (2004)
3. Foster, I., Kesselman, C.: Globus: A Metacomputing Infrastructure Toolkit. The International Journal of Supercomputer Applications and High Performance Computing. 11(2), 115-128
4. Huedo, E., Montero, R-S., Llorente, I-M.: A Framework for Adaptive Execution on Grids. Software Practice and Experience. 2004 (to appear)
5. InnerGrid Nitya Technical Specifications. GridSystems S.A., 2003.