Simulation of Micro Electro-Mechanical Systems (MEMS) on Grid

F. Khalil[1,2 *], B. Miegemolle[1,2], T. Monteil[1,2], H. Aubert[1,2], F. Coccetti[1], and R. Plana[1,2]

1- LAAS-CNRS ; Université de Toulouse ; 7, avenue du Colonel Roche, F-31077 Toulouse, France.
2 - Université de Toulouse ; UPS, INSA, INP.
{fadi.khalil,bernard.miegemolle,thierry.monteil,herve.aubert,
fabio.coccetti,robert.plana}@laas.fr
http://www.laas.fr

Abstract. The future Micro Electro-Mechanical Systems (MEMS) will involve high complex architectures. The design and the Electro-Magnetic simulation of those components require more and more computer memory and computational time. Grid Computing allows to execute all possible configurations of MEMS simulations. In this article, we consider the case of a phase shifter based on N Radio Frequency (RF) MEMS systems which creates 2^N possible configurations. A free open source Transmission Line Matrix (TLM) fullwave Electro-Magnetic solver, called YATPAC (Yet Another TLM Package), is used to simulate the MEMS. The extensions of YATPAC Simulation Package for an execution in grid context, and especially for the French experimental Grid Computing platform Grid'5000, are explained. The execution is done over several sites. A study of performance for deployment of code and data, execution time and recovery of results is commented to show the potential of TLM over a grid.

Key words: : Electromagnetic Simulation, TLM, RF MEMS, Grid Computing, High Performance Computing.

1 Introduction

In the last decade Grid Computing has been already extensively applied for a number of complex problems covering various disciplines such as nuclear physics [1], earth and space observation [2], computational fluid-dynamic (CFD) [3], weather forecast (Meteo-grid), chemistry and biology (Bio-Grid), vibro-acoustics (CEA-grid) (all three developed in the framework of a large European project called EUROGRID [4]). The list could be extended to many others. On the contrary it is only during the last five years that grid technology has been approached by the electromagnetic community under the pressure of the growing challenges in modeling and optimization of electromagnetic devices. Although

these works have paved the way of grid computing in Computational electromagnetic (CEM) they represent valuable but still isolated efforts. This paper deals with experimentation of the TLM method over the Grid'5000 platform, a French large-scale nation-wide infrastructure for Grid Computing and e-Science. It first describes TLM principle and its execution over a grid. Then different measures of performance are done.

2 Related Work

In the last forty years, CEM [5][6] has continuously evolved in both theoretical formulation and methodology and, more recently, in their numerical implementation. Even more emblematic has been the Moore's law race which has yielded enormous achievements in terms of memory and processor performances. However the recent development of wireless communication technologies has evolved with similar or even higher pace, generating problems of a size and complexity that are still out of reach. It is under these circumstances that the availability of cheap computing nodes and continuously growing network speed, enabling factors for the rise of grid technology, has become a natural and attractive resource to overcome present limitations. Although several examples of complex problem solving exploiting Grid Computing can be found in different domain only few deal with electromagnetics. In these latter most of the time the work is entirely carried out by pull of expert which competencies are more toward electromagnetic rather then grid technology oriented. They consist more properly in parallel computing in computer cluster rather than deployment of grid capabilities

Hereby follows an overview of CEM solved by means of parallel/distributed computing. This work may be considered as the precursor of the actual grid computing deployment in the CEM. In 1997 D.C Blake and T. Buter from the Air Force Institute of Technology in Ohio (USA), were the first to propose the solution of an electromagnetic scattering problem by using a grid of massively parallel computing platforms [7]. A group led by Prof. M. Tentzeris of Georgia Institute of Technology in Atlanta (USA) proposed in 2001 the parallel implementation upon a Beowulf cluster of a code for solving partial differential equations (PDE) and applied it to a finite difference time domain technique (FDTD) for simulation of RF packaging devices. A parallel computing implementation of a FDTD has been demonstrated in 2001 from the group of Prof. L. Tarricone of the University of Lecce (Italy) [8]. The same group has been working on the memory requirements optimization by introducing variable mesh size for very large electromagnetic compatibility (EMC) problems as those concerning the interaction between EM sources and humans [9]. Another very recent example of distributed computing using a code based on the TLM method has been demonstrated by the group of Prof. Russer [10]. In this case the code has been tested on homogeneous (5 P4 machine Linux cluster and 4 HP9000 machine UNIX cluster) and heterogeneous (cluster made of 1 P4 Linux and 1 P4 WINXP machine) computer cluster showing portability and scalability for simple test benchmarks.

The results have yielded a speed-up of around 3.7 for the homogeneous clusters. Important recommendations on the scalability and possible criteria for the optimal distribution of the computing tasks have been proposed. The only example of real customized combination of grid computing and CEM is the one proposed by the joint collaboration of Lumerical and WestGrid in Canada [11], even though their field of interest is more in optical and quantum device rather than in RF applications.

The Transmission Line Matrix is used in this article to study MEMS. TLM has reached a very high maturity of method and numerical implementation. Several are indeed the papers documenting the analytical derivation and optimisations [12][13]. Among these the use of graded mesh (finer discretization around singularities in order to resolve the EM field where needed and rougher discretization elsewhere). This has allowed to reduce significantly the memory requirements [12]. Another improvement has been the use of post processing System Identification (SI) technique in order to reduce the simulation time by maintaining the same the accuracy [14].

3 The Micro Electro-Mechanical Systems

3.1 MEMS

MEMS - also referred to as micromechanics or micromachines - can refer to both the technology and the specific devices, which range from a micrometer to a millimeter in size. They are the integration of mechanical elements, sensors, actuators, and electronics on a common substrate through microfabrication technology [15].

MEMS operate using the principles of conventional macromechanic device (such as relay) but miniaturized by Integrated Circuit (IC) compatible technique at micrometer scale. MEMS Physics is multidisciplinary. Although, electrical devices and very few mechanical devices at this scale are common, the scaling down of common mechanical devices found in the macro world has created a research area all its own. They helped the exploitation of new physics domains: Mechanics, Electrostatics, Fluidics, Ionics etc. These systems can have electromagnetic, electrostatic, thermal, magnetostatic, and piezoelectric actuation techniques.

MEMS technology can be implemented using a number of different materials and manufacturing techniques; the choice of which will depend on the device being created and the market sector in which it has to operate.

3.2 Advantages and Applications

Because MEMS devices are manufactured using batch fabrication techniques similar to those used for integrated circuits, unprecedented levels of functionality, reliability, and sophistication can be placed on a small silicon chip at a relatively low cost. In comparison to contemporary state-of-the-art switching devices, MEMS switches exhibit superior electrical performance and low power

consumption on a size scale commensurate with semiconductor devices. MEMS promise to revolutionize nearly every product category. It is an enabling technology allowing the development of smart products, augmenting the computational ability of microelectronics with the perception and control capabilities of microsensors and microactuators and expanding the space of possible designs and applications.

Although other MEMS technology does not match industrial reliability standard, are found in many places, including ink-jet printers, large-screen TVs, airbag deployment systems, and pressure sensors used in car tires and some blood-pressure systems. MEMS are also used in the motion-sensing controller featured in Nintendo's Wii video game system. With its need to reduce the size and weight of objects lifted into space, NASA uses MEMS for microgyroscopes, microthrusters, mass spectrometers and other devices [16], on the contrary RF-MEMS.

3.3 MEMS Phase Shifter

MEMS switches are composed of a thin metal membrane (or beam) which can be electrostatically actuated to the RF line using a DC bias voltage [15]. A shunt capacitive MEMS switch consists of a thin metal membrane "bridge" suspended over the center conductor of a CPW or microstrip line and fixed at both ends to the ground conductors of the CPW line (figure 1).

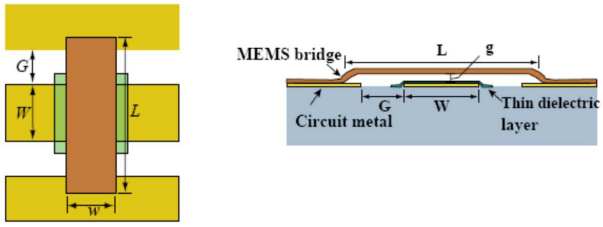

Fig. 1. Shunt capacitive switches

The circuit design proposed here is based on a true-time delay phase shifter which consists of a CPW transmission line loaded periodically with several shunt MEMS capacitors. Thus the circuit can be considered as a synthetic transmission line whose phase velocity can be varied by switching the MEMS capacitive switches up and down [15]. When a DC control bias is applied, for each switch, between the bridge pad and the ground plane, the voltage difference generates a strong electric field underneath the membrane of MEMS switch, which will force the membrane to snap down. The inherent elasticity of metal will help to bounce the membrane back to original state after bias is released. With multiple design parameters to handle, the number of simulations required for the analysis of all possible different configurations may become prohibitive. If we consider the case

of a phase shifter based on N RF-MEMS switches, 2^N configurations are possible. Besides, sometimes the isolation is lower than expected due to non perfect contact of bridge on circuit (Bridge shape, interface roughness and peaks, membrane deformation). To study these limiting factors of the performance many parameters must be considered, leading to more than 2^N configurations to explore. In such a scenario, design and optimization of the analyzed problems is a real challenge.

The phase shifter is accurately modeled using the TLM method, in order to obtain the full EM characterization by computing the S-Parameters.

3.4 Transmission Line Matrix (TLM) Modeling Method

The TLM method of EM modeling is a numerical simulation technique for modeling wave propagation and solving field problems. It was originally used for modeling EM wave propagation, but since it is based on Huygens principle it could be used for modeling any phenomenon which obeys this principle. Researchers showed that TLM can be used to solve the following problems: diffusion problems, vibration, heat transfer, radar, EM compatibility [17].

TLM belongs to the general class of differential time-domain numerical modeling methods. The continuous field functions that satisfy Maxwell's equations are approximated by samples of these functions defined only at discrete points in space and time. The basic approach of the TLM method is to obtain a discrete model (time and space) which is then solved exactly by numerical means. For EM systems, the discrete model is formed by conceptually decomposing the space with a network of transmission-lines in such a way that the voltage and current give information on the electric and magnetic fields.

Another advantage of the TLM method resides in the large amount of information in one single computation. Not only is the impulse response of a structure obtained, yielding, in turn, its response to any excitation, but also the characteristics of the dominant and higher order modes are accessible in the frequency domain through the Fourier transforms. The calculations required are very simple, and the TLM procedure is especially suited for implementation on digital computers and this efficient technique can be implemented not only in single CPU computers but also in parallel computers.

4 YATPAC Simulation Package

YATPAC [18] is a free (open source project licensed under the GNU General Public License) TLM-based full-wave EM simulation package. Various electromagnetic structures can be characterized with YATPAC in time-domain like hollow waveguides, transmission lines, planar microwave circuits and antennas, and more complex systems as 3D interconnects and MEMS indeed.

4.1 Overview of the Simulation Package

The simulation package combines various UNIX/Linux based programs developed and used for solving and visualizing three dimensional field problems using the TLM method. A simulation is performed by three steps: preprocessing, simulation, and postprocessing (figure 2).

Preprocessing. At this step, the design of the structure is prepared for the simulator kernel. A file containing the information of the engineered structure (`<filename>.gds`) and a file containing information on the discretization of the simulation object (`<filename>.laydef`) are prepared. Then, an output file (`I3D.<filename>`) containing all needed input data for the simulator is generated from the two input files.

Simulation. *yatsim* (Yet Another TLM Simulator) represents the core computational engine. As output two files are generated. The first contains all 6 field components of each cell in the simulation area at every time step. The second file contains certain current and voltage-signals in time domain, computed by integral paths over E- and H-field in defined planes in the input files.

Postprocessing. The visualization of the discretized structure and the computed EM field is done with special adapted visualization tools. The computed time-domain signals in EH-file and calculated S- and Z- Parameters can be regarded and processed by graph plotting and math computing programs like Grace, Scilab, Octave or MATLAB.

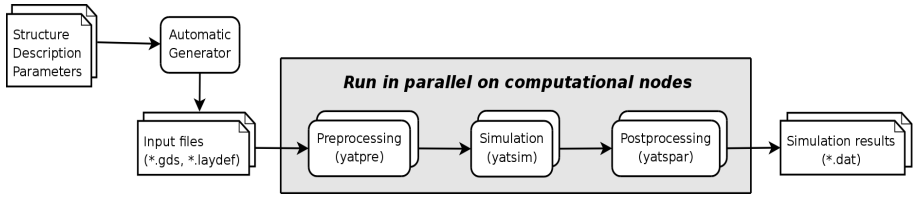

Fig. 2. Simulation workflow

4.2 Parallel YATPAC over the Grid

Circuit design traditionally involves a trial-and-error process. Series of prototypes are built and tested while iterating to an optimized design. In this new proposed approach, a wide range of design parameters are evaluated in a single analysis run with the goal of exploring the entire design space and selecting the optimized design without need for the normal iterative process.

As mentioned above, in the preprocessing step, the engineered structure and the discretization of each model must be prepared. To generate automatically the different prototypes with its proper specification, we have implemented a C++ code. This generator needs to be run once, and all the simulator input files are ready-to-use.

Preprocessing, simulations and postprocessing steps take long computational time, and must be repeated for each generated input file. Using a Grid infrastructure enables to parallelize this work (figure 2) by reducing the number of simulations treated by each node.

Once all of the models are analyzed, response data are automatically collected for the postprocessing step. The simulation executions on the Grid will be detailed in the next section.

5 Execution on Grid'5000

5.1 Presentation of the Grid Platform

The Grid'5000 platform [19] has been used to execute TLM application. The objective of this platform is to provide with a highly reconfigurable, controllable and monitorable experimental Grid that is geographically distributed over 9 sites in France. These sites are linked by RENATER [20], the French national network for educational and research purposes (figure 3). This platform enables researchers to experiment algorithms and tools related to different fields of Grid Computing, such as middlewares, parallel programming, networking, etc.

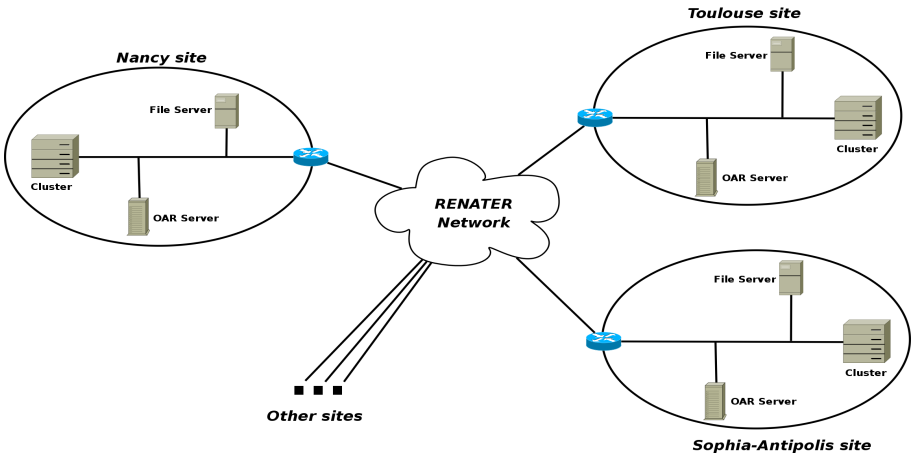

Fig. 3. Simplified schematic of Grid'5000 architecture

The objective of this paper is to point out the interest of using a Grid platform to perform TLM simulations. A set of simulations will be executed on Grid'5000

computational nodes, and a performance analysis will be carried out to show the benefit of using such an infrastructure.

5.2 OAR and Kadeploy

Grid'5000 allows the administration and the management of its resources with two tools: *OAR* and *Kadeploy*.

OAR [21] is a resource manager (batch scheduler) for clusters. It allows cluster users to submit their applications, enabling them to reserve nodes either in an interactive or a batch mode. The Grid'5000 sites are composed of one or more clusters, and each site is independent from each other. They have their own NFS file server, and an *OAR* server is set up on each of them. This server is responsible of the management of the corresponding resources. A meta-scheduler, *OAR* Grid, enables to submit grid applications that need resources distributed over several sites. The OAR Grid meta-scheduler is in fact an alias that delegates the node reservation to each concerned local *OAR* scheduler.

Kadeploy [22] is used to deploy a customized image of a Linux kernel over the computational nodes. This allows to deploy, without any effort, a working environment on each node used during an experiment. The nodes are then rebooted in order to run the deployed operating system. In our case, a custom Linux image based on an Ubuntu distribution has been created. An installation of the YATPAC Simulation Package and all its dependencies has been included in this image, in order to get this software suite present on each computational node used during our experiments.

5.3 Experiment Workflow

The simulation execution can be divided into the following five steps:

Initialization Phase. During this phase, the simulation input files are generated on each Grid'5000 sites, using the C++ prototypes generator (section 4.2), and then stored in the local NFS server. The generated files (`PS.laydef` files), particularly their number and their content, depend on the simulations that have to be performed. This is done independently but simultaneously on each site.

Deployment of the Customized Linux Image. The *Kadeploy* tool is used to reboot all the computational nodes in order to run the Ubuntu-based operating system that bundles the YATPAC Simulation Package. This operation is done for all nodes used during the experiment. Note that, since there is no file sharing between the different sites of Grid'5000, a copy of the Linux image is stored in each local NFS server of each of them, in order to be used by *Kadeploy*.

Deployment of the Simulation Input Files. When all computational nodes have been rebooted with a Linux image that contains YATPAC, the simulation input files generated in step 1 and stored in the local file servers, must be deployed on each of them. Indeed, even if these files are accessible from every node on the NFS server, it is better to make a local copy of them in order to optimize their access during the simulations, avoiding simulation slowdowns due to multiple networking accesses. The files that are transferred to a computational node are filtered in order to copy only the files corresponding to the simulations that will be run on this node. Two files are required to perform a simulation: `PS.laydef` (specific to each simulation) and the corresponding `PS.gds`.

Execution of the Simulations. All simulations are executed on the computational nodes. Each node executes a subset of simulations, corresponding to a limited number of input files. All nodes execute the same number of simulations, since spreading them equally over the nodes leads to optimize the duration of this phase. The simulations are independent, so they can be run in parallel. They use the input files generated in previous steps, and produce new result files. The following commands enable to run the simulation using YATPAC:

```
yatpre -t t -i PS.laydef PS.gds I3D.Temp
yatsim Temp
yatpre -t ref -i PS.laydef PS.gds I3D.Ref
yatsim Ref
yatpre -t C -i PS.laydef PS.gds I3D.Dev
yatsim Dev
yatpre -t C -i PS.laydef -M PS.gds TLMSTRUCT
yatspar -v -h 8.1 -l 3e-6 -f 25e7 -N 200 2:EH.Ref 8:EH.Dev S_Parameters_PS.dat
```

Retrieval of Result Files. The generated result files (`S_Parameters_PS.dat`), present on each computational node, must be retrieved and stored on the NFS servers. Indeed, when the experiments end, all nodes are rebooted to run a default operating system, so that other users are enabled to use them for their own experiments; this operation deletes all files stored on each node.

6 Experiments

6.1 Context

This paper considers an experiment composed of 192 TLM simulations. This experiment will be executed several times, for different numbers of computational nodes to use. The number of nodes used will be increased gradually, in order to measure the impact of this metric on the performance of each step of the simulations.

The experiments use up to 3 clusters (the ones of Nancy, Toulouse and Sophia-Antipolis), with a maximum of 64 nodes per cluster. The cluster characteristics are the following:

- **Cluster of Nancy:** Intel Xeon 5110, 1.6GHz

– **Cluster of Toulouse:** AMD Opteron 2218, 2.6GHz
– **Cluster of Sophia-Antipolis:** AMD Opteron 2218, 2.6GHz

The number of nodes used is gradually increased. First, 4 nodes of the cluster in Nancy are used. When the number of nodes reaches 64, a second cluster is used (the one of Toulouse); the 64 nodes of the first cluster are still used, and the number of nodes used on the second cluster is gradually increased to reach 64 as well. Finally, the third cluster (Nancy one) is included when more than 128 nodes are required.

6.2 Initialization Phase

This phase occurs at the beginning of the experiments. It is executed on one host per cluster (typically the *OAR* server node), and generated files are stored on the NFS server of each cluster. When several clusters are used, the execution on each cluster is done in parallel at the same time. Thus, the duration of this phase is almost constant, and depends not on the number of nodes used, but only on the load of *OAR* server, NFS server and network. The experiments show that this phase lasts between 1.5 and 2 seconds.

6.3 Deployment of the Customized Linux Image over the Nodes

Before launching the simulations, the Linux image that contains YATPAC is deployed over the different nodes. The figure 4 shows the duration of this phase for the different numbers of nodes used.

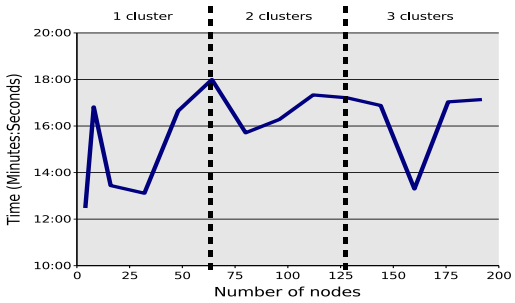

Fig. 4. Duration of the image deployment phase

The duration of this phase is between 12 and 18 minutes, but it is not accurately predictable. The deployment is performed in parallel on each node, so that the number of nodes used would not affect the performances of the system during this phase. Actually, this is not so simple because the Linux image to deploy is stored in the NFS server, and the its performances may depend on the number of requests, and thus on the number of nodes used. The activity of other

grid users can also have an impact on the performances of the NFS server, and so on the duration of the deployment of the image.

6.4 Copy of the Input Files on the Nodes

The next step is to deploy the input files required by each simulation on the corresponding nodes. The duration of this phase is represented by figure 5.

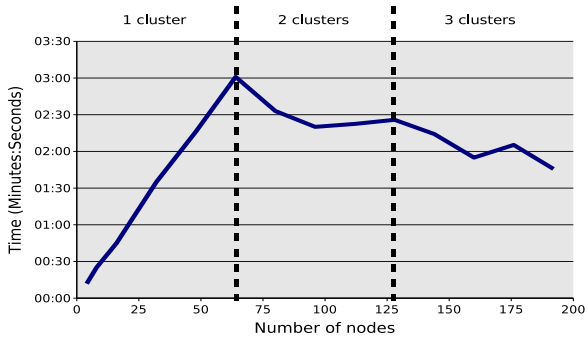

Fig. 5. Duration of the input files deployment phase

The profile of the measured performance points out two different cases. First, when only one cluster is used, the duration of this phase depends directly on the number of nodes used, in a linear way. The copy of the input files is performed by the *OAR* server node, which sends these files to each computational node. Thus, the duration of this phase increase with the number of nodes used. Note that both network load and NFS server performances have an impact on this duration.

If several clusters are used, this assertion is not verified anymore. For example, using two clusters involves using 64 nodes on the first one, and gradually increase the number of used nodes on the second one. Thus, in this case, there is a cluster for which the deployment of the input files must be performed for 64 nodes. The duration of this period is almost constant, and corresponds to the time needed for the copy of the input files on the 64 computational nodes of the cluster.

So when the number of hosts increases, it is interesting to distribute them over different sites, in order to bound the duration of this phase.

6.5 Execution of the Simulations

This phase is the longest one, since it corresponds to the execution of the YATPAC tools, mainly the *yatsim* the simulator, in order to explore all possible configurations of the simulated problem.

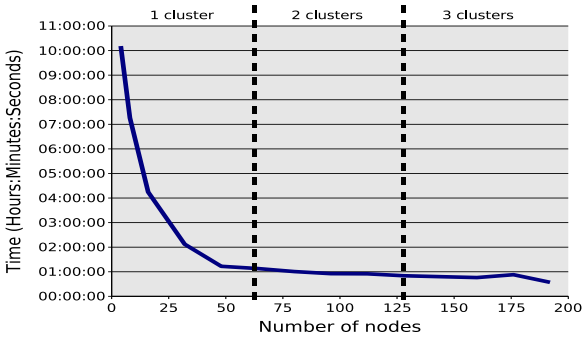

Fig. 6. Duration of the simulation execution phase

The duration profile of this phase (figure 6) presents a good regularity. The execution time of the simulations depends on the number of nodes that are used, or more precisely on the number of simulations that are executed on each node. Moreover, the experiments have been designed in order to obtain an homogeneous distribution of the simulations over the computational nodes.

Since the experiments are composed of 192 simulations, the mean number of simulations per node is: $\frac{192}{Number\ of\ nodes}$. This expression matches the hyperbolic appearance of the measured execution times. For example, if only 4 nodes are used, each of them executes 48 simulations. More than 10 hours are then needed to complete this phase. If 192 nodes are used (1 simulation per node), the execution of the simulations only takes 34 minutes.

This result points out the benefit from using a large number of computational nodes for running TLM simulations.

6.6 Results Recovery

The last step of the experiments is to retrieve the results from the computational nodes and to store them on the local NFS servers.

The figure 7 shows that the duration of this phase decrease with the number of nodes used. Some irregularities are present, since the performance of the results recovery depends on the NFS servers and the network load, and so on the other users activities.

7 Conclusion and Future Work

The experiments carried out in this paper show the interest of using Grid environment for electromagnetic simulations. We considered the case of a phase shifter based on N Radio Frequency MEMS switches simulated using TLM method. This system generates more than 2^N possible configurations that have to be explored. For this purpose, the French national Grid Computing platform, Grid'5000, was used.

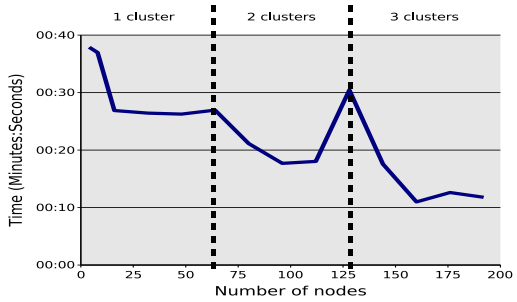

Fig. 7. Duration of the result recovery phase

The simulations were divided into 5 main parts, namely: Initialization (i.e. Generation of the Simulations Input Files), Deployment of a Customized Linux operating system that bundles YATPAC Simulation Package, Deployment of the Input Files over the Computational Nodes, Execution of the Simulations, and Results Recovery. These simulations were run in both monocluster and multicluster contexts. The tests of performances we carried out show that most of time is spent in the fourth phase, i.e. during the execution of simulations that relies on YATPAC. Using a large number of nodes enables to decrease significantly the execution time of the simulations. Distributing the simulations on several sites does not have an impact on the simulations execution time, since simulations are independent and do not communicate between each other. However, it enables to optimize the input files deployment by decreasing the network bandwidth consumption and file server requests for each site.

Another benefit of using a Computational Grid is the fault tolerance. Indeed, distributing the simulations over a large number of nodes enables to limit the number of simulations that fails if a node breaks down.

The overall execution time of the simulations can be improved by considering the characteristics of the nodes in order to take in account the resource heterogeneity to better distribute the simulations. Indeed, having the same number of simulations per node is not optimal, since some nodes can be less powerful than others. These nodes slow down the overall duration of the execution phase. The idea is so to adapt the number of simulations to the power of the nodes to have them finish the work at the same time.

Acknowledgments. The authors wish to acknowledge the National Research Agency (ANR) for support of MEG Project, and AMICOM the European Network of Excellence on RF-MST and RF-MEMS funded by the European Commission under the 6th Framework Program.

References

1. LHC Computing Grid (LCG) project, `http://www.cern.ch/lcg/`

2. Marchetti, P. G., Beco., S.: SpaceGRID: how to foster and Earth Observation GRID. In: Euroweb 2002 Conference, Oxford (2002)
3. The IST FlowGrid Project, http://www.unizar.es/flowgrid
4. The IST EUROGRID Project, http://www.eurogrid.org
5. Johns, P. B., Beurle, R. L.: Numerical solution of 2-dimensional scattering problems using a transmission-line matrix. In: Proc. IEEE 118 (1971), no. 9, 1203–1208
6. Yee, K. S.: Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media. IEEE Transactions on Antennas and Propagation, AP-14, 4, 302–307 (1966)
7. Douglas, C.: Advances in Time-Domain Electromagnetic Simulation Capabilities Through the Use of Overset Grids and Massively Parallel Computing. Blake Storming Media (1997)
8. Catarinucc, L., Palazzari, P., Tarricone, L.: Parallel simulation of radio-base antennas on massively parallel systems. Parallel and Distributed Processing Symposium., Proceedings 15th International 23-27 April 2001
9. Tarricone, L., Esposito, A.: Grid Computing For Electromagnetics. Artech House Publishers (2004)
10. Lorenz, P., Vagner Vital, J., Biscontini, B., Russer, P.: TLM-G – A Grid-Enabled Time-Domain Transmission-Line-Matrix System for the Analysis of Complex Electromagnetic Structures. IEEE Transactions on Microwave Theory and Techniques, vol. 53, no. 11, 3631–3637 (2005)
11. The Lumerical/WestGrid partnership, http://www.lumerical.com/westgrid_overview.php
12. Hoefer, W.: The transmission line matrix (TLM) method. In T. Itoh: Numerical Techniques for Microwave and Millimeter Wave Passive Structure. John Wiley and Sons, New York (1989)
13. Russer, P.: The Transmission Line Matrix Method.In: Applied Computational Electromagnetics, NATO ASI Series. Springer, Berlin (2000)
14. Coccetti, F.: Application of System Identification (SI) to Full-Wave Time Domain Characterization of Microwave and Millimeterwave Passive Structures. PhD Thesis, TUM Munich (2004)
15. Rebeiz, G. M.: RF MEMS: Theory, Design, and Technology. Wiley-Interscience (2002)
16. NASA GSFC Electrical Engineering Division, http://eed.gsfc.nasa.gov/562/SA_MEMs.htm
17. Christopoulos, C.: The Transmission-Line Modeling Method. Wiley-IEEE Press (1996)
18. The YATPAC Website, http://www.yatpac.org
19. Cappello, F., Desprez, F., Dayde, M., Jeannot, E., Jegou, Y., Lanteri, S., Melab, N., Namyst, R., Primet, P., Richard, O., Caron, E., Leduc, J., Mornet, G.: Grid'5000: A Large Scale, Reconfigurable, Controlable and Monitorable Grid Platform. In: Proceedings of the 6th IEEE/ACM International Workshop on Grid Computing, Seattle (2005)
20. The National Telecommunication network for Technology Education and Research (RENATER) Website, http://www.renater.fr/
21. The OAR Scheduler and Resource Manager Website, http://oar.imag.fr/
22. The Kadeploy Deployment System Website, http://kadeploy.imag.fr/