Deployment of Computational Electromagnetics Applications On Large Scale Architectures

Carlos-Jaime BARRIOS-HERNÁNDEZ *(1,2)*, Fadi KHALIL *(3)*, Yves DENNEULIN *(2)*,
Hervé AUBERT *(3)*, Fabio COCCETTI *(3)* and Robert PLANA *(3)*

1- LIG Laboratory Mescal Team, Montbonnot-St Martin, France
2- I3S UNSA Laboratory Rainbow Team, Sophia Antipolis, France,
3- LAAS-CNRS; Université de Toulouse, Toulouse, France

firstname.lastname@imag.fr, ***firstname.lastname@laas.fr***

Abstract

Large Scale Architectures, as Grid Computing, provide resources allowing to handle complex problems, vast collections of data storage, specific processing and collaborative interaction between distributed communities. Nowadays, there are several scientific applications that runs on Grid Computing architectures. However, in most of these cases, applications need to be adapted in order to better exploit the capacities and opportunities of scalability, heterogeneity and pervasive characteristics of Grid Computing.

Large Scale Computational Electromagnetics problems place computational limitations in terms of hardware capacities. Parallel approaches proposed to face the demand of Computational Electromagnetics (CEM) aim to provide solutions to tackle high degree of complexity of the application and interaction with these large scale architectures. In this work, a description of the adaptation and implementation of Computational Electromagnetics solutions in a grid computing enabled environment is presented in terms of scientific results and performance associated with architectural opportunities.

1 Introduction

Actually, fundamental problems in science and technology imply complexity, large set of data, high processing need and interdisciplinary analysis. Grid computing includes all the above characteristics and enables the development of large scientific applications to handle them. Grid-enabled applications benefit of more computational resources that are not generally available at a single local site[1], which make feasible the execution of larger applications that consume large resources and represent a high cost, in architectural terms.

Nevertheless, the design and implementation of efficiency grid-aware applications is typically a time-consuming task. The development of Grid computing applications adds complexity in building parallel and distributed applications too, because it represents new paradigms and new levels of interaction between applications, infrastructures and users.

Nowadays, several scientific problems are treated with programs that run in Grid-enabled environments. Fluid dynamics, astrophysics, biology, communications, health, among others, are subject of parallel and distributed applications on Grid-enabled environments.

Precisely, a main motivation to propose a Grid to support scientific research can be traced since the origins of Internet technology. This scientific inquiry with Grid computing involves technology evolution, new collaborative modalities of interaction and social challenges. The possibilities to make data-intensive science, simulation-based science, remote access to experimental devices among many others allow to perform science like *e-science* [8].

Interesting cases in scientific applications in which handling these problems is necessary to deal with are those involving scale changes. The growth of the scale ratio involves the use of more elements, and consequently, more resources. But the use of ad-

---

[1] A local site should be a set of compute infrastructures placed in a specific geographic location, such as a laboratory, supercomputing center or university.

ditional resources, in Grid-enabled environments, requires synchronisation, data coherence and planning without losing performance. In another words, multiscale treatment implies scheduling.

Several real-world electromagnetic problems like scattering, radiation, waveguiding etc, are not analytically calculable, for the multitude of irregular geometries designed and used. The inability to derive closed form solutions of Maxwell's equations under various constitutive relations of media, and boundary conditions, is overcome by computational numerical techniques. This makes Computational Electromagnetics (CEM), an important field in the design, and modeling of antenna, radar, satellite and other such communication systems, nanophotonic devices and high speed silicon electronics, medical imaging, cellphone antenna design, among other applications. It has continuously evolved in both theoretical formulation and methodology and, more recently, in their numerical implementation.

Nevertheless, the multi-scale aspect is very important while modeling such structures. And more where it exists a wide diversity of scales that implies a numerical resolution with a great computational effort associated for reaching the convergence of the numerical results. In a computational context, the motivation of this work is to guarantee the execution of the accurate implemented algorithm to reach an acceptable solution without increasing the time to obtain a result, compared this time to the time cost for the execution of the simplest scale.

In this work, Grid Computing capacities are investigated in order to enhance performances of numerical electromagnetic solvers to CEM issues. For this purpose, the major challenge to address is to adapt the middleware in order to guarantee an optimal scheduling, deployment and execution of the solvers in the selected local elements of the Grid Computing platform.

2 Computing Electromagnetic Modeling Techniques

Applied electromagnetics is playing a pivotal role in the development of advanced technologies that address society's challenges across a broad spectrum of communications, computing, materials processing, and sensing applications.

In electromagnetism, Maxwell's equations, which are a set of four partial differential equations that describe the properties of the electric and magnetic fields and relate them to their sources, charge density and current density, must be solved. Usually, these equations are computed on an enormous number of points representing the discretization of the physical domain of the studied structure. Naturally, such discretization of the computational space consumes computer memory, and solving the equations takes a long time. Large scale CEM problems place computational limitations in terms of memory space, and CPU time on the computer. Generally CEM problems, are run on supercomputers, high performance clusters, vector processors and parallel computer.

A number of different numerical techniques for solving electromagnetic problems are available. Each numerical technique is well-suited for the analysis of a particular type of problem. The numerical technique used by a particular EM analysis program plays a significant role in determining what kinds of problems the program will be able to analyze.

In this work, two numerical electromagnetic are concerned: the Transmision Line Matrix (TLM) Modeling Method [7] and the Scale Changing Technique (SCT) Method [2]. While applying the TLM, a volumetric time-domain modeling method, the entire region of the analysis is gridded. An advantage of the TLM method resides in the large amount of information in one single computation. But when the computation domain is too large and/or lot of precision is demanded, usually the number of unknowns to compute explodes, which renders these computations impossible on one computer.

The second modeling method, named SCT, is an original and efficient numerical technique for electromagnetic modelling of modern planar multiscale structures, like multi-band frequency-selective surfaces, active or passive reflect arrays, or self-similar (pre-fractal) planar objects. This method is known to be fast but due to the actual increase of scale ratio and the complexity of the structures, the parametric studies of convergence needed by this technique for electromagnetic analysis may become prohibitive in terms of computer memory and time.

3 SCT Algorithm

The Scale Changing Technique consists of decomposing the electromagnetic studied structure into different domains with respect with the scale ratio between them[2]. The strategy consists of artificially introducing intermediate scale levels such that two suc-

---

[2] Examples of multi-scale structures are given by multiband frequency-selective surfaces, active or passive reflectarrays, or self-similar (pre-fractal) planar objects.

cessive levels differ from a one (or two) decade(s). The SCT is based on the cascade multimodal Scale Changing Networks (SCN), each network modeling the electromagnetic coupling between two succesive scale levels [2].

The concept of SCT is close to the physical equivalence principle. Instead of describing the whole structure, one can characterize located electromagnetic effects that then will be integrated in the larger scale. In each sub-domain the higher-order modes are used for the accurate representation of the electromagnetic field local variation while lower-order modes are used for coupling the various scale levels. The transition from one scale to another looks like a discontinuity between two waveguides of different section. The integral equation method using entire domain trial functions enables determination of the N-port network associated with this discontinuity. The combination of all scales is then modeled by the cascade of elementary networks that are analogous at each scale [2] [16].

Obviously, the multiplicity of the scales present in a structure is a problem that SCT aims to handle. The multiscale nature of a structure is used to break up this one into sub-structures. Taking into account the entire problem corresponds to the cascading of these different sub-structures, each sub-structure characterize the transition of a scale towards another. This subdivision allows a modularity which largely eases the implementation of parametric studies. The high flexibility of the approach associated with the advantages of the Integral Equations Formulations renders this original approach powerful and rapid [2] [16].

Figure 1 presents the different modules M1, M2, M3, M4 and M5 related with 5 compute process. The modules allow relating two scale levels with different order of magnitude. The cascade of these modules with M6, M7, and M8 modules, allows crossing the scale from the lowest to the highest scale.

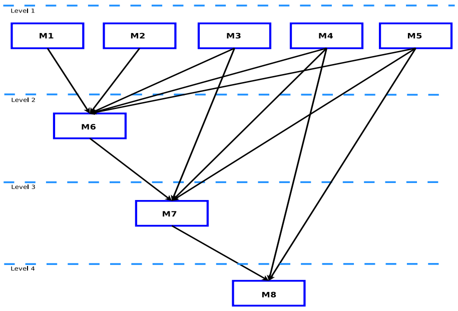

Figure 1. *Compute Flow of SCT Modules and Distribution in Levels*

Making a sequential description of the SCT technique from the Figure 1, we can propose the next pseudocode:

```
Read General Input Parameters
 Compute M1 M2 M3 M4 M5
 Reduce M1 M2 M3 M4 M5 Output in Level_1 Input
Read Level_1 Input
 Compute M6
 Reduce M6 M3 M4 M5 Output in Level_2 Input
Read Level_2 Input
 Compute M7
 Reduce M7 M4 M5 Output in Level_3 Input
Read Level_3 Input
 Compute M8
Print General Output
```

The computation codes originally written for such structures are sequential. But since these modules are independent, consequently, they can be computed in parallel. The modules are distributed by levels in the available compute resources. The distribution of the compute process is made using the opportunities of the batch scheduler in the platform, taking advantage of non-local resources, resources on a wide area network, or even the Internet when local compute resources are scarce, as explained in the Section 4.

Note that for simplicity of explanation, only a structure with limited scale changes is represented here. Usually, modern studied structures have a big scale ratio between the biggest and the small dimension which may explode the number of levels and modules to treat such problems.

4 Scheduling and Deployment: SCT Case

In Grid Computing, scheduling is important because it implies the coordination of users, jobs, task and resources. The scheduling process should support the processing of different algorithms at different times and to different queues (normally both). Scheduling should be able to interact with the resource manager in use [12].

Many tasks of scheduling may be dedicated to a resources manager or they may be treated by the resources manager and the scheduler together. This last possibility allows the generation of the *batch scheduler* to provide a exploitation of the resources in agreement with jobs processing. Obviously, a hierarchy of tasks and jobs exists. The different levels may be organized in behalf of the profit of the resources or by the order of the task in the job queue, in according with the politics for the use of the common Grid computing resources.

Our proposition corresponds to a general *hierarchical* solution for the scheduling in behalf of the parallelization opportunities present in a specific platform testbed: Grid'5000 (G5K) [3] [11], and the clusters contained in this national wide Grid computing platform.

Specifically, the computational electromagnetic method concerned is the Scale Changing Technique (SCT) [2] that allows to analyze the planar complex structures as a combination of many scale [2]. The scale levels interaction is sensitive; when the number of levels grows, the complexity grows too (and also the volume of information).

The SCT method is associated with some numerical solvers contained in a SCT application set that consists of a collection of the Matlab©[15] codes. The codes are organized in compute modules for specific compute jobs. The jobs are collected in levels. The different levels are associated with the interaction between the computational modules, which may be distributed on different nodes to take advantage of the opportunities of task distribution in each compute level.

Our approach for these numerical solutions can allow the deployment of the applications in the selected nodes of the Grid platform with a specific characteristics of software environment to guarantee scalable and pervasive execution. The use of necessary resources demands management of these distributed resources and good interaction with the middleware and operating system. In the case of G5K, the management is made by OAR[3] [4] [17] and the deployment by KaDeploy [10] [14].

A plug-in between the batch scheduler and the specific SCT codes is proposed to allow the distributed execution in the Grid computing platform. The analysis of these methods determines common opportunities, nevertheless they are different in computing. The plug-in is a set of scripts (written in batch and Taktuk [6] [18]) that allows the allocation, distribution and execution of tasks taking advantage of the OAR characteristics.

The possibilities given by TakTuk allows to write simple portable programs. Basically, Taktuk provides an efficient work distribution on heterogeneous platforms thanks to an adaptive work-stealing algorithm. The work-stealing technique allows to divide a procedure execution efficiently among multiple processors. Work-stealing is used in different forms: as adaptive algorithms or reactive algorithms implemented in different systems or programming languages such as Kaapi [9] [13] or Cilk [1] [5].

The processor maintains a stack on which it places each frame that it has to suspend in order to handle a procedure call. If it is executing the Modules of the level 1, and encounters a recursive call to level 1 Modules, it will save Module's Levels 1 states before to run the next level modules , including its variables and where the code suspended execution, and put that state on the stack. It will not take a suspended state off the stack and resume execution until the procedure call that caused the suspension, and any procedures called in turn by that procedure, have all been fully executed.

On the other hand, it is important to say that for the deployment function of the software environment for the execution of SCT programs, Kadeploy is used without any modification.

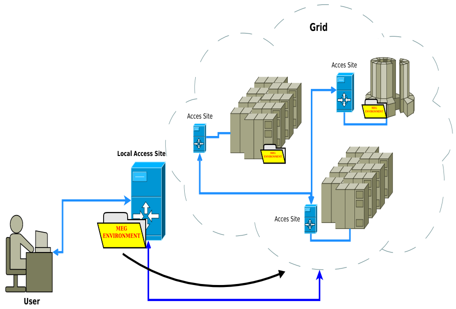

Figure 2. *MEG Environment Deployment in Grid 5000 Platform*

Figure 2 shows the deployment of the *MEG Environment* (represented by the yellow folder). The MEG Environment is an operating system image with the set of libraries, codes and programs to SCT compute. The MEG Environment is deployed in the reserved nodes of the specific selected clusters. The user interact with the Grid environment with a simple ssh connexion, via a local access site or frontend server. When the user select the nodes in the infrastructure, that contains the necessary hardware (mainly defined by the type of processing cores and number of nodes), and the time to work upon Grid'5000 (or Grid system, represented in the Figure 2 as a cloud), the user should be interact in two modes: active and passive. In the active mode, the interaction with the platform is in real time, contrary, the passive mode is programmed.

Actually, the work on one site is managed and scheduled by OAR. When the compute occurs in several sites, the OARGrid API is used. The interconnection among the sites is granted by OARSSH, a SSH API for OAR.

---

[3] And an API named OARGRID.

In practice, the user should select different nodes and sites (is not necessary to use the local site itself). The possibility of use is determined by the availability of resources at moment of the interaction (in the case of the active mode) or the availability of resources for the start programmed timed of job.

Observing the Figures 1 and 3, we can notice the different levels and the relation among them (levels are limited by discontinued blue lines). A data dependence exists between them. For example, after the compute of the first five modules (see the Figure 1) the output is the input of the next level (inputs and outputs are represented like rhomboids with the word ”input” or ”output” respectively), to compute the module 6. Same for the next modules.

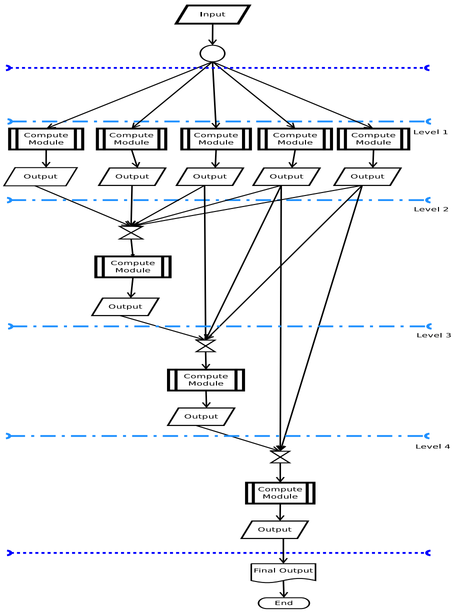

Figure 3. *SCT Scheduling Flow and Distribution in Levels*

The workflow start with a general input file of parameters. These parameters are placed/distributed into the compute modules (the process of placement and distribution is shown as a circle and the compute modules are represented by boxes into the Figure 3) to reach the first work level. In this level, the five first modules are distributed in the available nodes in equal quantities. For example, if there are 40 nodes reserved and available to compute, the compute work by compute module is made by 8 nodes.

The outputs of each one of the modules are concentrated to made the second level of compute [4], also distributed in all nodes available. In the third level shown in the Figure 3, we use the outputs of the level 2 and the outputs of the compute modules 2, 3, 4 and 5. In the next level four, we use the output of the level three and the outputs of the compute modules 4 and 5. In the same way that for the last level, the compute module is distributed in the available nodes. Finally, we can get the final outputs.

Note that the relation will be expressed like a hierarchical interaction with the platform resources. Figure 3 shows the computer workflow in terms of inputs, outputs and allocation of resources. Each module of the first level is allocated in function of the reserved resources. After that, in function of the outputs, a new re-allocation is possible, but in this case, in function of the first available resources. The first allocation of resources is really an activity of the resources manager. Obviously, the connections between the outputs, new inputs (manually or not), implying new computing levels. Then, the complexity and size of the levels related with the scale necessary to work, should be growing or decreasing.

As these tasks are associated with the management and use of the platform, this approach allows to observe the properties of the tool to build the layer between the batch scheduler such a plug-in.

Then, in accordance with the necessity of operation, the goals of the plug-in should be addressed as:

- Simplicity of use : The scientific user needs an easy interaction, mainly with the compute modules, the access and interaction with the Grid computing platform should be transparent.
- Integration with OAR and Kadeploy : The SCT modules are addressed with a mainly interaction with Grid’5000 infrastructure. Then the plug-in should be integrated with OAR, OARGRID API and Kaa-tools.
- Minimal workload added by the plug-in execution: It is recommendable that the plugin do not add significant computational cost compared to the cost of the SCT code itself.
- Automatic allocation of modules in nodes by level: As shown in Figures 1, **??** and **??**, the deployment and allocation of the MEG Environment, and the compute modules all distributed with efficiency in the available resources. The allocation will be automatic in accordance with the compute needs and availability.

There are requirements like fault tolerance and

---

[4] The concentration process is represented with the two inverse triangles symbol.

transparent integration with other batch schedulers that are not yet handled. However, in the case of fault tolerance, it is possible to analyze different factors that should be considered as a fault related with the SCT. For example, failures of hardware or connection are treated by the batch scheduler, in this case, OAR.

5 First Results

In the SCT application case, we took advantage of the distribution opportunities like data migration possibilities. Every module of the SCT application is treated like a black box. The perspective of processing efficiency, in accordance with the theory is between 15% and 30%. However, in view of the scalability possibilities of the problem, the efficiency is really unknown.

Nevertheless, it is necessary to evaluate the performance of the solution proposed, from the practical use. To handle this, two points are necessary: a performance evaluation in terms of HPC efficiency and the observation of the pertinence of the results from the specialists perspective.

The objective of the tests shown in this article is to evaluate the performance of the plugin from an infrastructure-tool relation. Then, we can observe the transfer cost of the image environment and also the efficiency of the distribution of the compute modules in the platform in terms of makespan. The makespan allows to observe the time difference between the start and finish of a sequence of job of tasks. Precisely, as is show after in this paper, one our interests is to see the compute time given a number defined of nodes to compute the SCT codes. The observation of this time provides information about the gain or loss of time of the application when the different modules run in the Grid computing platform.

The results presented here are obtained with the use of almost two platforms of the Grid'5000 infrastructure. The first one is the Toulouse site, with two clusters: Pastel and Violette. These two clusters allow to use ≈ 426 nodes of Sun Fire architecture, with core processors of 2.2 GHz and 2.8 GHz.

The other site is the Bordeaux site with ≈ 645 nodes of IBM and Dell Technology, which includes the use of AMD Opteron and Intel Xeon architectures. Clearly, the behavior of the implementation depends on the network performance, and consequently implies an analysis of the data transfer. Thus, we propose two different analysis: first, an analysis of data transfer of the image that implies an important set of bytes, and second, an analysis of the normal transfer of data, that implies a smaller quantity of bytes per data.

As explained before, in order to run the computation environment, deploying the image that contains the SCT computational codes on several nodes is necessary. Each image contains between 300 MB and 550 MB approximately. This deployment in Grid'5000 platform is made with Kadeploy of Ka-Tools.

A data transfer of 300 MB or 550 MB is a critical process, because this type of data transfer implies high bandwidth use, massive and parallel data transfer. In this type of transfer, the network resources implied are exploited in concurrence with others. Also, the capacity of the network should be saturated and when this transfer occurs in heterogeneous nodes, a high cost is associated with the network capacity and the use at same time of the deployment image.

Considering the worst case, a deployment process of 550MB on one node occurs during 250 seconds, where an important percentage of the time corresponds to transmission of the server in the node. Obviously, when the number of nodes implied in deployment is increased in the same cluster, the deployment time grows and the maximal use capacity of the network is reached. After this point, the behavior become stable.

If we observe the bandwidth in the same transfer, as is presented in the Figure 4, is possible to know the use of the capacity in terms of transfer. Same that the experiences made with the benchmark tool, the bandwidth grows to a point for two (2) links and after remains stable in interval for the two measures between 4.2 MB/s and 4.5 MB/s.

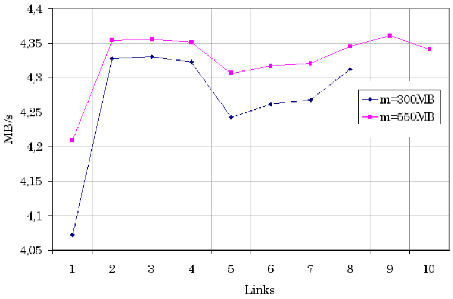

Figure 4. *Bandwidth in MEG Environment Image Deployment*

The experience described here shows the deployment using 10 links between 2 remote platforms (in this case between the *Pastel* cluster at Toulouse site and the *Bordemer* cluster in the site of Bordeaux, transfer direction from Toulouse to Bordeaux). The latency average between the two clusters at the moment of the transfer is the 75 microseconds.

The second point of view, implies only the data transfer of files of smaller quantity of bytes (files from 0 KB to 15 KB). The data transfer in this case for a same latency is very small (≈ 25 and 100 microseconds). Then, this situation suggest to analyze the processing time.

Using the same local platforms, the makespan performance for a typical execution of the SCT implementation that implies 1024 different configurations is investigated. Figure 5 shows the theoretical prediction for this execution compared to the experimental measurements.

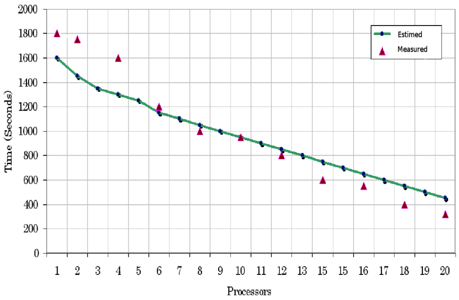

Figure 5. *Makespan Performance of the SCT Implementation*

In Figure 5, the prediction is represented by the green line and the measures for the red triangles. The confidence interval remains, and the values of the measures start more hight that the theoretical green line. The regularity of the curves shows good agreement with the theory. It is interesting to observe how the slope of the values increase when 4 processors are used. On the other hand, the reduction of the time comparing with the execution on one processor is the almost 45%. A prediction of the time with 50 processors in two remote local platforms has been performed.

In Figure 6, it is possible to observe that the reduction of time remains $\approx 45\%$. However the curve (in red and blue) reach the minimal possible value after the use of 46 processors where the estimation for this type of computation is the 10 seconds. Beyond this point, it becomes stable. This situation has been completely predicted, because there is always a limit for the number of processors contributing to the efficiency of the computation with respect to the grain of the problem. Moreover, increasing the number of processors should normally increase the total time of computation due to the communication between them.

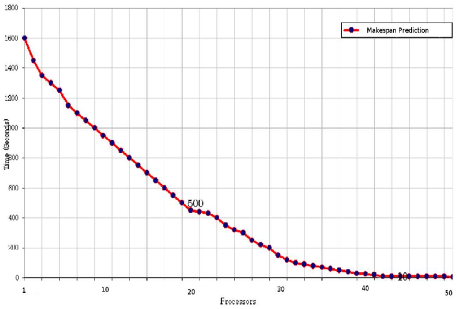

Figure 6. *Makespan Prediction of the* SCT *Implementation*

The Scale Changing Technique has proven to be an original and efficient numerical technique for electromagnetic modeling of modern planar multiscale structures. Using this modelling method, accurate numerical results are obtained with a substantial reduction in computer time and memory compared to direct full-wave electromagnetic analysis, due to the intrinsic scalability of this method.

The obtained results have confirmed the effectiveness of the parallel distributed approach compared to sequential computing. This approach shows very good computation performance while keeping the same accuracy.

Note that most of the time, when small geometry changes occur, only one or few SCT compute modules need to be recalculated, which is not the case of other numerical tools [2].

Besides, this method is very promising for optimizing circuit with multiple design parameters to handle and for the global electromagnetic simulation of multi-scale or/and over-sized structures.

6 Conclusions and Further Work

The adaptation and implementation of scientific applications, initially designed to run on simple computers (or monolithic supercomputers), for Grid computing environments is a big challenge. However the same HPC uses and assuming that the parallelization opportunities can be located in one HPC local platform, the distribution of the processing in another local platform implies heterogeneity and scalability, without taking into account the possibilities of a strong data dependency. These aspects should be treated from the infrastructure or computer science perspective without break the scientific efficiency of an application or implemented method.

Besides, scientific applications could be developed by scientists which are not necessarily computer specialists. The characteristics of each solution implemented in a computer program correspond to a numerical method with a high sensibility to change. Thus, treatment of a scientific application like a *black box* could be the efficient strategy to avoid modifying the numerical solution and leaving the problem of the efficient distribution and execution in the grid computing platform to a low level.

Our solution is based on the implementation of the SCT method without changes, using a plug-in to deploy the compute environment allows to be useful distribution opportunities and the accuracy of the method proposed by the electromagnetic specialists. Nowadays, there are several scientific applications that solve efficiently a lot of specific problems in their domains. A preoccupation of the scientific community is to run these applications into Grid computing infrastructures that optimize automatically the parallelism and distribution opportunities given for the infrastructure and middleware. However the specific solution proposed for the SCT, the observations in performance evaluation contribute to understand the pertinence and measure the efficiency of the development of plugins to provide parallel and distributed execution into Grid computing platforms.

On the other hand, the knowledge of the performance of the scientific application in terms of execution is necessary, and for this reason we have made an evaluation of the efficiency and accuracy in two phases: scientific results and performance evaluation from a computer science perspective. In this paper we are concentrated in the computer science perspective.

The first results presented in this paper, confirm an acceptable performance of the strategy implemented in accord with the efficiency of the shared resources. The decrease of the compute time in 45% suggest a good profit of the scalability opportunities. In the current work, new experiences have been executed growing the complexity to have an use necessity of almost two times more of processors (100-200 processors). However, not only the quantity of processors is an important parameter, the specific features of the processors too. For this reason, other work associated is made, that implies the observation of the heterogeneity between the characteristics of the platforms, to use efficiently the available resources.

7 Acknowledgement

The authors wish to acknowledge the National Research Agency (ANR) for support of MEG Project (ANR-06-BLAN-0006)[5].

References

[1] Agrawal, K., He, Y. and Leiserson C. E. *An Empirical Evaluation of Work Stealing with Parallelism Feedback.* Proceedings of the International Conference on Distributed Computing Systems (ICDCS) Lisboa, Portugal, July, 2006.

[2] Aubert., H *The Concept of Scale-Changing Network in the Global Electromagnetic Simulation of Complex Structures.* Progress Electromagnetics Research B, vol. 16, page 127-154, 2009.

[3] Bolze, R., Cappello, F., Caron, E., Dayde, M., Desprez, F., Jeannot, E., Jegou, Y., Lanteri, S. Leduc, J. Melab, N., Mornet, G., Namyst, R., Primet, P., Quetier, B., Richard, O., Talbi, E. and Touche I. *Grid'5000: A Large Scale and Highly Reconfigurable Experimental Grid Testbed.* International Journal of High Performance Computing Applications, 20(4):481-494, November 2006.

[4] Capit, N., Da Costa, G., Georgiou, Y., Huard, G., Martin, C., Mounie, G., Neyron, P. and Richard, O. *A Batch Scheduler with High Level Components*Proceedings of IEEE International Symposium on Cluster Computing and the Grid, 2005, CCGrid 2005. Volume 2, 9-12 May 2005 Page(s):776 - 783 Vol. 2 , Cardiff, United Kingdom 2005.

[5] *MIT- Cilk Project* http://supertech.csail.mit.edu/cilk/ Project Site.

[6] Claudel, B., Huard, G. and Richard, O. *Tak-Tuk, Adaptive Deployment of Remote Executions*. Proceedings of International ACM Symposium on High Performance Distributed Computing, Munich, Germany, 2009.

[7] Christopoulos, C.: The Transmission-Line Modeling Method. Wiley-IEEE Press, 1996.

[8] Foster, I. and Kesselman, C. (Editors) *The Grid 2: Blueprint for a New Computing Infrastructure.* Morgan Kaufmann Publishers, Elsevier Inc. San Francisco, CA, United Statesof America 2004.

---

[5] For more information about the MEG Project, please visit: http://mescal.imag.fr/membres/yves.denneulin/MEG/ .

[9] Gautier, T., Besseron, X. and Pigeon, L. *KAAPI: A thread scheduling runtime system for data flow computations on cluster of multi-processors.* Proceedings of the 2007 international workshop on Parallel symbolic computation, London, Ontario, Canada 2007.

[10] Georgiou, Y.; Leduc, J.; Videau, B.; Peyrard, J.; Richard, O. *A tool for environment deployment in clusters and light grids*. Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International Volume , Issue , 25-29 April 2006.

[11] *Grid'5000 Project* http://www.grid5000.fr Project Site.

[12] Iqbal, S., Gupta, R. and Fang, Y. *Planning Considerations for Job Scheduling in HPC Clusters*. Dell Power Solutions. Internet Document. 2005.

[13] *KAAPI Project.* http://kaapi.gforge.inria.fr/ Project Site.

[14] *Kadeploy: Scalable Deployment System for High Performance Computing*. http://kadeploy.imag.fr/ Project Site.

[15] *Matlab*. http://www.mathworks.com/ Mathworks Inc. Internet Site.

[16] Khalil, F., Rashid, A., Aubert, H., Coccetti, F., Plana, R., Barrios-Hernandez, C.J, and Denneulin. Y. *Application of scale changing technique-grid computing to the electromagnetic simulation of reflectarrays.* 2009 IEEE International Symposium on Antennas and Propagation, Charleston, South Carolina, June 1-6, 2009.

[17] *OAR: Resource Managment System for High Performance Computing*. http://oar.imag.fr/ Project Site.

[18] *TakTuk: Adaptive Large Scale Remote Executions Deployment.* http://taktuk.gforge.inria.fr/ Project Site.