An Intelligent Platform for Effective Management of Time-consuming Electromagnetic Simulation Problems

Andreas P. Kapsalis[1], **Panagiotis K. Gkonis**[1], **Constantinos L. Zekios**[2], **Dimitra I. Kaklamani**[1], **Iakovos S. Venieris**[1], **and George A. Kyriakou**[2]

[1]Intelligent Communications and Broadband Networks Laboratory
School of Electrical and Computer Engineering, National Technical University of Athens
9 Heroon Polytechneioy Str., Zografou, Athens, Greece
[2]Microwaves Lab, Department of Electrical and Computer Engineering
Democritus University of Thrace, Xanthi, Greece

Abstract— Modern simulation applications that carry out large scale iterative processes, such as Monte-Carlo simulations, or manipulate large data structures, tend to be extremely time-consuming due to the shortage of computational resources or the inherent nature of the simulation itself. Typical simulations can take up to several days to complete on conventional systems, while high-end supercomputers can be cost-prohibitive. Therefore, the need for effective parallelization of software execution and resource management is more imperative. The goal of this paper is to present a fully-distributed platform that enables software simulations to be executed within user-acceptable time periods, by predicting the resource requirements of each simulation. In this context, the platform analyzes files that contain historical data about past executions of the particular simulation. Processor and memory utilization, overall execution time, level of parallelization and distributed execution are some of the information collected and used by an efficient training algorithm, in order to determine the optimal amount of resources to be allocated in a particular simulation. The training algorithm applies several machine-learning techniques such as multi-linear regression in order to efficiently predict the resource vector that will satisfy the user requirements.

1. INTRODUCTION

A scientific application tends to be extremely resource heavy and time-consuming. A typical use case of such an application could require from several hours to several days to yield the desired results. That is usually caused due to the nature of the application itself, as simulations incorporate algorithms that require the execution of huge iterative operations or the management of large data structures. Data and computationally intensive applications are often executed in environments that lack the appropriate amount of resources that will allow the simulation to complete at an acceptable time period. A traditional solution to this rather common problem has been the incorporation of High Performance Computing Grids. High performance computing datacenters offer resources to users and thus provide a platform for executing their applications. However the nature of Computing Grids apply several limitations to the experimenters who many times lack the technical know-how to overcome them. Lately Cloud Computing has emerged as a strong alternative that removes these limitations and offers a robust and flexible environment for the execution of large scale applications. Users can deploy their applications in isolated virtual environments (Virtual Machines), which are able to scale statically or dynamically according to the needs of the infrastructure.

In this paper we present an intelligent platform which communicates with an underlying Cloud Infrastructure and provisions resources in the form of Virtual Machines for executing large scale time-consuming applications. The platform incorporates machine learning algorithms that analyze previous historical execution data and decide the resource vector that will be allocated for a particular application. That decision considers both execution time and infrastructure load and thus both under and over provisioning can be prevented. Also, the multi-tenancy of Clouds allows for simultaneous executions of different applications. The isolation of Virtual Machines ensures that even when multiple applications are executed at the same time, no running process will be interrupted of affected by another. Similarly, user data are also protected as they are not visible to other users or tenants.

In addition, the platform performs analysis on historical execution data and decides the optimal resource vector that needs to be allocated in each experiment. Users are able to submit new experiments from a single access point by using a unified Web based application that provides

easy management, monitoring and result retrieval for each experiment. The rest of the paper is organized as follows; in Section 2 we present some notable relevant research work. In Section 3 we analyze in more detail the advantages of Cloud Computing, while in Section 4 we describe the electromagnetic applications that are being supported by the platform. In Section 5 we describe the general platform architecture and in Section 6 we present the Application Profiling component. Finally, in the last Section we provide some concluding notes and present some interesting ideas for future research.

2. RELATED WORK

Scientific workflows and applications have been tested in Cloud infrastructures in the past. In [1] multiple Cloud providers and Grids are tested for scientific workflows. Among various results, some important points that need to be noticed is that resources are easier to be acquired when using Cloud infrastructures, and that Clouds were more predictable than Grids and showed more uniform performance. The study in [2] presents a deadline-driven platform for execution scientific applications, and again shows that Cloud Computing is becoming the model of choice as many obstacles such as the integration of legacy systems can be easily overcome by using virtualized environments. Finally, in the presented analysis of challenges and issues of deploying scientific applications in Cloud environments in [6], researchers conclude that Clouds are becoming more attractive as they offer scalability, on-demand resource allocation, better utilization and user experience.

3. CLOUD INFRASTRUCTURE

The proposed platform was specifically developed as a middleware that takes advantage of the power of Cloud Infrastructures. The platform architecturally lies in the PaaS (Platform as a Service) tier. Cloud infrastructures were chosen over traditional grids and supercomputers for various reasons. One important reason was to remove compatibility issues and limitations. In [3], the special computational Grid that is offered as a service imposes several operational limitations to the users. If an experimenter wants to submit his application for testing he has to meet certain requirements, such as operating system and programming language compatibility or development patterns (client-server model). That means that a researcher that has already developed his application on a special environment might have to re-develop it (especially when using low-level programming languages whose libraries depend on the operating system). On the contrary, the nature of Clouds impose no such barriers. Virtual instances (or machines) can be spawned by images that contain various environments, operating systems and libraries. That flexibility allows experimenters to deploy their applications as is, as the platform administrators will be able to create the relevant image to support the target application. In addition, most modern Cloud implementations provide built-in monitoring, orchestration, failsafe and data protection mechanisms, all of which are powerful tools and provide administrators the means to guarantee both for the availability of the platform and the protection of sensitive user data.

4. ELECTROMAGNECTIC AND MIMO SYSTEMS APPLICATIONS

The proposed architecture has been evaluated for the Eigenanalysis of large open and periodic electromagnetic structures as well as Monte-Carlo simulations for multiple antennas (MIMO) at both transmission ends. Both problems, due to their nature, are computationally and time intensive [7, 8]. Initially, the extensive study of open and periodic electromagnetic structures leads to a huge eigenvalue problem. In the first case, due to the increased number of possible eigenvalues, large matrices should be formulated and inversed. In the case of MIMO simulations, since multiple users and time instances are considered, in order to reduce feedback burden, Principal Component Analysis (PCA) is employed on the received data matrix of each active user. Therefore, it becomes apparent that computational complexity increases linearly with the number of users.

5. PLATFORM ARCHITECTURE

The platform is designed in a fully distributed fashion. The components are implemented using a peer-to-peer approach rather than the client-server model. In more detail the platform utilizes the actor model, a hierarchical model which comprises of one or more independent modules called "Actors". This modular approach guarantees asynchronous communication between the components, as well concurrency and parallelism whenever needed. Each module or "Actor" is responsible for a certain number of operations within the platform. Also, each module can be deployed separately

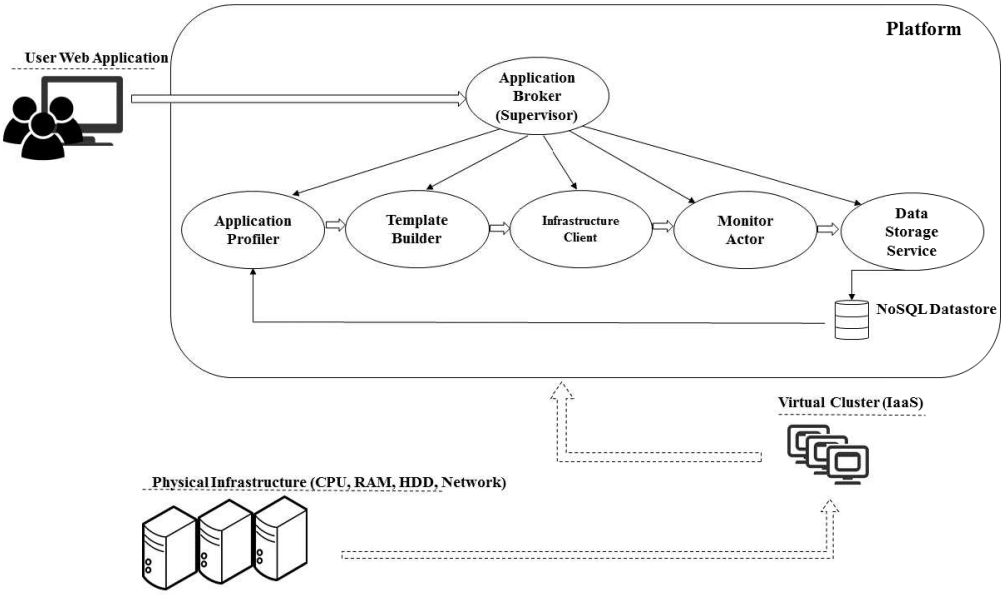

Figure 1: Overall architecture of the proposed platform.

within the network and this characteristic in combination with the elasticity of the Cloud can provide failsafe mechanisms in cases of failure or slow responsiveness.

As mentioned in previous sections, the platform supports certain types of electromagnetic applications. For that reason, a unique image has been prepared for application. Each image has been created taking into account the various requirements each application has in terms of operating system, libraries and software. In that way the experimenter is free to develop his application in the most convenient manner, and thus the platform has the ability to meet the users' requirements. Fig. 1 shows the overall architecture of the platform within an IaaS.

5.1. User Web Application

This is the front-end of the platform. It provides a friendly interface to the user, where he can initiate, manage and monitor experiments. The user enters the Web base application by entering his unique credentials that are provided by the platform administrator. The user can initiate new experiments by choosing the type of application he want to execute. Each application has its own configuration tab, where parameters and input data can be easily set, chosen and uploaded. The user can review all of his experiments (active, successful or failed) through the Experiments Page, where he can be informed on the progress of his currently active simulations and even download the results of completed operations. Status updates regarding active user experiments are pushed into the Web based application in a real-time fashion, allowing users and administrators to react as soon as possible to various events and alerts.

5.2. Application Broker

The Application Broker acts as the supervising entity in the platform. It receives requests from users and delegates those requests to the underlying components for processing. The Application Broker also pushes updates that receives from workers to the Web Application. For every new experiment request a single Application Broker instance is spawned that in turn spawns its own workers (or Actors). Consequently, each experiment has its own group of dedicated components that are responsible for its correct execution.

5.3. Application Profiler

This component holds a very important role in the experiment procedure as it implements algorithms that decide the resource profile that needs to be allocated to an application in order to be executed either in the requested amount of time by the user or according to the resource availability of the underlying Cloud Infrastructure. Details about the algorithms that are incorporated can be found in Section 6. Once the Application Profiler outputs the resource vector (CPU, RAM, HDD, Network) it modifies the Experiment request and forwards it to the Template Builder along with the appropriate operation signal.

5.4. Template Builder

The Template Builder component accepts an experiment request and builds the corresponding JSON template, that in turn will be communicated to the underlying Cloud Infrastructure so that the required resources will be allocated for the execution of the experiment. Once the Template Builder succeeds, it sends the Template along with the appropriate operation signal to the Infrastructure Client.

5.5. Infrastructure Client

The Infrastructure Client, is responsible for communicating with the underlying Cloud Infrastructure using the offered public APIs. A typical operation that this component is responsible for, is the posting of the Experiment Template to the Infrastructure in order to create the requested cluster of Virtual Machines and allocate them for the execution of the application. Once the Communicator receives the appropriate ACK from the Infrastructure, meaning that the VM cluster is up and running, it then spawns monitor workers that perform periodic checks on that specific cluster of VMs.

5.6. Monitor Actor

The Monitor Actor is spawned by the Infrastructure Client. It performs periodic checks by requesting monitoring data from the Monitor Server of the Cloud Infrastructure. Monitor specific data are typically data that include information about CPU and RAM utilization that is affected by the execution of certain applications. The Monitor Actor is also responsible for alerting its supervisor in cases of alert situations, i.e., when a application is completed or fails, or when a VM become unavailable due to high usage of its resources. All monitor data that are retrieved from the Monitor Server are sent to the Data Storage Service.

5.7. Data Storage Service

The Data Storage Service is responsible for storing monitor and other related data into a persistent datastore. For perfomance reasons the chosen datastore is a NoSQL document database, allowing for faster read-write times and batch processing of large amounts of data.

6. APPLICATION PROFILER COMPONENT

One of the most important aspects of the platform is that it incorporates algorithms that enable the automatic resource allocation to applications. As stated above it is crucial that applications have the proper amount of resources that will enable them to complete in a reasonable time period. However, in the same time the platform must ensure that the underlying infrastructure is not under or over utilized. That means that user time related requirements are of high priority provided that sufficient amount of resources are indeed available, as in the same time multiple users could have submitted their own experiments. Luckily, due to the nature of the Cloud that provides computational resources in the form of Virtual Machines, there is no need to worry about processes that might be executed in the same environment or operating platform.

Due to the nonlinearity of the problem the traditional Linear Regression model was not sufficient. Different kinds of resources (CPU, RAM) affect in various ways the execution of a application. Since the nature of the described applications in Section 4 is computationally intensive, the execution time is expected to be smaller whenever applications are executed within a multi-core VM. On the contrary, memory increase does not seem to contribute much in the decrease of the execution time. However, the relation is still not linear as adding more CPUs to a VM will eventually increase the overhead of communication between multiple threads.

For that reason, the prediction component incorporates Regression Trees using the M5P algorithm which in turn is based on the M5 algorithm presented by Quinlan in [4]. The algorithm organizes the feature vector for each training example into a decision tree and performs multiple Linear Regression runs. The feature with the weakest regression coefficient is removed from the tree and the process is repeated until no further improvement in the estimated prediction errors is achieved. The performance of the proposed approach is tested in [5] and shows satisfying results. In addition, the measured output in our case is only the execution time as the platform does not impose complex SLAs (Service Level Agreement) or cost penalties. This is partly due to the fact that resources that are allocated to an experiment are immediately available to the resource pool once the experiment completes.

7. CONCLUSION AND FUTURE WORK

In this paper we described the use of a distributed platform that provides a middleware for the efficient management of time-consuming electromagnetic applications. The proposed platform shows that Cloud Infrastructures can support computationally intensive applications and workflows as they apply mechanisms and characteristics that traditional grids or supercomputers dont. In addition, the use of machine learning algorithms aid in effective resource management and provide experiments sufficient resources in an automated way. The decrease of execution time helps researchers by allowing them to benchmark their solutions and algorithms multiple times in an acceptable time period.

In the process of further optimizing and evolving the proposed platform, we plan to implement a series of steps. Distributed execution of applications that can lead to further decrease of execution times, is already being studied. Also, apart from the available resources, an experiment execution time is dependent on specific simulation parameters that are chosen by the experimenter, i.e., the number of simulated users in a PCA experiment. In its optimal state, the prediction component should include user defined parameters in the feature vector, as different values can determine how computationally intensive each experiment is. Finally, one interesting idea for future research could be the integration of workflow management tools to the platform that will allow the execution of different applications in the form of separate tasks.

ACKNOWLEDGMENT

This research has been co-financed by the European Union (European Social Fund ESF) and Greek national funds through the Operational Program Education and Lifelong Learning of the National Strategic Reference Framework (NSRF)-Research Funding Program: THALIS DUTH, Design Techniques for Digitally Controlled RF-Microwave Structures Appropriate for Software Defined Cognitive Radio (MIS 379521).

REFERENCES

1. Juve, G., M. Rynge, E. Deelman, J. S. Vockler, and G. B. Berriman, "Comparing futuregrid, Amazon EC2, and open science grid for scientific workflows," *Computing in Science & Engineering*, Vol. 15, No. 4, Apr. 2013.
2. Vecchiola, C., R. N. Calheiros, D. Karunamoorthy, and R. Buyya, "Deadline-driven provisioning of resources for scientific applications in hybrid clouds with Aneka," *Future Generation Computer Systems*, Vol. 28, No. 1, 58–65 Jan. 2012.
3. Open Science Grid, Accessed, Mar. 2015, http://www.opensciencegrid.org/about/.
4. Quinlan, R. J., "Learning with continuous classes," *Proc. of the 5th Australian Joint Conference on Artificial Intelligence*, 1992.
5. Xiong, P., Y. Chi, S. Zhuy, H. J. Moon, C. Pu, and H. Hacigumus, "Intelligent management of virtualized resources for database systems in cloud environment," *Proc. of 2011 IEEE 27th International Conference on Data Engineering (ICDE)*, 87–98, Hannover, Apr. 2011.
6. Zhao, Y., X. Fei, I. Raicu, and S. Lu, "Opportunities and challenges in running scientific workflows on the cloud," *Proc. of CyberC 2011*, 455–462, Beijing, Oct. 2011.
7. Zekios, C. L., e al., "Analytical and numerical eigenanalysis of electromagnetic structures: A review," *European Microwave Week*, Paris, Sep. 2015, Accepted.
8. Gkonis, P. K., et al., "On the performance of spatial multiplexing in MIMO-WCDMA networks with principal component analysis at the reception," *EuCAP 2015*, Lisbon, Apr. 2015.