Volume 4, Issue 2, February 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

High-performance Execution of Scientific Multi-Physics Coupled Applications in a Private Cloud

Mohamed-K HUSSEIN
Tabuk University, Saudia Arabia.
Faculty of Computers and Informatics,
Suez Canal University, Egypt.

Mohamed-H MOUSA
Faculty of Computers & Informatics
Suez Canal University
Egypt

Abstract—Scientists are increasingly interested in simulating complex chemical systems and physical phenomenon by coupling distinct individual physical models of different phenomena in order to provide an accurate simulation of the system under study. In computational terms, coupled model applications are distributed applications, in which the components of the application are synchronized via exchange of information during runtime. Such applications are long-running parallel applications, which require high-performance distributed environment in order to run. Emerging Cloud computing technology offers cheap and large-scale high-performance computing environment for the benefit of scientific applications. However, coupled model applications require complex communication/computation pattern, such as optimized MPI for communication, which is a challenging in a Cloud environment. This paper presents a framework for the execution of long running MPI multi-physics coupled applications on a private Cloud environment. Further, the paper presents a performance analysis of the execution of the MPI coupled application on virtual resources managed by Eucalyptus Cloud.

Keywords: Cloud computing, Eucalyptus, high-performance computing, MPI applications, Scientific computing.

I. INTRODUCTION

Scientists use computer-based simulations to deepen their understanding of complex phenomena, especially in areas such as earth science, biology and physical sciences. Such scientific application involves the construction of complex mathematical models and numerical solution to simulate the scientific problem under study. Multi-physics coupled applications are large-scale scientific computing comprising of interacting software component applications. These applications have gained increasing attentions since multi-disciplinary multi-component models are used to accurately model phenomena of interest in the areas of climate, space weather, solid rockets, fluid structure interaction, heart disease

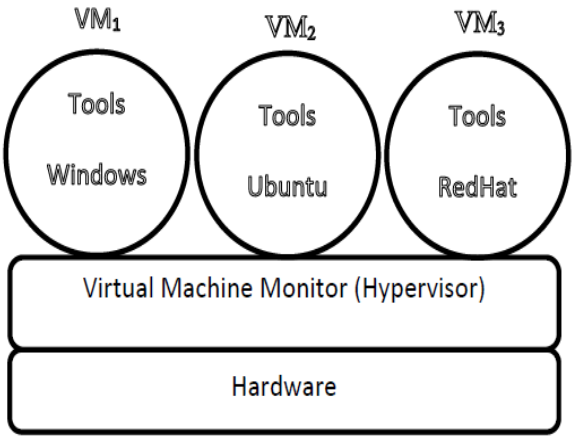

Fig. 1. A server host and a virtualization layer include three virtual machine each include an operating system and different user level tools.

and cancer studies. For example, to study fluid flow over a surface [2], the interaction of fluid molecules with the surface is modelled atomistically using classical molecular dynamics, while the fluid bulk is modelled using a continuum method. Coupling together different models of individual systems, which affect the system of interest, provides an accurate simulation for phenomena under study [3]. These applications are large-scale, long running distributed compute intensive application. Such simulations require high-performance computing resources to reduce the computational complexity into a reasonable time [1].

The needs for high-performance computing have been addressed with the Grid computing technology to provide the scientists with cheap high-performance and large scale computing resources through collaboration among multiple academic organizations [2, 4-7]. Due to the huge advances of networking technology, the computational Grid has emerged to address the problem of effectively and efficiently exploiting multi-institutional distributed resources to

benefit a single application. The goal of the computational Grid is to allow coordinated and collaborative resource sharing to enable the development and execution of large-scale, complex scientific computational problems, which would otherwise require the power of expensive supercomputers or would have been impossible to develop. However, there could be limitations between the hosting operating systems and the software requirements of the developed scientific applications. For example, scientific applications may require specific tools and APIs that have to be offered during the runtime on the Grid's resources. For example, scientific coupled applications require complex communication patterns during runtime, which would require message-passing tools, such as PVM and MPI on the Grid resources during runtime. However, the required tools and APIs may not be available on the Grid resources where applications are scheduled [8].Cloud computing has emerged as the cutting edge IT technology to provide a flexible, on-demand elastic computing infrastructure for a number of applications [9, 10]. The increasing popularity of the cloud computing technology is driven by the ultimate advancement of the virtualization technology, which allows different logical machines to share the same hardware but to run isolated from each other. The main goal of the virtualization is to improve resource sharing and utilization. The isolated logical host may include different running operating systems (OS) as well as user's level software installed on an operating system. The logical host is called *Virtual Machine* (VM) images, as shown in Fig. 1. The Virtual Machine Monitor (VMM), also called *hypervisor*, is a software layer mediates the access of the VMs to the physical resources, and allows the VMs to operate as if they were running on different machines independently [11]. As a result, a flexible control over the VM image can obtained which is hard to have such feature with other high-performance infrastructure, such as the Grid and Clusters [12]. Popular virtualization hypervisors are VMware vSphere [13], XEN [14] and KVM [15].

The *Cloud computing management* layer is a software layer on top of the virtual machine hypervisor in order to controls the VMs, as shown in Fig. 2. This layer accesses the entire physical infrastructure, manages all the available virtual resources and delivers the virtual resources as a service over high-speed Internet. Cloud computing technology brings the potential for providing high-performance and the available supercomputing power to the public free or for small charges. Companies such as Google, IBM, Amazon, and Yahoo provide services based on their Cloud architectures, such as computing resources, online storage, social networking sites, e-commerce, and web-based email. As a result, large institutions as well as individuals will be able to use Cloud resources for their computing and storage needs as they need them and only pay for what they use instead of paying for a certain amount of time [10]. There is a

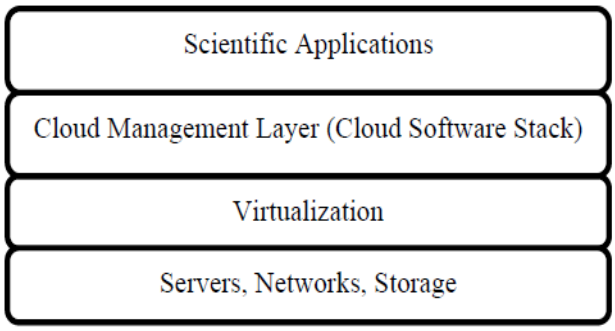

Fig. 2. The overall architecture for a scientific Cloud.

number of open source Cloud management layers, such as Eucalyptus [16, 17], Nimbus [18] and OpenNebula [19], which allow organizations and individuals to build private clouds in order to have full control and to improve the utilization of the available computational resources. Fig. 2 shows the overall architecture for a scientific private Cloud. In the last layer, the scientific coupled model application is executed on top of the Cloud. However, coupled model application are distributed, and require support for the complex communication pattern between its distributed components, such as optimised MPI communication, which is a challenging problem in a Cloud environment.

This paper proposes a framework based on Eucalyptus in order to explore the usability and efficiency of executing MPI distributed applications on a private Cloud. The remainder of this paper is organized as follows. Section 2 presents the related work on Cloud computing and experiences in executing scientific applications in Cloud environments. Section 3 presents the proposed approach for executing a scientific application on the Cloud. Section 4 describes the proposed Cloud environment, the experimental setup, and the experimental results as well as the performance analysis. Finally, conclusions and future work are given in Section 5.

II. BACKGROUND

This section presents two broad categories of related work. The first category discusses the services offered by the Cloud, and the second category presents the related work to executing applications on the cloud environment.

A. Services in the cloud

Cloud computing is a large-scale parallel and distributed computing infrastructure. It consists of a collection of inter-connected and virtualized computing resources that are managed to be one or more unified computing resources [9]. Further, the provided abstract, virtual resources, such as networks, servers, storage, applications and data, can be delivered to the end user as a service rather. Services are delivered on demand over the Internet as three types of computing architecture, namely Software as a Service (SaaS), Platforms as a Service (PaaS) and Infrastructure as a service (IaaS). The main goal is to provide users with flexible services in a transparent manner, cheaper, scalable, highly

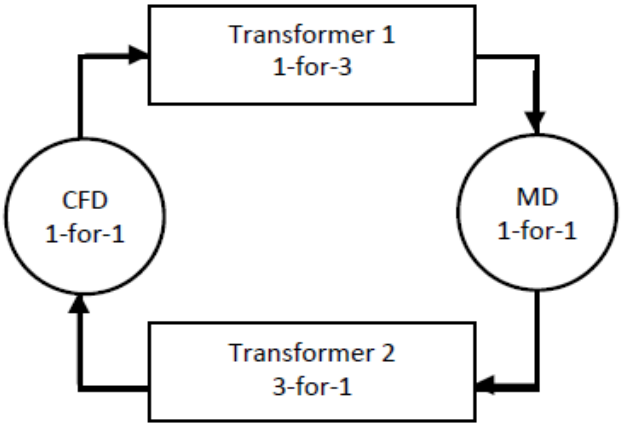

Fig. 4. Data flow between the component models for the HybridMD coupled model.

available and powerful computing resources with certain Quality of Services (QoS) [10]. The SaaS architecture provides software applications hosted and managed by a service provider to the end-user replacing locally-run applications with web services applications. In the IaaS architecture, service includes provision of hardware and software for processing, data storage, networks and any required infrastructure for deployment of operating systems and applications which would normally be needed in a data center managed by the user. In addition, the IaaS architecture allows the end user to execute data/compute intensive applications over the infrastructure by employing parallel runtimes over the accessed VMs images. In the PaaS architecture, service includes programming languages and tools and an application delivery platform hosted by the service provider to support development and delivery of end-user applications [9, 10].

Further, clouds can also be classified according to their deployment model into public and private clouds. A public cloud is generally available to the public on pay-per-use basis, such as Amazon EC2, Azure and Google Apps, as shown in Fig. 3. Several open source Cloud management tools have emerged to enable the creation of private clouds, such as Eucalyptus, Nimbus and Open Nebula. Private Clouds that are only accessible from within an organization.

B. Clouds and Scientific Applications

Google File System (GFS) [20], Hadoop Distributed File System (HDFS) [21] and Hadoop MapReduce [22] are Cloud technologies which provide programming frameworks for distributed applications over a private PaaS Cloud architecture. The data are stored in compute nodes or data centers and the computations moves to the data for processing [23]. A number of open source Cloud management layers exist which can be deployed inside an organization in order to setup a private Cloud, specifically IaaS architecture. For example, Eucalyptus [16], Nimbus [18] and OpenNebula [24] are used to build IaaS private Cloud infrastructure. They are mainly virtual machine management that allows a dynamic deployment and reallocation of virtual machines. IaaS utilise existing virtualization resources, such as computing, virtualized networks and data storage, to come up with a new virtualization layer on top the existing physical infrastructure. Several private Clouds utilise these management layers to provide a high-performance execution environment for intensive data/compute scientific applications. For example, Science Cloud [1] and Future Grid [8]. VMs are provided to the user under certain SLAs, where the Cloud provider guarantees a specific level of system's performance to their clients. Further, in [25], an evaluation of the impact of Xen on MPI distributed applications on a private Cloud is conducted. However, their study did not provide a detailed description whether the number of VMs on the same node or on different nodes. In addition, their study did not try to optimize the commination using tools such as Open-MX. In [26], an infrastructure based on Spark MapReduce model using Xen virtualization layer and a private Cloud managed by XCP for executing MPI parallel applications on the Cloud. However, in their framework, it is not clear that the used benchmark is based on MPI communication.

III. THE PROPOSED ARCHITECTURE

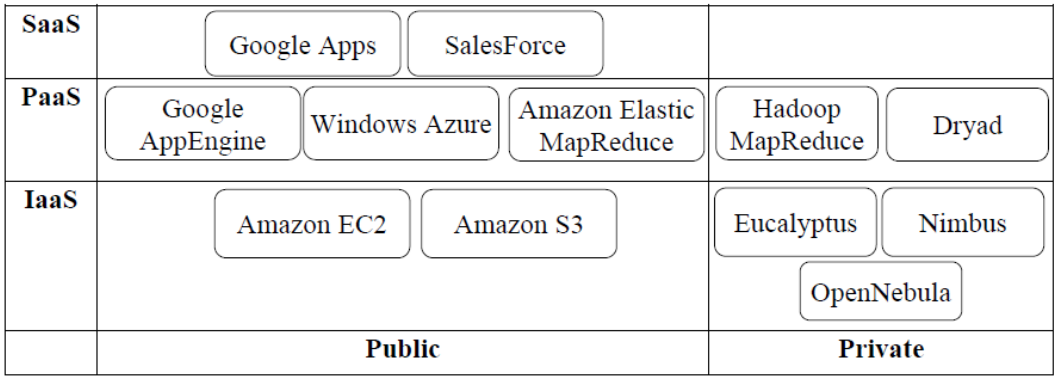

Fig. 3. Available public Clouds and open source Cloud management software for building private Clouds [1].

This section starts with a detailed description of the scientific coupled model application used in the study, then a presentation of the cloud architecture used for executing the application

A. The Coupled Model Application

Coupled modelling is an outstanding physical approach for simulating complex scientific phenomena. A coupled model is normally used to produce a more accurate model for the system of interest by coupling different models of individual systems, which affect the system of interest. Each individual model may act at a different length or time scale, or focus on a distinct underlying physical phenomenon. In computational terms, coupled models can be viewed as distributed component-based applications in which each individual model becomes a software component. In the general coupling framework, reported in [27], for example, each individual model embodies three phases of operation, namely *initialization*, *iteration* and *termination*. The models are synchronized via exchange of information in a series of *put()* and *get()* calls to the run-time system during each iteration. For each message transfer, the sender model executes *put()* with a suitable data structure, and the receiver model receives the communicated data by executing a corresponding *get()*. The data sent between the models is termed *coupling data*.

The individual models of most interest are termed *scientific models*, which repeatedly perform one round of *get()*s, then do some work based on the received coupling data, then perform one round of *put()*s. A single cycle of this iteration is known as a *minor cycle* and is equivalent to one time-step in the underlying iterative model. But, different scientific models are allowed to iterate at different rates and *transformer models* are then required in order to reconcile these rates. For example, if there are two models coupled together, and one model executes n rounds of *put()*s for every round of *get()*s in the second model, an intermediate n-for-1 transformer model is required to perform the necessary reconciliation [3]. An n-for-1 transformer thus performs n minor cycles during which it performs n rounds of *get()*s before performing one round of *put()*s. A 1-for-n transformer performs n minor cycles during which it performs one *get()* group followed by n *put()* groups. The resulting cycle of the entire coupled model is termed a *major cycle*.

B. HybridMD Coupled Model Application

The HybridMD coupled model is an interesting modelling and simulating complex fluids that handles molecular dynamics (MD) coupled to computational fluid dynamics (CFD) within a single simulation. The main goal is to study fluid flow over a surface [27]. The interaction of fluid molecules with the surface is modelled atomistically using classical molecular dynamics, while the fluid bulk is modelled using a continuum method. Each model tackles a separate physical region of the system. The data exchange between models provides the boundary conditions for each region (e.g. temperature and pressure). Considering the interaction between the hydrodynamics and the specific molecular processes near the surface clarifies the properties of these physical systems and addresses key problems, such as how hydrodynamics effects influence molecular interactions at interfaces, at lipid bilayer membranes and within individual macromolecules or assemblies of them.

The HybridMD coupled model consists of two models (CFD and MD) and two transformers, as shown in Fig. 4. MD iterates at three times the frequency of CFD. *Transformer1* is a 1-for-3 transformer that is used to reconcile the communic-ation between CFD and MD. *Transformer2* is a 3-for-1 transformer that reconciles the communication between MD and CFD. One major cycle of the entire coupled model involves three minor cycles of the MD model and one minor cycle of the CFD model.

C. *The Proposed Scientific Private Cloud Architecture*

Table 1 shows the overall architecture of the proposed infrastructure for a scientific private Cloud for executing a distributed scientific application. The infrastructure is composed of five layers. The first layer is the hardware layer, which represents the computational resources and networks. The second layer is the VMM hypervisor where Xen is used to create a number of VMs of a desired configuration on the available computational resources. The third layer is the Cloud Management layer. In this layer, Eucalyptus is used to manage the VMs images. The fourth layer is the runtime software layer, which provides the required tools for the application during runtime. In this layer, The virtual machines employs OpenMPI 1.6 as MPI [28]. Further, Open-MX is an optimized implementation of message-passing stack over Ethernet. This is used to reduce the overhead latency of the MPI communication over Ethernet networks using TCP [29]. Finally, the fifth layer is the scientific application layer where the distributed component are submitted to the Cloud and each component is assigned to a specific VM to run.

IV. EXPERIMENTAL SETUP AND EVALUATION

The Eucalyptus version 4 is used to setup a small-scale scientific private Cloud. Two nodes are deployed. Each node has i7 core Intel 2.2 GHz processor and 32GB memory. Each node runs Ubuntu version 12 operating system. OpenMPI version 1.5 along with Open-MX version 1.4 are used as MPI. 1 Gigabit Ethernet network fabric is used for networking. The virtualization layer is based on Xen hypervisor version 4.3. The VMs are deployed using Eucalyptus based on the previous configuration. The number of CPU cores assigned to each VM image can be controlled by Eucalyptus, as shown in Table 1.

The hybridMD application is run 50 times on each configuration. The different VM images used for deployment are shown in Table 2. The configuration number 4 is used to compare the performance on the Cloud with the performance on standalone machine. On each configuration, each model with its transformer is run on a single VM. For configurations 2 and 3, each model and its transformer is deployed on a different node.

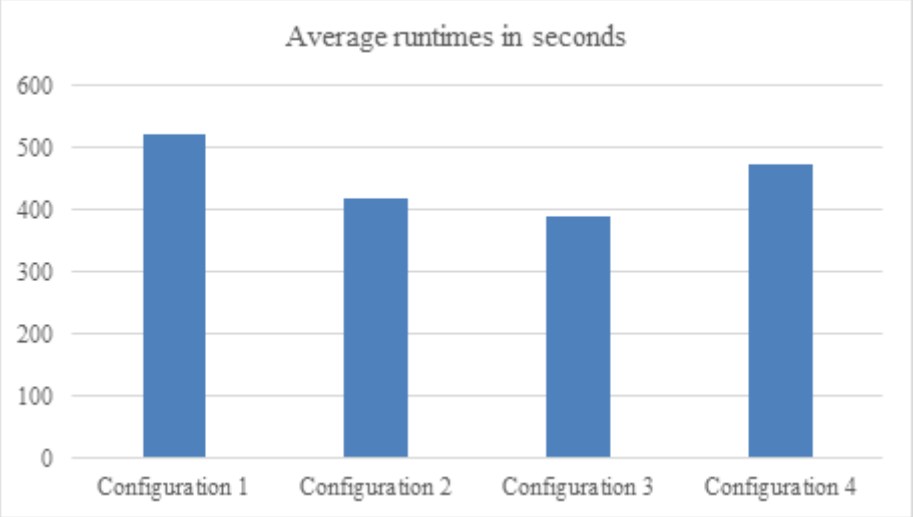

Fig. 5. Average execution time of HybirdMD under diffirent VM configuration of Table 2.

TABLE 1. PROPOSED INFRASTRUCTURE FOR SCIENTIFIC PRIVATE CLOUD.

Fifth Layer	**The Scientific Application Layer**	Distributed Coupled Model
Fourth Layer	**Runtime Layer**	OpenMPI OpenMX
Third Layer	**Cloud Management Layer**	Eucalyptus
Second Layer	**Virtual Machine Monitor Layer**	Virtual machines using Xen
First Layer	**Hardware Layer**	Computational resources Networks

TABLE 2. THE VMS CONFIGURATION USED IN EXPERIMENTS

Configuration Number	**Number of cores assigned to a VM**	**Number of VM on each node**	**Amount of memory assigned to each VM**	**Total number of VM**
1	4	1	16	2
2	2	2	8	4
3	1	4	4	8
4	A standalone machine with 32 GB memory, Intel I7 core 2.2 GHz			

Fig.5 shows the average runtime execution time for each configuration. The performance of the application on configuration number 1 is worse than the deployment in configuration 4 because of the virtualization overhead. Configurations 2 and 3 show a slight better performance than the performance on configuration 4. This is because the each VM is assigned to multiple cores can reduces the virtualization overhead, and as a result gives better execution time of the HybirdMD application on parallel resources.

V. CONCLUSION AND FUTURE WORK

Cloud Computing provides a promising infrastructure for executing scientific distributed applications. A Cloud comp-uting (infrastructure as a service) is presented for executing parallel scientific applications. A HybirdMD is a distributed scientific application that requires complex computation as well as communication pattern is used as a benchmark. The communication pattern is employed using MPI and Open-MX as optimized runtime tools for communication. The performance analysis shows that assigning multiple cores per VM can achieve a slight better execution times performance than execution times achieved using a standalone machine. In the future, An investigation will be conducted to study the performance of the distributed application developed using Hadoop MapReduce with MPI on a scientific private Cloud. Further, because the cloud is highly dynamic environment, an investigation will be conducted to provide techniques for high-level checkpointing and migration of the distributed components of the application.

REFERENCES

[1] Jha, S., et al., *Understanding Scientific Applications for Cloud Environments*, in *Cloud Computing*. 2011, John Wiley & Sons, Inc. p. 345-371.

[2] Pordes, R., et al., *New science on the Open Science Grid.* Journal of Physics: Conference Series, 2008. **125**(1): p. 012070.

[3] Hussein, M., et al., *Adaptive performance control for distributed scientific coupled models*, in *Proceedings of the 21st annual international conference on Supercomputing*. 2007, ACM: Seattle, Washington. p. 274-283.

[4] Foster, I. and C. Kesselman, *The Grid 2: Blueprint for a New Computing Infrastructure*. 2003: Morgan Kaufmann Publishers Inc.
[5] Coveney, P.V., et al., *Scientific Grid Computing: The First Generation.* Computing in Science and Engg., 2005. **7**(5): p. 24-32.
[6] *A grid-enabled web service for low-resolution crystal structure refinement.* Acta Crystallographica Section D Biological Crystallography, 2012. **68**(3): p. 261.
[7] Pordes, R., t.O.S.G.E. Board, and J. Weichel, *Analysis of the current use, benefit, and value of the Open Science Grid.* Journal of Physics: Conference Series, 2010. **219**(6): p. 062024.
[8] Vecchiola, C., S. Pandey, and R. Buyya, *High-Performance Cloud Computing: A View of Scientific Applications*, in *Proceedings of the 2009 10th International Symposium on Pervasive Systems, Algorithms, and Networks*. 2009, IEEE Computer Society. p. 4-16.
[9] Armbrust, M., et al., *A view of cloud computing.* Commun. ACM, 2010. **53**(4): p. 50-58.
[10] Buyya, R., et al., *Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility.* Future Gener. Comput. Syst., 2009. **25**(6): p. 599-616.
[11] Uhlig, R., et al., *Intel Virtualization Technology.* Computer, 2005. **38**(5): p. 48-56.
[12] Ekanayake, J. and G. Fox, *High Performance Parallel Computing with Clouds and Cloud Technologies*, in *Cloud Computing*, D. Avresky, et al., Editors. 2010, Springer Berlin Heidelberg. p. 20-38.
[13] Burd, S.D., et al. *Virtual Computing Laboratories Using VMware Lab Manager*. in *System Sciences (HICSS), 2011 44th Hawaii International Conference on*. 2011.
[14] Barham, P., et al., *Xen and the art of virtualization.* SIGOPS Oper. Syst. Rev., 2003. **37**(5): p. 164-177.
[15] Childers, B., *Virtualization shootout: VMware server vs. VirutalBox vs. KVM.* Linux J., 2009. **2009**(187): p. 12.
[16] Nurmi, D., et al., *The Eucalyptus Open-Source Cloud-Computing System*, in *Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*. 2009, IEEE Computer Society. p. 124-131.
[17] Kijsipongse, E. and S. Vannarat, *Autonomic resource provisioning in rocks clusters using Eucalyptus cloud computing*, in *Proceedings of the International Conference on Management of Emergent Digital EcoSystems*. 2010, ACM: Bangkok, Thailand. p. 61-66.
[18] Tudoran, R., et al., *A performance evaluation of Azure and Nimbus clouds for scientific applications*, in *Proceedings of the 2nd International Workshop on Cloud Computing Platforms*. 2012, ACM: Bern, Switzerland. p. 1-6.
[19] Sempolinski, P. and D. Thain, *A Comparison and Critique of Eucalyptus, OpenNebula and Nimbus*, in *Proceedings of the 2010 IEEE Second International Conference on Cloud Computing Technology and Science*. 2010, IEEE Computer Society. p. 417-426.
[20] Ghemawat, S., H. Gobioff, and S.-T. Leung, *The Google file system.* SIGOPS Oper. Syst. Rev., 2003. **37**(5): p. 29-43.
[21] Shvachko, K., et al., *The Hadoop Distributed File System*, in *Proceedings of the 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*. 2010, IEEE Computer Society. p. 1-10.
[22] Dittrich, J., et al., *Efficient big data processing in Hadoop MapReduce.* Proc. VLDB Endow., 2012. **5**(12): p. 2014-2015.
[23] Srirama, S.N., P. Jakovits, and E. Vainikko, *Adapting scientific computing problems to clouds using MapReduce.* Future Gener. Comput. Syst., 2012. **28**(1): p. 184-192.
[24] Milojicic, D., I.M. Llorente, and R.S. Montero, *OpenNebula: A Cloud Management Tool.* IEEE Internet Computing, 2011. **15**(2): p. 11-14.
[25] Youseff, L., et al., *Evaluating the Performance Impact of Xen on MPI and Process Execution For HPC Systems*, in *Proceedings of the 2nd International Workshop on Virtualization Technology in Distributed Computing*. 2006, IEEE Computer Society. p. 1.
[26] Tabaa, Y. and A. Medour, *Towards a next generation of scientific computing in the Cloud.* International Journal of Computer Science Issues (IJCSI), 2012. **9**(6): p. 7.
[27] Ford, R.W., et al., *GCF: a general coupling framework.* Concurr. Comput. : Pract. Exper., 2006. **18**(2): p. 163-181.
[28] M. Squyres, J. and A. Lumsdaine, *The Component Architecture of Open MPI: Enabling Third-Party Collective Algorithms**, in *Component Models and Systems for Grid Applications*, V. Getov and T. Kielmann, Editors. 2005, Springer US. p. 167-185.
[29] Goglin, B., *High-performance message-passing over generic Ethernet hardware with Open-MX.* Parallel Comput., 2011. **37**(2): p. 85-100.

