2011 Third IEEE International Conference on Coud Computing Technology and Science

ViteraaS: Virtual Cluster as a Service

Frank Doelitzscher, Markus Held and Christoph Reich
Department of Computer Science
Hochschule Furtwangen University, Germany
Email: {frank.doelitzscher, markus.held, rch}@hs-furtwangen.de

Anthony Sulistio
Department of Applications, Models and Tools
HPC Center Stuttgart (HLRS) Germany
Email: sulistio@hlrs.de

***Abstract*—The idea behind cloud computing is to deliver Infrastructure-, Platform- and Software-as-a-Service (IaaS, PaaS and SaaS) over the network on an easy pay-per-use business model. In this paper, we present our work, Virtual Cluster as a Service (ViteraaS), that provides on-demand high performance computing for research projects, and e-Learning and teaching purposes in a private cloud. Moreover, ViteraaS can be extended to use Amazon's public cloud infrastructure as needed. ViteraaS can be categorized as PaaS that leverages OpenNebula, a virtual infrastructure manager, to dynamically create a cluster of virtual machines (VMs) on idle resources or dedicated servers. In addition, ViteraaS is integrated within the university's existing IT infrastructure like Single Sign-On for seamless authentication and authorization. Finally, a Quality of Service monitoring module is used by ViteraaS to monitor the performance and status of these VMs.**

***Keywords*-cloud computing, virtualization, virtual cluster, HPC**

I. Introduction

Foster et al. [1] discuss the basic concepts of cloud computing, whereas Armbrust et al. [2] give a good overview of cloud computing by highlighting obstacles and proposing several opportunities to solve them. From our point of view, cloud computing delivers Infrastructure-, Platform-, and Software as a Service (IaaS, PaaS, and SaaS) on a simple pay-per-use basis. Cloud computing enables companies to avoid over-provisioning of IT infrastructure and training personnel. As a result, these companies can take advantage of cloud computing if IT capacity needs to be increased on the fly. This increment is typically needed for services which are only requested for a certain period of time.

In a typical university scenario, PC labs and servers are under-utilized during the night and semester breaks. In addition, these resources are on high demand mainly towards the end of a semester. Moreover, cloud computing can also be used to support e-Learning services [3]. Finally, cloud computing delivers many advantages to institutions, such as reducing costs, increasing operational efficiency, and providing the flexibility to workload demands, according to an analysis done by Sterling and Stark [4]. With the aforementioned motivations and scenarios, we offer a private cloud solution using an existing IT infrastructure [5].

The targeted users of our private cloud are staff and students of the Hochschule Furtwangen University (HFU) running compute-intensive applications. In this paper, we introduce our work in building **Vi**rtual Clus**ter a**s **a S**ervice (ViteraaS), a PaaS model for running high performance computing (HPC) programs as part of e-Learning and teaching purposes, and various research projects. ViteraaS leverages OpenNebula [6], a virtual infrastructure manager, to dynamically create a cluster of virtual machines (VMs) on idle resources or dedicated servers [7]. In addition, ViteraaS is integrated within the university's existing IT infrastructure like Single Sign-On (SSO) for seamless authentication and authorization. Finally, a monitoring module is used by ViteraaS to monitor the Quality of Service (QoS) attributes of these VMs, such as CPU load and network traffic, through a web browser.

The rest of this paper is organized as follows. Section II provides some related work, whereas Section III explains ViteraaS' architecture. Section IV presents several performance benchmark scenarios and results. Finally, Section V concludes the paper and gives future work.

II. Related Work

Amazon offers special HPC cloud VMs [8] with quadruple extra large instances in their data center (in US East Virginia at the time of writing). These instances use InfiniBand on a CentOS-based operating system (OS). However, they come without any cluster management software unlike our proposed solution.

Ekanayake et al. [9] compare different cloud technologies like MapReduce and DryadLINQ, and their suitability in running HPC applications. Similarly, Akioka and Muraoka [10] evaluate the computational performance of Amazon instances with several HPC benchmarks, and estimates its operational cost. In our work, we focus on a virtual cluster running MPI jobs. In addition, ViteraaS can leverage Amazon instances and provides a management system that monitors QoS attributes and costs.

Amedro et al. [11] describe an efficient high-level Grid and cloud framework, which allows a smooth transition from clusters and Grids to Clouds. Hence, applications can easily run on clusters, grid or a mixture of them without changing the application's code. ViteraaS is a purely cloud approach that separates infrastructure information from an application. Instead of the grid approach for the local side, ViteraaS virtualizes all cluster nodes, uses one central cloud manager and can setup different cluster sizes as required. Furthermore, ViteraaS allows over-provisioning if the deployed virtual clusters are not in use or have a lower priority, e.g. for test provisioning purposes.

978-0-7695-4622-3/11 $26.00 © 2011 IEEE
DOI 10.1109/CloudCom.2011.101

The Virtual Computing Laboratory (VCL) [3], developed by North Carolina State University (USA), enables students to reserve and access virtual machines (VMs) with a basic image or specific applications environments, such as Matlab and Autodesk. On the other hand, ViteraaS mainly focuses on deploying HPC applications running on MPI and it is able to deploy jobs on Amazon's public cloud as needed.

III. ViteraaS Architecture

Figure 1 shows a high level overview of our hybrid cloud infrastructure that is built on top of our university's existing resources. It consists of three computer pools (`PC Pool`, `Research Pool` and `Server Pool`) located at different locations and use separate IP subdomains within the university. The `PC Pool` has 18 computers with Ubuntu OS and KVMinstalled, whereas the other pools have Debian OS and Xenconfigured. The `PC` and `Server` pools are used by students and staff for teaching purposes during the day. In contrast, the `Research Pool` is used for research and development purposes. All pools are managed by a Cloud Management System [5]. Amazon's public cloud is also used to handle peak loads, typically during the end of a semester.

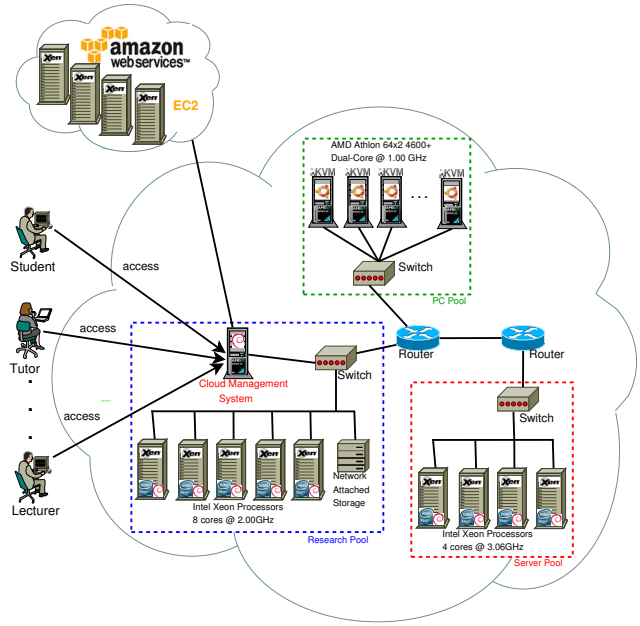

Fig. 1. Overview of the university's hybrid cloud infrastructure.

With ViteraaS, users can create a dynamic HPC cluster consisting of VMs which are deployed on the aforementioned computer pools. The size of each cluster can be determined by the user, and is created on demand. After a job is finished, the cluster is "destroyed" and resources are released in time for new clusters. A user accesses ViteraaS either through a command line-based interface (CLI) shown in Figure 2, a XML-RPC based Application Programming Interface (API), or via web browser. In all cases, the user can create, alter (including deletion), and get status information about his virtual cluster(s). Moreover, in this figure, the user can add, show status and delete a job.

```
**********************************************************
*                CLI client for ViteraaS                 *
**********************************************************
Server: 127.0.0.1
Port: 8084
Please enter your username: hpc
Please enter your password:
***********************************
[0] - exit Programm.
[1] - change username and password.
***********************************
Cluster tools
[2] - create a cluster.
[3] - show cluster info.
[4] - expand a cluster.
[5] - delete a cluster.
***********************************
Job tools
[6] - add a job.
[7] - show jobs.
[8] - show job information.
[9] - delete a job.
```

Fig. 2. The main menu of ViteraaS' command line interface.

Using a browser or the XML-RPC interface API, it allows the usage of Single Sign-On (SSO) authentication via *Shibboleth* [12]. Using Shibboleth avoids the usage of multiple user management systems, and provides a SSO solution, which is the preferred authentication and authorization system at our university. The *Web Server* redirects unauthenticated users to a Shibboleth Identity Provider for authentication, and grants access by Shibboleth attributes, as depicted in Figure 3.

Figure 3 shows the components of ViteraaS: The *Cluster & Job Management* module acts as a controller to various sub-components and builds a virtual cluster. It interacts with *OpenNebula*, a domain name system (*PowerDNS* [13]), a *QoS Monitoring* module and a *Batch System* for scheduling jobs.

The user specifies a cluster size and a memory requirement of each VM. Then, the *Cluster & Job Management* creates the virtual cluster by leveraging *OpenNebula*. Next, it informs the *Batch System* that a new cluster is available and ready to use. *Batch System* then deploys the user's job file from a webserver upload-directory to its client component installed on every VM. Job files are stored on NFS storage, where they are shared among the VMs within the same virtual cluster. Once the job has been completed, the output file gets stored on a web server directory for downloading. Note that users can only have access to their job and output files.

A. Integration with OpenNebula

OpenNebula is used as the central cloud management. It is responsible for finding available resources, creating VMs based on a master image, and deploying the image to the physical host. The size of a cluster is chosen by the user, up to 32 VMs to prevent misuse of resources. Each VM gets contextualized in order to have a unique name, a unique MAC address, IP and a virtual network (vNet) ID. Each vNet consists of a fixed set of generated IP-MAC pair addresses, as shown in Figure 4. Afterwards, a vNet template is created (Figure 5), it is used by OpenNebula to create a virtual network and to map on top of the physical network for every virtual

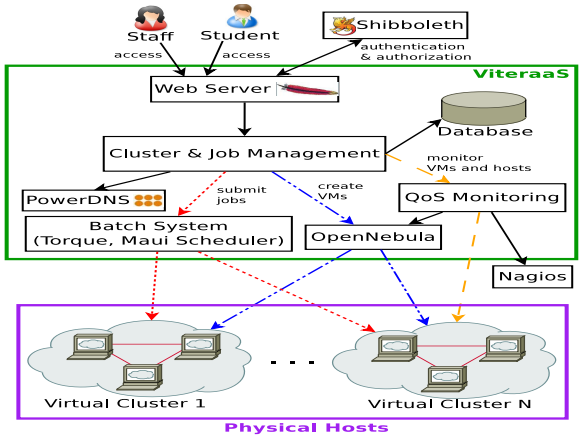

Fig. 3. Architecture of ViteraaS.

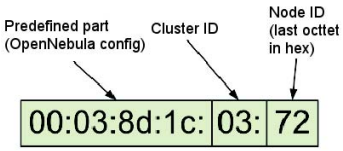

Fig. 4. A generated MAC address of a virtual cluster.

cluster. Each cluster stands on its own vNet, in order to isolate the various virtual clusters. To achieve this, *ebtables* is used.

```
TYPE = FIXED
NAME = cluster0
BRIDGE = eth0
LEASES =[IP="141.28.99.113",MAC="00:03:8d:1c:0:71"]
LEASES =[IP="141.28.99.114",MAC="00:03:8d:1c:0:72"]
...
```

Fig. 5. A vNet template in OpenNebula.

B. Batch System with Torque and Maui Scheduler

Torque [14] is used as a basic Portable Batch System (PBS) for ViteraaS. Torque offers a rudimentary scheduling engine. Thus, ViteraaS uses Maui Scheduler [15], as it is a very powerful scheduler for clusters and supercomputers and extends the default Torque scheduling system with various strategies, e.g. the computation of job priorities, and support for QoS and fairness rules.

The *Batch System* module of ViteraaS forms a virtual cluster, where it can change in size dynamically at runtime. The module controls Torque and Maui, and allows jobs addition or deletion, collects status information from compute nodes, and shows a job queue. Since Torque supports only a CLI, a wrapper in Java has been developed. Thus, PBS of ViteraaS can be divided into three processes, i.e. `pbs_server` to provide a batch system and is running on the server node; `maui` to run the PBS scheduler on the server node; and `pbs_mom` to run on the compute nodes for monitoring its implementation and to start MPI jobs.

C. Other ViteraaS Components

The *PowerDNS* module handles communication with a domain name system (DNS) using a PowerDNS backend [13]. With this module, it is possible to create or delete DNS entries in a database. These changes are done at runtime in the DNS database, and are effective immediately.

A *physical host* consists of a hypervisor, a firewall, and multiple VMs or virtual compute nodes. The host uses Xen version 3.2-1 on a Debian OS. To isolate different VMs which are running on a same physical host, ebtables is used.

The *virtual compute node* are actually cluster's worker nodes running MPI programs. The nodes are generated from a master image, and automatically configured using bash scripts. The master image is a minimal Debian installation, containing `mpich2` and `gfortran` libraries, as well as a PBS client agent. Each image is about 500 MB in harddisk size which can be adjusted according to users needs.

D. Quality of Service Monitoring

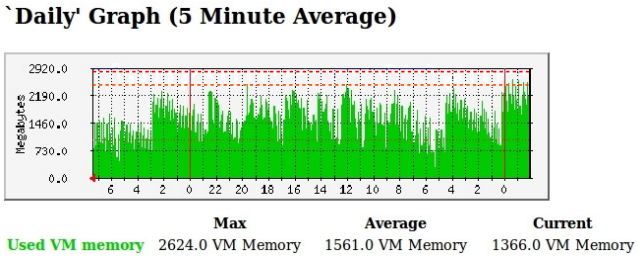

Fig. 6. Timeline of the memory usage of the virtual machine.

Quality of Service (QoS) serves as an assurance to performance characteristics of individual applications during runtime. A Simple Network Management Protocol (SNMP) is used to monitor the VMs with regards to CPU and memory usages, and network bandwidth. SNMP uses an UDP-based network protocol, and therefore requires an agent to be installed in each VM. The *QoS Monitoring* module acts as an SNMP master that periodically collects real-time information from agents and stores the data into a database.

For monitoring the physical and virtual machines, the *QoS Monitoring* module interacts with OpenNebula and Nagios [16], as shown in Figure 3. OpenNebula's XML-RPC interface is used to collect real-time information about the status of hosts at a certain interval, e.g. every five seconds. For example, in Figure 6, a memory usage a VM is drawn by NagiosGrapher. NagiosGrapher [17] is a Nagios plugin that displays the data into an online chart. Note that the horizontal lines in this figure indicate a "warning" level (85% of memory size), and "alarm" level (95% of memory size) in Nagios, respectively. As a result, the *QoS monitoring* module allows users to debug their application, e.g. in case of crashes due to low memory or to optimize the parallelism of a job.

IV. Performance Experiment

To show the performance of ViteraaS, we perform several NAS Parallel Benchmarks (NPB) [18] on the university's *Research Pool*, on Amazon's HPC machines, and on a mixed testbed using ViteraaS managing virtual cluster nodes in the *Research Pool* and in Amazon's data center.

The machines of the local testbed at the university's `Research Pool` are equipped with 8x CPU: Intel(R) Xeon(R) CPU E5504 @ 2.00GHz 64-bit architecture, 12 GB of memory and 1 Gigabit Ethernet. In Amazon, we used two HPC cluster compute instances (`cc1.4xlarge`). They each offer 23 GB of memory, 2 x Intel Xeon X5570, quad-core "Nehalem" 64-bit architecture and 10 Gigabit Ethernet. To carry out the experiments, it has to be installed `libgfortran44` and `mvapich2`, which is an MPI-2 implementation (conforming to the MPI 2.2 standard).

Worker nodes are instances that compute MPI jobs. These jobs are load-balanced and distributed evenly over the nodes by the cluster manager. For example, eight jobs are distributed to four compute nodes (CNs) resulted in two jobs in each node. Virtualized worker nodes are basically pre-configured VMs, so called Cluster VMs (CVMs). They are not distributed evently over the compute nodes because the policy of ViteraaS and OpenNebula respectively, places the CVMs on the same compute node, as long as the compute node has enough resources. Hence, eight CVMs are placed on the same compute node (8-core machine) if the requirement for the CVM is one core and does not exceed the memory need of the total memory of the compute node. Therefore, for example, eight jobs are distributed to one compute node.

A. Benchmark Description

The NAS Parallel Benchmarks comprise of a widely used set of programs, that is designed to evaluate the performance of HPC systems. The benchmark consists of eight programs, which mimic typical scientific computation problems. Of that are five kernels (EP, MG, FT, CG, IS) and three pseudo applications (LU, SP, BT) programs. For our experiments, we only use the following benchmark types:

- *EP (Embarrassingly Parallel):* Embarrassingly parallel Monte Carlo kernel to compute the solution of an integral. This test is a CPU intensive with no communication among the MPI nodes.
- *SP (Scalar Pentadiagonal):* Computational fluid dynamics application using the Beam-Warming approximate factorization method. This benchmark tests the inter-communication among MPI nodes.

We carried out the NPB on four different environments:

- *physical*: A physical cluster testbed without virtualization at our university with a maximum number of 4 compute nodes.
- *ViteraaS*: The same testbed like the physical one, but using ViteraaS and virtualized cluster nodes.
- *Amazon*: Amazon's EC2 infrastructure using two dedicated HPC compute nodes.
- *ViteraaS + Amazon*: using both ViteraaS and Amazon EC2, where the nodes are partitioned, according to a demand.

B. Benchmark Results

Table I shows the benchmark results, where the time measurements are mean values of 10 test runs. Note that *no. of CNs* represents the underlying physical CNs. In addition, the name of the benchmark represents the benchmark type, the class type and the number of worker nodes. For example, *ep.D.32* uses EP benchmark type of class D with 32 worker nodes. For the same benchmark type, class D runs longer than class C. Finally, the number of CNs for *ViteraaS & Amazon* denotes the number of worker nodes used in each site, e.g. "2 & 1" means two ViteraaS nodes and one Amazon node are created and running.

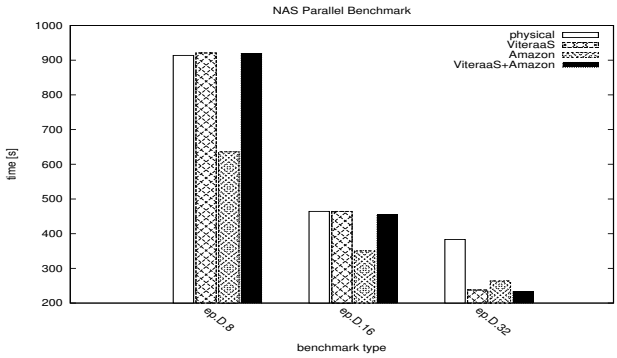

Fig. 7. Result of the EP benchmark type.

As shown in Figure IV-B, the EP benchmark results are comparable, and as expected they depend on the CPU performance of the hardware. The results are within $\pm$ 1% of each other. Taking a closer look, it can be seen that the *ep.D.32* benchmark performs better on a virtualized environment then using the bare-metal hardware. After some investigation, we found out that the main difference between the two setups is the Linux kernel. The Xen kernel (2.6.26-2-xen-686) used for virtualization in this experiment is about 18% to 25% faster then a standard Linux kernel (2.6.26.2-2-i686). Table II compares the performance of a regular Linux kernel and a Xen Linux kernel, using the linsolve benchmark [19] that solves linear equations with the complexity of 500 and 600.

Figure IV-B shows the results of the SP communication-intensive benchmark. They clearly show that a mixture of virtual cluster worker nodes at two different locations slows down the completion time significantly. This is because of the network performance between our university in Germany and Amazon's data center in Ireland. Figure IV-B also shows a worse performance by ViteraaS compared to the physical cluster, which is caused by the memory limitation of the VMs.

Overall can be said, that for embarrassingly parallel jobs, the placement of the virtual cluster nodes depends only on the CPU performance needed. Jobs that need a lot of communication, it is advisable to keep the virtual compute nodes

benchmark	*physical*		*ViteraaS*		*Amazon*		*ViteraaS & Amazon*	
	no. of CNs	time (s)	no. of CNs	time (s)	no. of CNs	time (s)	no. of CNs	time (s)
ep.D.8	4	913.86	1	920.66	2	635.86	1 & 1	918.43
ep.D.16	4	464.10	2	464.18	2	350.07	1 & 1	455.43
ep.D.32	4	383.34	4	237.47	2	263.36	2 & 1	232.78
sp.C.9	4	329.42	2	504.83	2	279.86	1 & 1	5386.41
sp.C.16	4	214.86	2	417.79	2	262.06	1 & 1	3982.69

TABLE I
RESULT OF NAS PARALLEL BENCHMARKS ON DIFFERENT TESTBEDS.

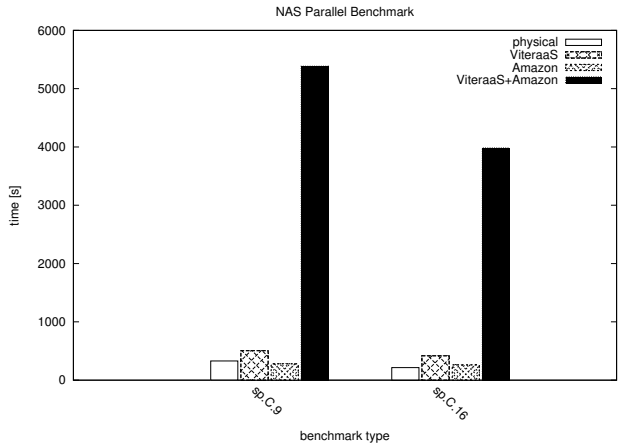

Fig. 8. Result of the SP benchmark type.

benchmark	2.6.26.2-2-i686	2.6.26-2-xen-686
complexity: 500	1872s	1505s
complexity: 600	3803s	3047s

TABLE II
BENCHMARKING OF DIFFERENT LINUX KERNEL

in one environment, rather than expanding it to a different cloud environment. Finally, the benchmarks show that a virtual cluster's performance loss is negligible compared to a physical cluster of the same hardware.

C. Advantages of ViteraaS

By having ViteraaS as PaaS, students can start learning about parallel programming and working on their projects, instead of spending time on installing MPI and Torque on their computers. The same scenario can also be applied to a tutor or lecturer in testing and marking the students' work. Another advantage is that ViteraaS can be used multiple times for students and staff for different purposes, such as lab modules, projects, experiments, and teaching environments.

For example, students at the Product Engineering Department at the HFU need to handle an embarrassingly parallel use case as lecture exercises for an optical simulation seminar. These exercises are compute-intensive since these students are trying to find optimal shapes, material and surfaces for optimizing a light focus. In a single desktop machine, one simulation takes approximately one hour. With a need of changing parameters of the simulation model, the total time for the whole exercise can exceed more than 16 hours. It can be easily seen, that this problem is trivial to parallelize and therefore very suitable for HPC. With ViteraaS, students can set up their own cluster in minutes, save significant execution time, and gain valuable experience in using parallel computing and MPI. Therefore, ViteraaS as PaaS provides a fool-proof environment and enhances the overall learning experience.

Finally, the IT department (IMZ) at the HFU can include ViteraaS into HFU's learning management system, which enable the usage of higher computing resources without purchasing new hardware. As a result, IT costs on hardware purchase, hardware maintenance, and administration can be reduced. In addition, with this approach, IMZ can monitor the usage of ViteraaS to control the use of Amazon EC2 and to prevent miscellaneous access, since IMZ is reponsible for the payment of Amazon services. Then, IMZ can charge VMs running on Amazon to the appropriate faculties.

V. CONCLUSION AND FUTURE WORK

In this paper, we introduce our work in building Virtual Cluster as a Service (ViteraaS), a Platform-as-a-Service model for running high performance computing (HPC) programs. ViteraaS is a virtual cluster management tool, which allows users to deploy on-demand virtual clusters of various sizes. The objective of ViteraaS is to simplify the creation/deletion of virtual HPC clusters and job submission, so that students and lecturers can take advantage of them. Therefore, ViteraaS has been integrated in the university's e-Learning platform to support courses like MPI programming.

ViteraaS is Single-Sign-On enabled using Shibboleth to simplify the user management process. Monitoring of the dynamic changing virtual cluster is automatically handled by the system. It allows the user to get status information about the cluster activity and get an instant notification if monitored resources exceed alarm levels.

The experiments show that the overhead of running embarrassingly parallel jobs are negligible for virtual clusters whose (virtual) cluster nodes can be deployed to different clouds. Jobs that need network communication should only be submitted to cluster whose (virtual) cluster nodes are within a single cloud infrastructure.

For future work, we would like to investigate university-specific Service Level Agreements (SLAs), such as granted number of resources for research and granted bandwidth. Of interest is reserving resources to grant a virtual cluster at

certain lecture time. Furthermore, we would like to investigate more on over-provisioning hardware nodes, when a running virtual cluster is idling.

ACKNOWLEDGMENT

This work was partially-carried out under the HPC-EUROPA2 project (project number: 228398), with the support of the European Community - Research Infrastructure Action of the FP7.

REFERENCES

[1] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud Computing and Grid Computing 360-Degree Compared," in *Proceedings of the Grid Computing Environments Workshop (GCE'08)*, Austin, Texas, USA, Nov. 16 2008.

[2] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the Clouds: A Berkeley View of Cloud Computing," Univ. of California at Berkeley, Tech. Rep. UCB/EECS-2009-28, Feb. 2009.

[3] M. Vouk, S. Averritt, M. Bugaev, A. Kurth, A. Peeler, H. Schaffer, E. Sills, S. Stein, and J. Thompson, ""Powered by VCL" – Using Virtual Computing Laboratory (VCL) Technology to Power Cloud Computing," in *Proceedings of the 2nd International Conference on the Virtual Computing Initiative (ICVCI'08)*, May 16–17 2008.

[4] T. Sterling and D. Stark, "A high-performance computing forecast: Partly cloudy," *Computing in Science Engineering*, vol. 11, no. 4, pp. 42 –49, jul. 2009.

[5] F. Doelitzscher, A. Sulistio, C. Reich, H. Kuijs, and D. Wolf, "Private Cloud for Collaboration and e-Learning Services: from IaaS to SaaS," *Computing*, 2010, in print.

[6] OpenNebula, "Open-source toolkit for building a cloud," http://www.opennebula.org, 2010.

[7] A. Sulistio, F. Doelitzscher, and C. Reich, "Automated Virtual Machine Creation with On-Demand Software Installation," in *Computer Science Research and Technology*, J. P. Bauer, Ed. Nova Science, 2010, vol. 3, in print.

[8] A. HPC, http://aws.amazon.com/hpc-applications, July 2011.

[9] J. Ekanayake, X. Qiu, T. Gunarathne, S. Beason, and G. Fox, *High Performance Parallel Computing with Clouds and Cloud Technologies*. CRC Press (Taylor and Francis), 07/2010 2010.

[10] S. Akioka and Y. Muraoka, "HPC Benchmarks on Amazon EC2," in *Proceedings of the International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, Perth, Australia, Apr. 20–23 2010.

[11] B. Amedro, F. Baude, D. Caromel, C. Delb, I. Filali, F. Huet, E. Mathias, and O. Smirnov, "An efficient framework for running applications on clusters, grids, and clouds," in *Cloud Computing*, ser. Computer Communications and Networks, N. Antonopoulos and L. Gillam, Eds. Springer London, 2010, vol. 0, pp. 163–178.

[12] Shibboleth, http://shibboleth.internet2.edu, 2010.

[13] PowerDNS Server, http://www.powerdns.com, 2010.

[14] Torque, http://www.clusterresources.com/pages/products/torque-resource-manager.php, 2010.

[15] "Maui," http://www.clusterresources.com/pages/products/maui-cluster-scheduler.php, 2010.

[16] Nagios, "The Industry Standard in IT Infrastructure Monitoring," http://www.nagios.org, 2010.

[17] NagiosGrapher, https://www.nagiosforge.org/gf/project/ nagiosgrapher, 2010.

[18] D. Bailey, E. Barszcz, J. Barton, D. Browning, R. Carter, L. Dagum, R. Fatoohi, P. Frederickson, T. Lasinski, R. Schreiber, H. Simon, V. Venkatakrishnan, and S. Weeratunga, "The Nas Parallel Benchmarks," *International Journal of High Performance Computing Applications*, vol. 5, no. 3, pp. 63–73, September 1991.

[19] linsolve, http://archives.math.utk.edu/software/msdos/discrete. math/tslin/.html.