Integration of Heterogeneous HPC-clusters Using OpenStack Platform

Alexander Feoktistov, Ivan Sidorov, Vadim Sergeev,
Roman Kostromin, Vera Bogdanova

Matrosov Institute for System Dynamics and Control Theory of SB RAS, Irkutsk, Russia
{agf,ivan.sidorov,sergeev,kostromin,bvg}@icc.ru

Abstract. Supercomputers, clusters, servers, distributed computing environments of different purposes and other computer systems are actively used for high-performance computing. Of particular interest is the problem of integrating such systems in the cloud infrastructure within IaaS model. We present an approach to the integration and virtualization of heterogeneous clusters in a cloud computing environment. This approach uses the platform OpenStack for creation and management of the environment. We develop a specialized hypervisor shell for running virtual machines as an additional module for the OpenStack. This hypervisor shell allows the running of virtual machines in the cluster nodes without the need for significant changes in their middleware. The advantages of this approach compared to the known ones are the use of complex knowledge about computational processes for the resource allocation, and the capability for the "painless" integration of heterogeneous clusters with the preinstalled system software for computational management to the virtual cluster with the required configuration.

Keywords: high-performance computing, virtualization technologies, OpenStack, hypervisor shell.

1 Introduction

In today's world of technology, computational clusters have become an essential element used to carry out various problems in such fields as fundamental and applied sciences. The nature of the target problems calls for a variety of hardware and software platforms, architectures and communication environments of computer clusters which in turn significantly affects their performance, efficiency and reliability.

The proliferation and variety of cluster systems makes reasonable the combined usage of clusters to increase the total available computing power and cutting the mean waiting time for starting a job. The latter is achieved by using extended planning strategies which consider the variations in time of the loading of different clusters and at the same time the flexibility in sharing the resources [1]. Jobs could be redirected from the currently most loaded cluster to the idle cluster, and a higher priority and other privileges could be assigned to them.

There are different approaches to the creation of integrated cluster environments (ICE) [2]. Our contribution to the solution of this problem is the development of methods and tools for creating heterogeneous distributed computing environments for different purposes with multi-agent management of computations [3-6]. These developments are based on the interaction with traditional middleware Condor, Globus Toolkit, gLite, PBS Torque and other known systems. The creation of the environments was implemented in the Irkutsk supercomputer centre of SB RAS [7].

However, there remain many unresolved problems as follows:

- Providing in operating system of the allocated node the full set of libraries that are necessary to correctly launch and execute the instance of the distributed application whether it has the form of an executable file or a piece of program code;
- Ensuring the reliability and fault-tolerance for computing processes in the nodes of the environment;
- Difficulty of tracing the real causes of faults in computing processes in debugging applications due to a lack of direct access to the computational nodes;
- Security of the distributed computing due to a lack of reliable mechanisms of screening the executable files for the presence of malicious code.

One of the most promising ways to solve such problems is using virtualization techniques [8], which enable execution of the instances of the distributed and parallel applications in isolated environments with the required hardware and software characteristics. In this case, the physical nodes, which support the virtualization, can significantly differ in performance, hardware characteristics, types of their operating systems and other parameters.

We describe the new approach to creating the ICE using virtualization techniques. This approach has the following advantages: the possibility of the complex knowledge use about the job requests; resource characteristics and current state of the environment; the expertise of its administrators; and the capability for the "painless" integration of heterogeneous clusters with the preinstalled local resource managers PBS, SLURM or SGE to the virtual cluster with the required configuration.

2 Related Work

Nowadays, a wide variety of software tools that support resource virtualization exists [9]. In particular, they include the following popular tools:

- Docker for deployment and application management in a virtualization environment at the operating system level [10];
- QEMU for emulating hardware for different platforms [11];
- KVM for supporting virtualization in a Linux environment for processors based on x86 architecture [12];
- Xen for virtual machines with para-virtualization support for processors based on x86 architecture [13];

– VMware ESXi for enterprise-level virtualization, offered by VMware as a VMware vSphere component [14].

We have practical skills of these tools to use for solving the following problems:

– Developing containers for web services with Docker;
– Testing software on various hardware configurations with QEMU;
– Enabling virtual machines for the resource management with KVM;
– Creating virtual computer clusters of various configurations for debugging and testing parallel and distributed applications with XenServer;
– Providing virtual machines of the required configuration using VMware vSphere to users.

There are other software tools also. However, they are based on the above listed tools or highly tailored.

When the aforementioned tools are used, problems of the cluster resources virtualization, integration of heterogeneous computer clusters, and providing service-oriented interfaces for integrated cloud infrastructure require additional solutions.

The platforms OpenStack [15], Apache CloudStack [16], Eucalyptus [17] and Open Nebula [18] are package suites for creating cloud services according to the concept Infrastructure-as-a-Service (IaaS). The last three platforms are not widespread. In contrast, the platform OpenStack leads the field of the distributed computing virtualization.

The following are the advantages of OpenStack:

– Involvement of the major players in the cloud industry in the work with this platform;
– Wide range of software components and functional solutions, including service-oriented interface for users;
– Availability of extensive documentation covering components;
– Application of open standards;
– Availability of APIs for interoperability with external software;
– Support of work with different hypervisors (KVM, XEN, ESXi, QEMU and others);
– Extensive capabilities for developing own solutions, etc.

We have selected the platform OpenStack for the ICE virtualization and the hypervisor KVM for launching virtual machines in cluster nodes. However, the selected software requires changing traditional system software in cluster nodes which is not desirable to do. To solve this problem, we have developed a specialized hypervisor shell for running virtual machines as an additional module for the OpenStack. This hypervisor shell allows the running of virtual machines in the cluster nodes without the need for significant changes in their system software.

3 Architecture of the Integrated Cluster Environment

The architecture of the ICE is presented in Fig. 1. It includes the following main components:

– Interfaces of an access level;
– Environment management node;
– Platform for the environment virtualization management;
– Specialized system software of a hypervisors level;
– Software and hardware of a computing resources level.

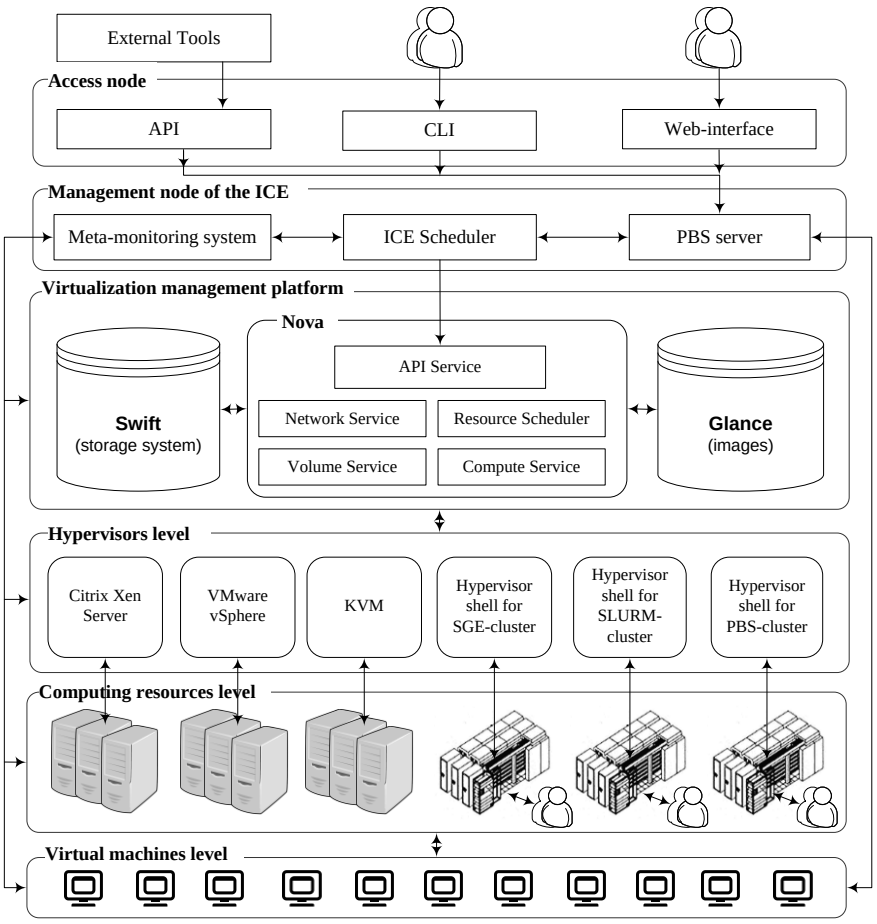

Fig. 1. Architecture of the integrated cluster environment

At the access level, there are three main interfaces for the interaction of different user categories with the environment.

The command line interface (CLI) provides the interaction with a server for man-

agement of virtual machines. We use the PBS Torque for its implementation.

Application programming interface (API) allows the external tools to interoperate with the environment to set up jobs, monitor them and get the results. API is based on the REST approach to creation of web services.

The job for the ICE is the specification of the resource requirements for the execution of parallel or distributed programs. The job specifies the following requirements: number of nodes and cores, RAM size, specialized accelerators (Intel Xeon Phi, GPU), interconnectors, disk space, operating system type and others.

Visual user interface includes the tools for adding jobs in the environment queues, and the tools for creating the virtual dedicated clusters of the required configurations similar to Amazon EC2 cloud services.

The following actions are carried out at the environment management node:

- Servicing the environment queues;
- Planning and distributing the requests for creating virtual machines;
- Managing pools of connected virtual machines created by virtualization management platform;
- Monitoring the environment components.

One of the key components of the ICE is the scheduler which processes jobs, allocates resources, interoperates with job queues of the PBS Torque, platform OpenStack and meta-monitoring system [19, 20]. The meta-monitoring system provides ICE scheduler extensive data on the state of the cluster queues, the state of physical and virtual nodes and other necessary information. The specialized algorithms for this system provide the reliability of distributed computing [21].

The ICE scheduler defines the number and characteristics of virtual machines taking into account user's requirements. It sends defined information to the virtualization management platform (OpenStack) for forming and launching jobs, represented by virtual machines, in cluster nodes.

Currently, we use the three main components of OpenStack: computing resources controller (Nova), cloud file storage (Swift) and library of virtual machines images (Glance). OpenStack process information received from ICE scheduler prepare virtual machine images and launch virtual machines by means of the traditional hypervisors and the special hypervisors shell developed within the proposed approach.

The special hypervisors shell provides the following important additional possibilities:

- Monitoring the current state of cluster queues to spot idle resources;
- Adding a job to the selected cluster queue for launching the virtual machines;
- Configuring the launched virtual machines;
- Monitoring the computations for virtual machines;
- Migrating virtual machines;
- Finishing the virtual machines and cleaning the cluster resources.

The local resource managers PBS Torque SLURM and SGE with open source code are used for the job flow management at the clusters.

4 Model of Resource Allocation for Virtual Machines

In this section, we describe a model of the resource allocation (queues selection) for virtual machines. The model is based on a classification of jobs for an execution of virtual machines in nodes (resources) of the environment.

Let us have a finite set $H = \{h_1, h_2, ..., h_k\}$ of job characteristics (requests). Each characteristic h_i is described by information structure including the following components:

– the domain D_i of values with the symbol θ of an uncertainty;
– the rank $r_i \geq 1$;
– the weight $w_i \geq 0$.

All elements of the set H are partially ordered descending in accordance with the ranks.

Denote by $C = \{c_1, c_2, ..., c_m\}$ a finite set of job classes. Each class c_j is defined by main (mandatory) and additional (optional) sets of characteristics. The characteristics of the main and additional sets are presented by the Boolean matrixes A and B of dimension $k \times m$. The matrix elements $a_{ij} = 1$ or $b_{ij} = 1$ mean that the characteristic h_i is a member of the main or additional set, used in the definition of the class c_j and has the specialized domain $D_{ij}^* \subseteq D_i \setminus \{\theta\}$ for this class. If the condition $a_{ij} \vee b_{ij} = 0$ is satisfied, then $D_{ij}^* \equiv \{\theta\}$.

The matrixes A and B must satisfy the following conditions: $\bigvee_{j=1}^{m} \bigwedge_{i=1}^{k} \overline{a_{ij}} = 0$ and $\bigvee_{i=1}^{k} \bigvee_{j=1}^{m} (a_{ij} \wedge b_{ij}) = 0$.

An administrator of the environment forms the sets of characteristics and classes, and defines an accordance of resources to job classes.

The scheduler automatically determines characteristics used in a job specification and their domains for the job execution.

Denote by x a Boolean vector of dimension k for the job characteristics representation. Between elements of the vector x and indexes of the characteristics there is bijection. Element values of the vector x are defined as follows:

$$x_i = \begin{cases} 0, if\ D'_i \equiv \{\theta\}, \\ 1, if\ D'_i \subseteq D_i \setminus \{\theta\}, \end{cases}$$

where D'_i is the domain of the characteristic h_i requested for the job execution, $i \in \{1, 2, ..., k\}$. The vector x must satisfy the following condition: $\bigwedge_{i=1}^{k} \overline{x_i} = 0$.

We use the characteristic function $\chi(x)$ to check the domain D'_i of the charac-

teristic h_i on an accordance to the domain D_{ij}^* of this characteristic of the class c_j. This function is defined as follows:

$$\chi_j(x) = \begin{cases} 0, if\ \exists i : (a_{ij} \vee b_{ij} = 1) \wedge (D'_i \subseteq D_{ij}^*), \\ 1\, otherwise, \end{cases}$$

where $i \in \{1, 2, ..., k\}$, $j \in \{1, 2, ..., m\}$.

The function $\chi(x)$ allows fulfilment of a primary job classification. As the result of this classification, the job can relate to several classes. In this instance, we use additional information such as the probabilistic measure of belonging to class, ranks and weights of characteristics, and computational history of jobs for more detailed classification.

Define the functions $\rho_j(x)$, $\sigma_j(x)$, $\omega_j(x)$ и $\phi_j(x, z)$, which calculate the following parameters of the characteristics relatively the class c_j:

- the probabilistic measure of belonging to class;
- the aggregated rank;
- the summary weight;
- the probabilistic measure of belonging to class taking into account the computational history z of jobs.

These functions allow implementation of the various variants of the additional job classification based on the primary one. The job classification is intended for the primary filtration of the resources set. It provides forming the residual set V of resources for the job execution. The further filtration of resources from the set V is implemented with the help of multi-criterion methods of choice.

5 Experimental Results

In order to estimate practicability and benefits of the proposed approach, we performed simulation modelling of the ICE using the GPSS World system [22].

Simulation modelling was performed on the job flow corresponding to the flow of real jobs executed on the cluster Blackford of the Irkutsk computer centre. We processed 62.249 jobs in total.

We modelled dedicated and non-dedicated nodes of the cluster in various proportions. The dedicated nodes execute virtual machines and not execute jobs from common queues of the local resource managers. The non-dedicated nodes execute all jobs, including jobs for executing virtual machines.

The traditional manager of a virtual environment does not accept jobs for execution of virtual machines whose requirements exceed the possibilities of environment nodes. In contrast, the hypervisor shell allows decomposing such jobs and passing its part to non-dedicated nodes.

Fig. 2 and Fig. 3 show correspondingly the utilization of dedicated and non-dedicated nodes under management using the hypervisor shell (case 2) and without it (case 1).

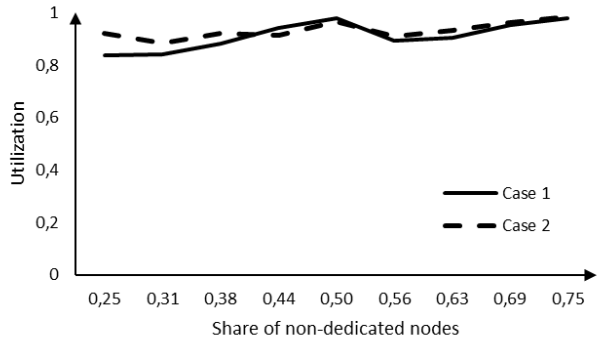

Fig. 2. Utilization of dedicated nodes

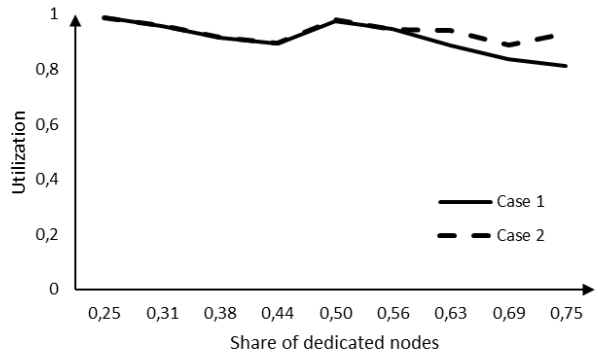

Fig. 3. Utilization of non-dedicated nodes

The modelling results show that the hypervisor shell allows to somewhat improve the utilization of dedicated nodes by means of allocation of non-dedicated nodes for executing virtual machines. Therefore, the utilization of non-dedicated nodes also increases in many cases.

In the future, we will study the ICE in more detail in the process of its practical use.

6 Conclusion

In this paper, we address a problem of the distributed computing virtualization. We give a brief overview of the traditional tools for the management of virtual machine and highlight the advantages of the platform OpenStack for an interaction with these hypervisors.

Our contribution is multifold. We propose the approach for the heterogeneous cluster environment virtualisation using the platform OpenStack jointly with the new specialized hypervisor shell for local resource managers such as PBS, SLURM or SGE. Within this approach, we developed the model of the resource allocation for virtual machines and realized the tools for the integration of heterogeneous clusters.

The developed model of the resource allocation for virtual machines allows us jointly to use the knowledge about the job requests, resource characteristics and current state of the environment, and the expertise of it administrators.

The realized tools provide the capability for the "painless" integration of heterogeneous clusters with the preinstalled local resource managers listed above to the virtual cluster with the required configuration.

The proposed approach is being developed in the Irkutsk computer center and its practicability and benefits are being demonstrated on the model example.

Acknowledgements. The study was partially supported by RFBR, projects no. 15-29-07955 and no. 16-07-00931.

References

1. Gergel, V., Senin, A.: Metacluster System for Managing the HPC Integrated Environment. In: Ching-Hsien, H., Malyshkin, V. (eds.) MTPP 2010. LNCS, vol. 6083. pp. 86-94. Springer, Heidelberg (2011)
2. Mladen, A., Eric, S., Patrick, D.: Integration of High-Performance Computing into Cloud Computing Services. In: Handbook of Cloud Computing, pp. 255-276. Springer, Heidelberg (2010)
3. Bychkov, I.V., Oparin, G.A., Novopashin, A.P., Feoktistov, A.G., Korsukov, A.S., Sidorov, V.A. High-performance computing resources of ISDCT SB RAS: State-of-the-art, prospects and future trends. Comput. Tech. 15, 69-81 (2010)
4. Bogdanova, V.G., Bychkov, I.V., Korsukov, A.S., Oparin, G.A., Feoktistov, A.G.: Multiagent Approach to Controlling Distributed Computing in a Cluster Grid System. J. Comput. Syst. Sci. Int. 53, 713–722 (2014)
5. Bychkov, I.V., Oparin, G.A., Feoktistov, A.G., Bogdanova, V.G., Pashinin, A.A.: Service-oriented multiagent control of distributed computations. Automat. Rem. Contr. 76, 2000–2010 (2015)
6. Bychkov, I.V., Oparin, G.A., Feoktistov, A.G., Sidorov, I.A., Bogdanova, V.G., Gorsky, S.A.: Multiagent control of computational systems on the basis of meta-monitoring and imitational simulation. Optoelectron., Instr. and Data Process. 52, 107–112 (2016)
7. Irkutsk Supercomputer Centre of SB RAS, `http://hpc.icc.ru`
8. Buyya, R., Broberg, J., Goscinski, A.M.: Cloud Computing: Principles and Paradigms. Wiley (2011)

9. Sridharan, S.: A Performance Comparison of Hypervisors for Cloud Computing. University of North Florida (2012)
10. Docket, http://docker.com
11. QEMU, http://qemu.org
12. KVM, http://www.linux-kvm.org/
13. Xen, http://cam.ac.uk/research/srg/netos/projects/archive/xen/
14. vSphere ESXi, https://vmware.com/support/vsphere-hypervisor.html
15. Bumgardner, V.K.: OpenStack in Action. Manning Publications (2016)
16. Apache CloudStack, https://cloudstack.apache.org/
17. Euacalyptus, http://www.eucalyptus.com/
18. OpenNebula, https://opennebula.org
19. Bichkov, I.V., Oparin, G.A., Novopashin, A.P., Sidorov, I.A.: Agent-Based Approach to Monitoring and Control of Distributed Computing Environment. In: Malyshkin, V. (ed.) PaCT 2015. LNCS, vol. 9251. pp. 253–257. Springer, Heidelberg (2015)
20. Sidorov, I.A.: Methods and tools to increase fault tolerance of high-performance computing systems. In: Proc. of the 39th International Convention MIPRO, pp. 242–246 (2016)
21. Feoktistov, A.G, Sidorov, I.A.: Logical-Probabilistic Analysis of Distributed Computing. In: Proc. of the 39th International Convention MIPRO, pp. 247–252 (2016)
22. GPSS World Tutorial Manual, http://www.minutemansoftware.com/