DOI: 10.14529/cmse170203

VIRTUALIZATION OF HETEROGENEOUS HPC-CLUSTERS BASED ON OPENSTACK PLATFORM*

© 2017 A.G. Feoktistov, I.A. Sidorov, V.V. Sergeev,
R.O. Kostromin, V.G. Bogdanova
Matrosov Institute for System Dynamics and Control Theory of SB RAS
(Lermontova str., 134, Irkutsk, 664033, Russia)
E-mail: agf@icc.ru, ivan.sidorov@icc.ru, vvsergeev@mail.ru, kostromin@icc.ru, bvg@icc.ru
Received: 01.05.2017

The paper addresses to the problem of integration of heterogeneous computing clusters to the united environment based on a virtualization technology. OpenStack software is selected as a platform for managing the virtual environment. The OpenStack platform provides a wide range of components and solutions to a functional interaction with different hypervisors. These include KVM, XEN, ESXi, QEMU and other systems. In addition to the OpenStack platform, we developed a specialized hypervisor shell. It helps to start virtual machines using queues of the traditional resource management systems, such as PBS, SLURM, LSF, or SGE, that are used on clusters of a center of collective usage. The developed model of the resource allocation for virtual machines allowed us to use the knowledge about 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 for creating the virtual cluster with the required configuration. Extensive modeling shows that the hypervisor shell can improve efficiency of integrated environment nodes through reallocating virtual machines to queues of the traditional resource management systems.

Keywords: computer clusters, HPC, virtualization technologies, OpenStack, simulation modeling

FOR CITATION

Feoktistov A.G., Sidorov I.A., Sergeev V.V., Kostromin R.O., Bogdanova V.G. Virtualization of Heterogeneous HPC-clusters Based on OpenStack Platform. *Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering.* 2017. vol. 6, no. 2. pp. 37–48. DOI: 10.14529/cmse170203.

Introduction

In the field of high-performance computing (HPC), computational clusters have become an essential element used to carry out various large problems within fundamental and applied studies. The nature of the target problems calls for a variety of hardware and software platforms, architectures and communication environments of HPC-clusters, which in turn significantly affects their performance, efficiency and reliability.

A variety of cluster systems makes the combined usage of HPC-clusters reasonable as it increases the total available computing power and reduces 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 can be redirected from the currently most loaded cluster to the idle cluster, and a higher priority and other privileges can be assigned to them.

* The paper is recommended for publication by the Program Committee of the International Scientific Conference "Parallel Computational Technologies (PCT) 2017".

Jobs specify processes of problem solving. They include the information about the required computational resources, executable programs, input and output data, and other required items.

We assume that the integrated cluster environment (ICE) is based on the cluster systems that differ in their computational characteristics of nodes and parameters of administrative policies. There are different approaches to the creation of the 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 the traditional middleware Condor, Globus Toolkit, gLite, PBS Torque and other known systems. The creation of the environments was implemented in the Center of collective usage SB RAS "Irkutsk supercomputer center" [7].

Nevertheless, 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 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.

In this regard, we describe the new approach to creating the ICE using virtualization techniques. This approach has the following advantages: the possibility of using complex knowledge about 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.

1. Related work

In this section, we give a brief overview of software for a distributed computing virtualization. There is a wide variety of software tools that support resource virtualization [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 [12];
– Xen for virtual machines with para-virtualization support [13];
– VMware ESXi for enterprise-level virtualization, offered by VMware as a VMware vSphere component [14].

We have practical skills in using these tools 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 users with virtual machines of the required configuration using VMware vSphere.

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

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

There are following advantages of OpenStack:

- Involvement of 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 OpenStack platform 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 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 running virtual machines in the cluster nodes without the need for significant changes in their system software.

2. Architecture of the integrated cluster environment

We propose an architecture of the ICE that includes the following main components:

- Interfaces of an access level and the environment management node (Fig. 1);
- Platform for the environment virtualization management, specialized system software of a hypervisors level and software, and hardware of a computing resources level (Fig. 2).
- 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 management 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.

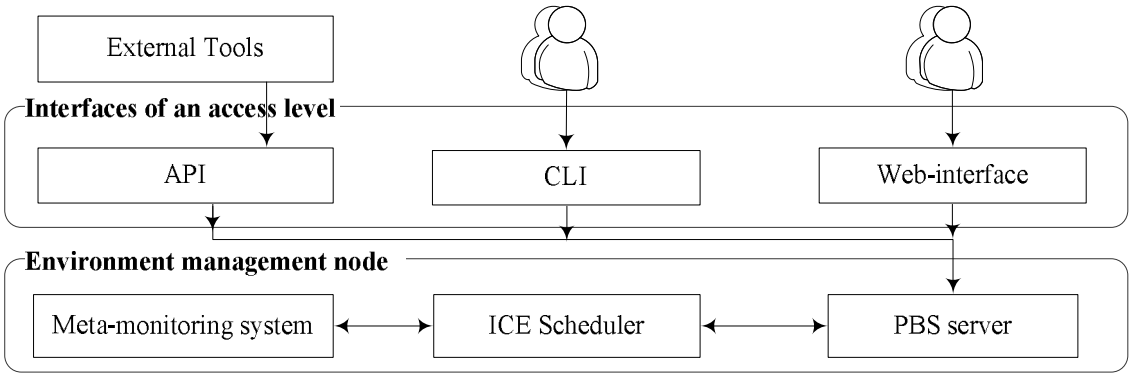


Fig. 1. Architecture of the access level and environment management node

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.

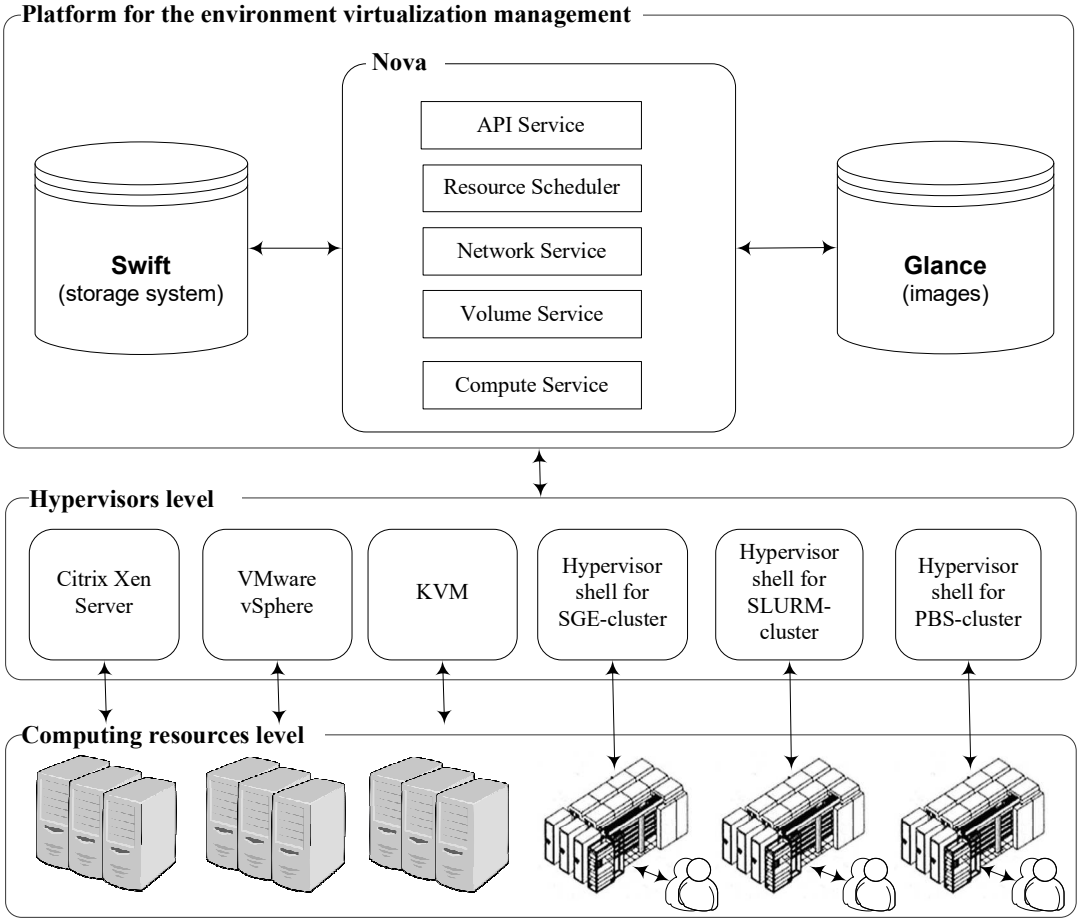


Fig. 2. Architecture of the ICE virtualization

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, OpenStack platform and meta-monitoring system [19, 20]. The meta-monitoring system provides ICE scheduler with 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 machine images (Glance). The OpenStack processes information received from ICE scheduler, prepares virtual machine images and launches virtual machines by means of the traditional hypervisors and special hypervisors shell developed within the proposed approach.

The special hypervisor 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 in the clusters.

3. 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 the classification of jobs for an execution of virtual machines in nodes (resources) of the environment. The proposed job classification is based on the feature vector method [22].

Let us have a finite set $H = \{h_1, h_2, ..., h_k\}$ of job characteristics (requests). They include problem solving time, RAM and disk space size, number of nodes, processors and cores, program libraries, compilers, their keys, etc.

Each characteristic h_i is described by information structure, including the following components:
– Domain D_i of values with the symbol θ of uncertainty;
– Rank $r_i \geq 1$;
– Weight $w_i \geq 0$.

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

Let us denote by $C = \{c_1, c_2, ..., c_m\}$ a finite set of job classes. Each class c_j is defined by the 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 have satisfied 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.

Let us denote by x a Boolean vector of dimension k for the job characteristics representation. There is bijection between elements of the vector x and indexes of the characteristics. Element values of the vector x are defined as follows:

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

where D'_i is the domain of the characteristic h_i requested for the job execution, $i \in \{1, 2, ..., k\}$. The vector x has satisfied 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 characteristic h_i for an accordance to the domain D_{ij}^* of this characteristic of the class c_j. This function is defined as follows:

$$\chi_j(x) = \begin{cases} 0, \text{ if } \exists i : (a_{ij} \vee b_{ij} = 1) \wedge (D'_i \not\subseteq D_{ij}^*), \\ 1 \text{ otherwise}, \end{cases}$$

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

The function $\chi(x)$ allows fulfilling of a primary job classification. As the result of this classification, the job can be related 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.

Let us 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 relative to the class c_j:

- Probabilistic measure of belonging to class;
- Aggregated rank;
- Summary weight;
- Probabilistic measure of belonging to the class, taking into account the computational history z of jobs.

Let us denote by the symbols δ_ρ, δ_σ, δ_ω and δ_ϕ the upper limits of an equivalence for the functions ρ, σ, ω and ϕ, respectively. When the absolute difference of two values for one of the functions ρ, σ, ω or ϕ is less or equal to the upper limit of its equivalence δ_ρ, δ_σ, δ_ω or δ_ϕ, then these values are equivalent.

Let us define the following characteristic functions:

$$\chi_j^\rho(x,y)=\begin{cases}0, \text{ if } \max\limits_{\forall l:y_l=1}\{\rho_l(x)\}-\rho_j(x)>\delta_\rho,\\ 1 \text{ otherwise,}\end{cases}$$

$$\chi_j^\sigma(x,y)=\begin{cases}0, \text{ if } \max\limits_{\forall l:y_l=1}\{\sigma_l(x)\}-\sigma_j(x)>\delta_\sigma,\\ 1 \text{ otherwise,}\end{cases}$$

$$\chi_j^\omega(x,y)=\begin{cases}0, \text{ if } \max\limits_{\forall l:y_l=1}\{\omega_l(x)\}-\omega_j(x)>\delta_\omega,\\ 1 \text{ otherwise,}\end{cases}$$

$$\chi_j^\phi(x,y,z)=\begin{cases}0, \text{ if } \max\limits_{\forall l:y_l=1}\{\phi_l(x,z)\}-\phi_j(x,z)>\delta_\phi,\\ 1 \text{ otherwise,}\end{cases}$$

where $y_j=\chi_j(x)$, $j\in\{1,2,...,m\}$.

These functions allow implementing 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 of resources for the job execution. The further filtration of resources from the set V is implemented with help of the lexicographical or majority methods of choice [23].

We use the lexicographic method with the following modified rule of choice:

$$V^*=\{v_s:(\forall v_l\in V\ \exists p:(\hat{q}_{1,s}=\hat{q}_{1,l})\wedge...\wedge(\hat{q}_{p,s}=\hat{q}_{p,l})\wedge(\hat{q}_{p+1,s}>\hat{q}_{p+1,l}))\},$$

where V^* is the filtered set of resources, $v_s\in V$, n_v is the number of resources, q_{js} is the characteristic of the sth resource, $\hat{q}_{js}$ is its estimation, n_q is the number of compared characteristics, $p\in\{1,2,...,n_q-1\}$, $j=1,2,...,n_q$, $s\in\{1,2,...,n_v\}$, $l\in\{1,2,...,n_v\}$ and $v\neq l$.

We use the majority method with the following modified rule of choice:

$$V^*=\left\{v_s:\left(\neg\exists v_l\in V:\sum_{j=1}^{n_q}\text{sign}(\hat{q}_{jl}-\hat{q}_{js})>0\right)\right\},$$

where $\text{sign}(0)=0$.

To estimate the values q_{js}, $s=1,2,...,n_v$, their set is divided into subsets that do not intersect pairwise. They are ordered by ascending or descending. Accordingly, each subset receives its index used as an estimate of the values belonging to the given subset. If the resulting set V^* contains more than one element, then the final choice of the single resource v_s is done randomly.

4. Experimental analysis

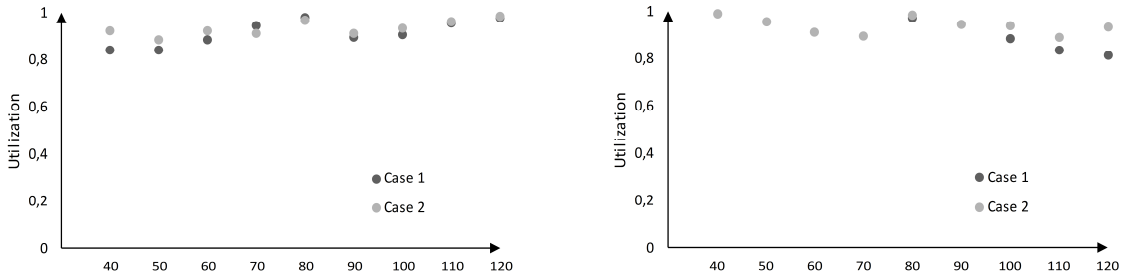
Within experimental analysis, we performed simulation modeling of the ICE using the GPSS World system [24]. The examples of the utilization of ICE nodes using the hypervisor shell and without it are given bellow.

Simulation modeling was performed on the job flow corresponding to the flow of real jobs executed on the cluster that includes 20 nodes. Each node has two quad-core processors Intel Xeon E5345. We processed over 60000 jobs in total.

We varied dedicated and non-dedicated nodes of the cluster in various proportions. The dedicated nodes execute virtual machines and do 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. 3 show the utilization of dedicated and non-dedicated nodes under management, respectively, using the hypervisor shell (a) and without it (b).



a) dedicated nodes b) non-dedicated nodes

Fig. 3. Utilization of nodes

The represented results show that the hypervisor shell allows improving 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.

Conclusion

In this paper, the approach to the integration and virtualization of heterogeneous clusters in a cloud computing environment was presented. A brief overview of the traditional tools for the management of virtual machines was given; the advantages of the OpenStack platform for an interaction with these hypervisors were highlighted.

Our contribution is multifold. We proposed the technology for the heterogeneous cluster environment virtualization using the OpenStack platform together 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 allowed us to use the knowledge about 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 for creating the virtual cluster with the required configuration.

The practicability and benefits of the represented approach were demonstrated on the model example. All experiments were carried out in the Center of collective usage SB RAS "Irkutsk supercomputer center".

The study was partially supported by RFBR, projects no. 15-29-07955 and no. 16-07-00931, and Program 1.33P of fundamental research of Presidium RAS, project "Development of new approaches to creation and study of complex models of information-computational and dynamic systems with applications".

References

1. Gergel V., Senin A. Metacluster System for Managing the HPC Integrated Environment. Methods and Tools of Parallel Programming Multicomputers. Second Russia-Taiwan Symposium, MTPP 2010 (Vladivostok, Russia, May 16–19 2010). LNCS, vol. 6083. pp. 86–94. DOI: 10.1007/978-3-642-14822-4
2. Mladen A., Eric S., Patrick D. Integration of High-Performance Computing into Cloud Computing Services. Handbook of Cloud Computing. 2010. pp. 255-276. DOI: 10.1007/978-1-4419-6524-0_11
3. Bychkov I.V., Oparin G.A., Novopashin A.P., Feoktistov A.G., Korsukov A.S., Sidorov I.A. High-performance computing resources of ISDCT SB RAS: State-of-the-art, prospects and future trends. Comput. Tech. 2010. vol. 15. pp. 69–81. (in Russian).
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. 2014. vol. 53. pp. 713–722. DOI:10.1134/S1064230714040030
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. 2015. vol. 76. pp. 2000–2010. DOI: 10.1134/S0005117915110090
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. 2016. vol. 52, pp. 107–112. DOI: 10.3103/S8756699016020011
7. Irkutsk Supercomputer Center of SB RAS. Available at: http://hpc.icc.ru (accessed: 16.02.2017).
8. Buyya R., Broberg J., Goscinski A.M. Cloud Computing: Principles and Paradigms. Wiley, 2011. 637 p. DOI: 10.1002/9780470940105
9. Sridharan S. A Performance Comparison of Hypervisors for Cloud Computing. University of North Florida, 2012. 269 p.
10. Docker. Available at: http://docker.com (accessed: 16.02.2017).
11. QEMU. Available at: http://qemu.org (accessed: 16.02.2017).
12. KVM. Available at: http://www.linux-kvm.org (accessed: 16.02.2017).
13. Xen. Available at: http://cam.ac.uk/research/srg/netos/projects/archive/xen (accessed: 16.02.2017).
14. vSphere ESXi. Available at: https://vmware.com/support/vsphere-hypervisor.html (accessed: 16.02.2017).
15. Bumgardner V.K. OpenStack in Action. Manning Publications, 2016. 358 p.
16. Apache CloudStack. Available at: https://cloudstack.apache.org/ (accessed: 16.02.2017).

17. Euacalyptus. Available at: http://www.eucalyptus.com/ (accessed: 16.02.2017).
18. OpenNebula. Available at: https://opennebula.org (accessed: 16.02.2017).
19. Bichkov I.V., Oparin G.A., Novopashin A.P., Sidorov I.A. Agent-Based Approach to Monitoring and Control of Distributed Computing Environment. Parallel Computing Technologies: 13th International Conference, PaCT 2015 (Petrozavodsk, Russia, August 31-September 4). LNCS, vol. 9251. pp. 253–257. DOI: 10.1007/978-3-319-21909-7_24
20. Sidorov I.A. Methods and Tools to Increase Fault Tolerance of High-performance Computing Systems. In proc. of the 39th International Convention on information and communication technology, electronics and microelectronics, MIPRO-2016 (Opatija, Croatia, 30 May – 3 June 2016). Riejka: CSICTEM 2016. pp. 242–246. DOI: 10.1109/MIPRO.2016.7522142
21. Feoktistov A.G, Sidorov I.A. Logical-Probabilistic Analysis of Distributed Computing. In proc. of the 39th International Convention on information and communication technology, electronics and microelectronics, MIPRO-2016 (Opatija, Croatia, 30 May-3 June 2016). Riejka: CSICTEM 2016. pp. 247–252. DOI: 10.1109/MIPRO.2016.7522142
22. Hastie T., Tibshirani R., Friedman J. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2001, 533 p.
23. Sholomov L.A. *Logicheskie metodi issledovaniya diskretnih modelei vibora* [Logical Research Methods of Discrete Choice Models]. Moscow: Nauka, 1989, 288 p. (in Russian)
GPSS World Tutorial Manual. Available at: http://www.minutemansoftware.com (accessed: 16.02.2017).

УДК 004.45, 004.386, 004.382.2 **DOI: 10.14529/cmse170203**

ВИРТУАЛИЗАЦИЯ HPC-КЛАСТЕРОВ С ПОМОЩЬЮ ПЛАТФОРМЫ OPENSTACK

© 2017 г. **А.Г. Феоктистов, И. А. Сидоров, В.В. Сергеев, Р.О. Костромин, В.Г. Богданова**

Институт динамики систем и теории управления имени В.М. Матросова СО РАН (664033, Иркутск, Лермонтова, 134),

E-mail: agf@icc.ru, ivan.sidorov@icc.ru, vvsergeev@mail.ru, kostromin@icc.ru, bvg@icc.ru

Поступила в редакцию: 01.05.2017

Статья посвящена проблеме интеграции разнородных вычислительных кластеров в единую среду на основе технологий виртуализации. В качестве платформы управления виртуальной средой выбран программный комплекс OpenStack. Данный комплекс обеспечивает широкий набор компонентов и функциональных решений для взаимодействия с различными гипервизорами. В их числе KVM, XEN, ESXi, QEMU и другие системы. В дополнение к программному комплексу OpenStack разработана специализированная оболочка гипервизора, обеспечивающая запуск виртуальных машин из очередей традиционных систем управления заданиями, например, PBS, SLURM, LSF или SGE, используемых на кластерах суперкомпьютерного центра коллективного пользования. Разработанная модель распределения ресурсов для виртуальных машин позволяет использовать знания о заданиях, характеристиках ресурсов и текущем состоянии виртуальной вычислительной среды, а также опыт ее администраторов. Реализованные инструменты обеспечивают возможность «безболезненной» интеграции разнородных вычислительных кластеров, использующих различные системы управления заданиями, в виртуальный кластер с требуемой конфигурацией. Модельные эксперименты показывают, что оболочка гипервизора позволяет повысить коэффициент полезного использования узлов интегрированной среды путем переназначения виртуальных машин в очереди традиционных систем управления заданиями.

Ключевые слова: вычислительные кластеры, высокопроизводительные вычисления, технологии виртуализации, OpenStack, имитационное моделирование

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Feoktistov A.G, Sidorov I.A., Sergeev V.V., Kostromin R.O., Bogdanova V.G. Virtualization of heterogeneous HPC-clusters based on OpenStack platform // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2017. Т. 6, № 2. С. 37–48. DOI: 10.14529/cmse170203.

Литература

1. Gergel V., Senin A. Metacluster System for Managing the HPC Integrated Environment // Methods and Tools of Parallel Programming Multicomputers. Second Russia-Taiwan Symposium, MTPP 2010 (Vladivostok, Russia, May 16-19 2010). LNCS, Vol. 6083. P. 86–94. DOI: 10.1007/978-3-642-14822-4
2. Mladen A., Eric S., Patrick D. Integration of High-Performance Computing into Cloud Computing Services // Handbook of Cloud Computing. 2010. P. 255–276. DOI: 10.1007/978-1-4419-6524-0_11
3. Бычков И.В., Опарин Г.А., Новопашин А.П., Феоктистов А.Г, Корсуков А.С., Сидоров И.А. Высокопроизводительные вычислительные ресурсы Института динамики систем и теории управления со ран: текущее состояние, возможности и перспективы развития // Вычислительные технологии 2010. Т. 15. С. 69–81.
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. 2014. Vol. 53. P. 713–722. DOI:10.1134/S1064230714040030
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. 2015. Vol. 76. P. 2000–2010. DOI: 10.1134/S0005117915110090
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. 2016. Vol. 52, P. 107–112. DOI: 10.3103/S8756699016020011
7. Иркутский суперкомпьютерный центр СО РАН. URL: http://hpc.icc.ru (accessed: 16.02.2017).
8. Buyya R., Broberg J., Goscinski A.M. Cloud Computing: Principles and Paradigms. Wiley, 2011. 637 p. DOI: 10.1002/9780470940105
9. Sridharan S. A Performance Comparison of Hypervisors for Cloud Computing. University of North Florida, 2012. 269 p.
10. Docker. URL: http://docker.com (дата обращения: 16.02.2017).
11. QEMU. URL: http://qemu.org (дата обращения: 16.02.2017).
12. KVM. URL: http://www.linux-kvm.org (дата обращения: 16.02.2017).
13. Xen. URL: http://cam.ac.uk/research/srg/netos/projects/archive/xen (дата обращения: 16.02.2017).
14. vSphere ESXi. URL: https://vmware.com/support/vsphere-hypervisor.html (дата обращения: 16.02.2017).
15. Bumgardner V.K. OpenStack in Action. Manning Publications, 2016. 358 p.
16. Apache CloudStack. URL: https://cloudstack.apache.org/ (дата обращения: 16.02.2017).
17. Euacalyptus. URL: http://www.eucalyptus.com/ (дата обращения: 16.02.2017).
18. OpenNebula. URL: https://opennebula.org (дата обращения: 16.02.2017).

19. Bichkov I.V., Oparin G.A., Novopashin A.P., Sidorov I.A. Agent-Based Approach to Monitoring and Control of Distributed Computing Environment // Parallel Computing Technologies: 13th International Conference, PaCT 2015 (Petrozavodsk, Russia, August 31 – September 4 2015). LNCS, Vol. 9251. P. 253–257. DOI: 10.1007/978-3-319-21909-7_24
20. Sidorov I.A. Methods and Tools to Increase Fault Tolerance of High-performance Computing Systems // In proc. of the 39th International Convention on information and communication technology, electronics and microelectronics, MIPRO-2016 (Opatija, Croatia, 30 May-3 June 2016). Riejka: CSICTEM 2016. P. 242–246. DOI: 10.1109/MIPRO.2016.7522142
21. Feoktistov A.G, Sidorov I.A. Logical-Probabilistic Analysis of Distributed Computing // In proc. of the 39th International Convention on information and communication technology, electronics and microelectronics, MIPRO-2016 (Opatija, Croatia, May 30 – June 3 2016). Riejka: CSICTEM 2016. P. 247–252. DOI: 10.1109/MIPRO.2016.7522142
22. Hastie T., Tibshirani R., Friedman J. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2001, 533 p.
23. Шоломов Л.А. Логические методы исследования дискретных моделей выбора. М.: Наука, 1989. 288 с.
24. GPSS World Tutorial Manual. URL: http://www.minutemansoftware.com (дата обращения: 16.02.2017).

Феоктистов Александр Геннадьевич, к.ф.-м.н., доцент, лаборатория параллельных и распределенных вычислительных систем, Институт динамики систем и теории управления им. В.М. Матросова СО РАН (Иркутск, Российская Федерация)

Сидоров Иван Александрович, к.ф.-м.н., доцент, лаборатория параллельных и распределенных вычислительных систем, Институт динамики систем и теории управления им. В.М. Матросова СО РАН (Иркутск, Российская Федерация)

Сергеев Вадим Викторович, аспирант, лаборатория параллельных и распределенных вычислительных систем, Институт динамики систем и теории управления им. В.М. Матросова СО РАН (Иркутск, Российская Федерация)

Костромин Роман Олегович, аспирант, лаборатория параллельных и распределенных вычислительных систем, Институт динамики систем и теории управления им. В.М. Матросова СО РАН (Иркутск, Российская Федерация)

Богданова Вера Геннадьевна, к.ф.-м.н., доцент, лаборатория параллельных и распределенных вычислительных систем, Институт динамики систем и теории управления им. В.М. Матросова СО РАН (Иркутск, Российская Федерация)