Virtual Machine Migration Strategy in Federated Cloud

Yanjue XU† Yuji SEKIYA‡

† Faculty of Engineering, University of Tokyo 2-11-16 Yayoi, Bunkyo-ku, Tokyo, 113-8658 Japan

E-mail: † yanjue@cnl.k.u-tokyo.ac.jp, ‡ sekiya@wide.ad.jp

Abstract Cloud computing and the as-a-service paradigm have gained a lot of interest recently. At the same time, virtualization technology has been shown to be an attractive path to increase overall system resource utilization by its powerful management mechanism such as isolating resources schedulers, suspend/resume, and virtual machine(VM) live migration.

In this paper we present a new VM migration strategy using VM live migration technology in the federated cloud environment. This method will be used to detect the overloaded servers and initiate the migration to the optimized location in the cloud automatically, thus eliminating the hotspots and balancing the load not only CPU load, including memory and network utilization. According to the experimental result, our technique has been proven that it can detect and remove the hotspots efficiently and balance the load.

Keyword Virtual Machine, Live Migration, Dynamic Resource Allocation, Load Balancing

1. Introduction

The pay-per-use model in the infrastructure as a service (IaaS) paradigm in cloud computing offers the ability to scale up compute and storage resources on demand. The Amazon Elastic Compute Cloud (Amazon EC2) is the best-known example of this paradigm of elastic capacity provisioning.

By introducing virtualization to the clouds, users can be isolated with each other while sharing the same physical machine in the public cloud. Furthermore, Virtual Machine (VM) independence from hardware and support for heterogeneous software stacks has exempted cloud users from manual configuration. And the use of live VM migration technology has enabled more effective sharing of system resources across multiple physical severs.

In the cloud environment, the workload of servers will fluctuate due to the incremental growth, time-of-day effects, and flash crowds. When a server is overloaded, Service Level Agreement (SLA) will start to degrade, which leads, for instance, that the response time of the request from the user will become longer. Therefore, how to allocate the virtual resources dynamically has become a widely concerned problem of cloud computing.

Some researches about dynamic allocation of resources of grid computing have been working on allocating the processes to the CPU resources, rather than consider VM as an individual unit[1]. So it is difficult to directly apply then to the virtualized servers. The widely used cloud manager, such as OpenNebula[11], has solved the problem of initial provisioning of the virtual resources, but doesn't consider the automatic dynamic allocation while the workloads and demands of VMs fluctuate in real time using migration technology. The cloud user has to detect the hotspot and migrate it to a less loaded server manually. VMware DRS adopts the live VM migration technology to manage the operational cost of the cloud[6][10]. And since it's a commercial product of VMware company, the information of the mechanism of it is not available, and it can be only applied to the VMs running on the hypervisor from VMware.

In order to address these problems, this paper proposes a dynamic virtual resources allocation mechanism using live VM migration. This approach automates tasks of monitoring the current load of servers and VMs, detecting the hotspots, deciding the best physical location of the busy VM, and initiating the migration. According to the experimental result, our technique has been proven that it can detect and remove the hotspots efficiently in the mostly under-loaded cloud, and balance the load in the mostly overloaded cloud.

This paper is structured as follows: Section 2 presents the research background and system overview. In section 3, the results of experiments are showed and analyzed. And we will discuss about the current work and future work in section 4.

2. Background and Problems

In this section the background of cloud computing and methods of resource management are described.

2.1. Federated Cloud

There were two types of clouds: public cloud and private cloud. Public cloud or external cloud describes cloud computing in the traditional mainstream sense, whereby resources are dynamically provisioned by an off-site third-party provider. Private cloud and internal cloud are resources available on private networks.

However, one set of cloud services is not going to be able to serve all the needs of a customer. Instead, a more federated approach where the needs of an organization are serviced by multiple clouds is needed to the end users. Nowadays, there is a new type called federated cloud has drew the attentions, in which independent providers contribute resources to a shared pool.

In the federated cloud environment, Physical servers are belonging to different infrastructure providers and located in different networks, and all the VMs serve as individual servers running multiple applications, which are unknown to the cloud manager. Therefore, effectively management of the virtual resources across the whole cloud without knowing the detail of the services VMs are providing, while meeting the SLA became a complex task[5].

2.2. Live VM Migration

The locations of physical machines of VMs are in various parts of network. Live migration allows a server administrator to move a running virtual machine between different physical machines while continuously running, without any noticeable effects from the point of view of the end users.

For performance sensitive applications, VM live migration offers great benefits on optimizing the utilization of available resources[9].

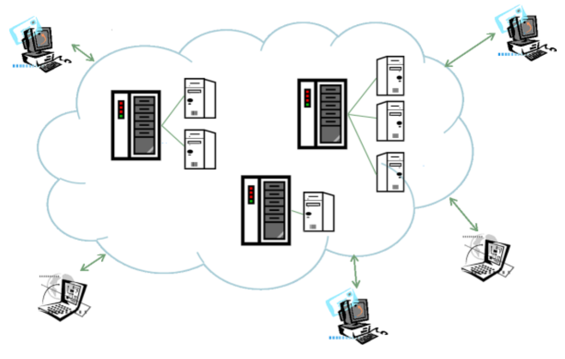

Fig. 1: diagram of cloud computing

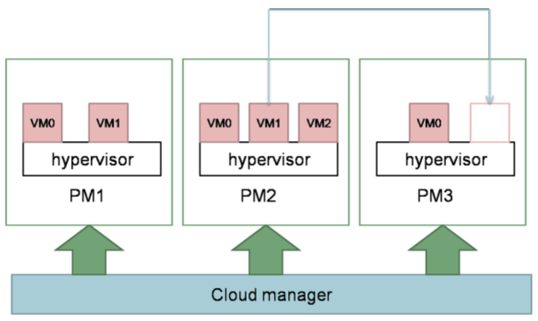

Fig. 2: cloud manager

Traditionally, VM migration technology now cannot enable VM to migrate across different hypervisors. But recently as the OVF (Open Virtualization Format) become widely used, the VMs running on the latest release of Xen and VMware can be migrated to the hypervisors of each other. We expect VMs can be seamlessly deployed in any hypervisors without any manual configurations in the future.

However, the live migration of storage still remains unsolved. The migration time of the storage is so long that it cannot be migrated lively. Therefore, in our research, we basically don't consider the possibility of hard disk migration.

2.3. Cloud Manager

The cloud manager for the federated cloud should

- monitor the current workload of the virtual servers through the interaction with hypervisors and VMs without knowing the details of services
- detect the hotspots efficiently and proactively migrate them before the SLA starts to degrade
- optimize the VM placement using heuristic algorithm which is easy to be embedded into the existing cloud manager

As shown in the Fig.2, the cloud manager is composed of monitor component and migration management component.

2.4. Problems of Resource Management

The existing cloud managers fail to dynamically allocate the virtual resources according to the load balance for the sake of SLA.

The widely used cloud manager, such as OpenNebula, has solved the problem of initial provisioning of the virtual resources, but doesn't consider the automatic dynamic allocation while the workloads and demands of VMs fluctuate in real time using migration technology. The cloud user has to detect the hotspot and migrate it to a

Table. 1: monitoring table

CPU
Memory
Network bandwidth
I/O performance

less loaded server manually. VMware DRS adopts the live VM migration technology to manage the operational cost of the cloud by migrating several VMs to a single server and shutting down the idle ones. However, it makes little effort on improving the load balance.

3. Proposed Migration Algorithm

3.1. Parameters

As shown in Table. 1, we need four system inputs to define server overload and make migrating decision. Running a VM on a physical server requires certain amount of CPU, memory and network bandwidth usage[2][8].

If the CPU and storage of a virtual server are located in different physical servers, the migration manager needs to address the multidimensionality of the network usage. Since the VM will read and write the remote disk through the network, thus consuming certain amount of network bandwidth[4][7]. We define the speed of reading and writing disk as I/O performance of VM. But as mentioned above, this parameter is related to network speed. Therefore, it cannot be considered and modeled separately.

In this research, we consider the CPU, memory, and network bandwidth usage. The problem of the I/O performance will be settled in the future work.

Given the state of each virtual server, the current load is defined as CPU usage ($CPUu$), memory usage ($MEMu$), network bandwidth usage ($NETu$) over each capacities.

$$LoadCPU = \frac{CPUu}{CPUcap} \quad (1)$$

$$LoadMEM = \frac{MEMu}{MEMcap} \quad (2)$$

$$LoadNET = \frac{NETu}{NETcap} \quad (3)$$

The values of the current load will be from 0% to 100%.

In this research, we simply sum up all the current load of VMs running on the same physical server as the current load of this physical server. But in the real condition, physical server consumes CPU and memory to run the hypervisor and manage all the VMs. The amount of the CPU and memory usage will vary according to the number of VMs running on it.

3.2. Hotspot Detection

The hotspot detection algorithm follows a threshold violation manner. This research applies a proactive migration since the threshold is defined in the level before the SLA start to degrade.

The CPU, memory, network bandwidth usage are defined separately, and hotspot will be flagged if the threshold is exceeded any of the dimensions.

$$LoadThreshold = < Tcpu, Tmem, Tnet > \quad (4)$$

3.3. Measuring Multi-dimensional Loads

In order to capture the multi-dimensional loads of the VM for the cloud manager to take migration decision using single parameter, we define $Volume$ as a metric to measure the combined CPU, Memory and Network loads of the VMs.

$$Volume1 = \frac{1}{1-cpu} * \frac{1}{1-mem} * \frac{1}{1-net} \quad (5)$$

On the other hand, we defined another metric considering the difference of the thresholds.

$$Volume2 = \frac{cpu}{Tcpu} * \frac{mem}{Tmem} * \frac{net}{Tnet} \quad (6)$$

3.4. Methods for Monitoring

Our research adopts Simple Network Management Protocol (SNMP), a feasible, UDP-based, lightweight monitoring software that is installed inside each VM and hypervisor.

Cloud manager gathers the basic information, such as disk and network status, of each VM through requiring for the XML file defining the VM. The file can be stored in hypervisor when VM was defined, or in the OVF when the VM was rebooted[3]. A software component called an agent runs on each VM and hypervisor, and reports information of CPU, memory and network bandwidth usage information via SNMP to the SNMP manager, which runs on the cloud manager.

In order to avoid the operational burden on the VMs and hypervisors and needless migration caused by the small

Table. 2: underloaded sever table

Server1
Server4
Server0
Server3

Table. 3: overloaded sever table

	Server2	
VM1	VM2	VM0
	Server5	
VM2	VM1	VM0

transient spike of the load, cloud manager requires the average value of all the parameters being monitored every two minutes through SNMP.

3.5. Selecting Destinations

Once a server is flagged as a hotspot, the migration algorithm will be triggered to search for a best migrating destination. Determining a new mapping of VMs over physical servers is NP-hard. Our destination selection deploys a greedy algorithm determine where to migrate the VMs from an overloaded physical server.

As shown in Table. 2 and Table. 3, the cloud manager keeps the table of underloaded servers and overloaded servers, it will be updated every time when the monitored parameters change during the monitoring interval, or any VM has been migrated by to a new server. CPU, memory, network bandwidth usage have them own table respectively.

In the overloaded server table, each overloaded server has a sub table of their VMs. The VM of the largest usage will be mapped to the most underloaded server.

Before initiating the real migration, the cloud manager calculates the server status after migration. If CPU, memory, network bandwidth usage can all be lower than the destination’s threshold after migration, migration process will be executed by the cloud manager.

As in the graph, If the VM1 cannot be successfully mapped to the destination, the next VM in the sub table, VM2, of server2 will be tried to migrate. If all the VM in server2 cannot be successfully mapped to server1, the cloud manager will give up dealing with server2, and turn to server5. If all the VMs on the servers in the overloaded server table cannot be successfully mapped to server1, the cloud manager will start to map them to the next server in the underloaded server table, which is server4. The loop will continue until the migration of any VM is initiated and the table by updated, or no migration can be executed among all on the table.

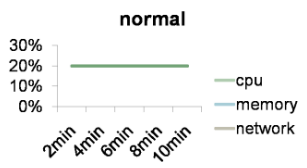

Fig. 3: normal node

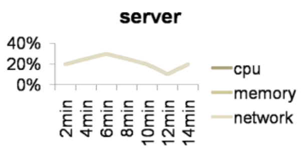

Fig. 4: server node

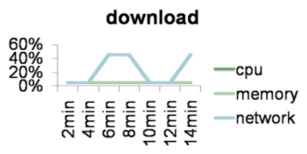

Fig. 5: download node

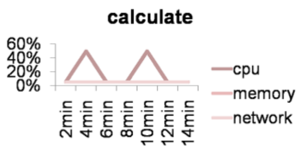

Fig. 6: calculate node

4. Evaluation

4.1. Simulation Environment

Our experiments present results from a simulation program written by C language. In the simulation, we defined a cloud consists 50 servers with the same CPU, memory, and network bandwidth capacities and the thresholds of each load vary randomly among 60%, 70% and 80%.

In order to present the VM behavior in the federated cloud, we defined 4 different types of VMs.

We managed to do 100 times experiments based on the random data. And the average values and 95% confidence interval will be shown in the results.

4.2. Types of VMs

In the federated cloud, VMs are usually set up according to a certain task. The normal node represents those VMs built up by the individual users to do the routine work such as sun the office software or visit the website. So the CPU, memory and network bandwidth usage usually keep in a certain amount. Server node serves as normal mail, ns or website server whose load fluctuate within the certain width when the number of visitor changes.

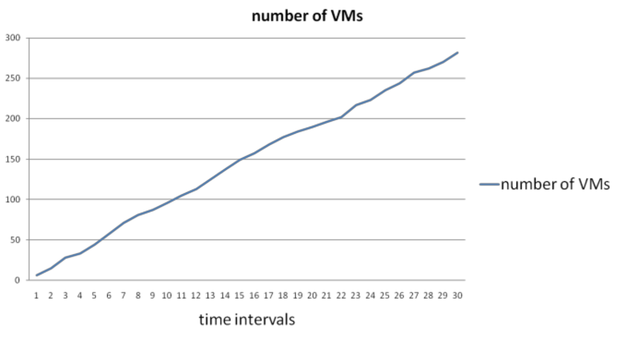

Fig. 7: # of increasing VMs

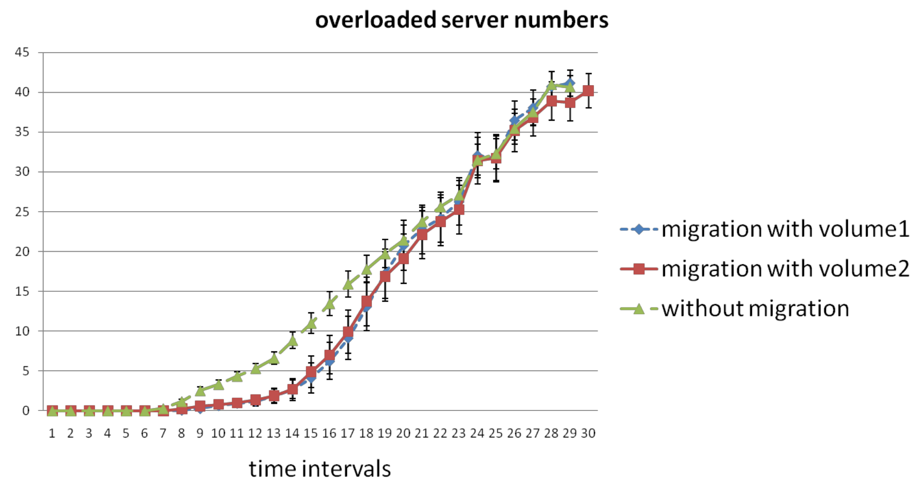

Fig. 8: of overloaded servers

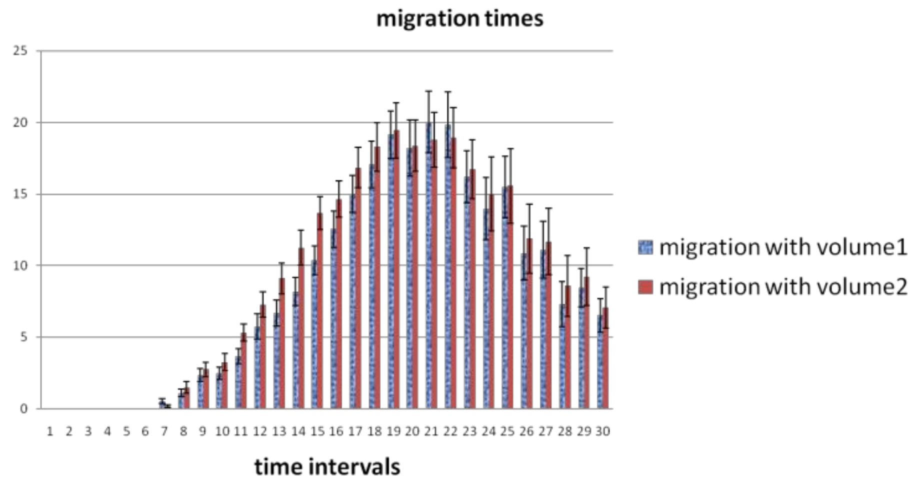

Fig. 9: Times of migration

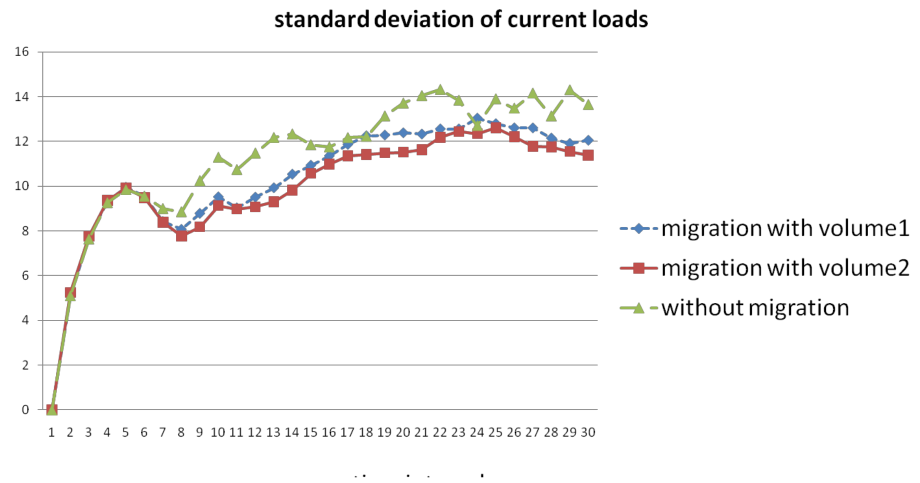

Fig. 10: Standard deviation of current loads

The network bandwidth usage of down node increases during the certain period within the certain circle when users connect to the FTP, HTTP or P2P servers and start the download process.

In the typical distributed computing system, the sub worker nodes will be assigned the task from the master nodes, accomplishing the calculation task and reporting the answers in a short time. Calculation node represents this behavior by driving the CPU using to the peak in the random short time over a certain time circle.

In the simulation, the width and peak values of the load in every VM are randomly assigned by the program.

4.3. Results

In this simulation, we set the interval of monitoring as 2 minutes.

The number of VMs increases every 2 minute randomly by from 10 to 15, and decrease randomly by from 0 to 5 until no available server can be found to host the VM. As shown in Fig. 7, the total number of VMs is increasing nearly linearly. The initial locations of VM are chosen from the most under-loaded servers according to their values of *volume* by the cloud manager.

Fig. 8 shows the numbers of over-loaded server between systems with migration function using *volume1* and *volume2*, and without migration mechanism.

Fig. 9 shows the migration time of the system with migration function using two metrics over the time.

Fig. 10 shows the standard deviation of current loads under three conditions, which stands the load balance of the whole cloud.

As shown in the graph, when the number of VMs is within the overall load of the cloud, the hotspots can be effectively eliminated by the migration process. However, if VMs grow too much, the cloud manager fails to find any available physical servers for VMs in the overloaded server to migrate to, so the times of migration in Fig. 9 starts to drop even the overloaded server number continues to rise. The performance of eliminating the hotspots using *volume1* and *volume2* are almost the same, while the load balance with *volume2* is slightly better than that with *volume1*.

5. Conclusions and Future works

In this research, we proposed a dynamic virtual resources allocation mechanism using light-weighted monitoring by SNMP and live VM migration technology. This approach automates tasks of monitoring the current load of servers and VMs, detecting the hotspots, deciding the best physical location of the busy VM, and initiating the migration.

According to the simulation results, this approach can detect and eliminate the hotspots efficiently and balance the load.

Our future work of this research is:

- Improve the VM models with real data.
- Model migration cost of the network traffic caused by the copying of the memories.
- Improve the migration mechanism considering the change of I/O performance between CPU and hard disk.

References

[1] Ali Afzal, John Darlington, and A. McGough. Qos-constrained stochastic workflow scheduling in enterprise and scientific grids. In GRID '06: Proceedings of the 7th IEEE/ACM International Conference on Grid Computing, pp. 1–8, Washington, DC, USA, 2006. IEEE Computer Society.

[2] Emmanuel Arzuaga and David R. Kaeli. Quantifying load imbalance on virtualized enterprise servers. In WOSP/SIPEW '10: Proceedings of the first joint WOSP/SIPEW international conference on Performance engineering, pp. 235–242, New York, NY, USA, 2010. ACM.

[3] Ferm´ın Gal´an, Americo Sampaio, Luis Rodero Merino, Irit Loy, Victor Gil, and Luis M. Vaquero. Service specification in cloud environments based on extensions to open standards. In COMSWARE'09: Proceedings of the Fourth International ICST Conference on COMmunication System softWAre and middlewaRE, pp. 1–12, New York, NY, USA, 2009. ACM.

[4] Takahiro Hirofuchi, Hidemoto Nakada, Hirotaka Ogawa, Satoshi Itoh, and Satoshi Sekiguchi. A live storage migration mechanism over wan and its performance evaluation. In VTDC '09: Proceedings of the 3rd international workshop on Virtualization technologies in distributed computing, pp. 67–74, New York, NY, USA, 2009. ACM.

[5] Sanjay Kumar, Vanish Talwar, Vibhore Kumar, Parthasarathy Ranganathan, and Karsten Schwan. vmanage: loosely coupled platform and virtualization management in data centers. In ICAC '09: Proceedings of the 6th international conference on Autonomic computing, pp. 127136, New York, NY, USA, 2009. ACM.

[6] Liang Liu, HaoWang, Xue Liu, Xing Jin, Wen BoHe, Qing Bo Wang, and Ying Chen. Greencloud: a new architecture for green data center. In ICAC-INDST '09: Proceedings of the 6th international conference industry session on Autonomic computing and communications industry session, pp. 29–38, New York, NY, USA, 2009. ACM.

[7] Beomjoo Seo and Roger Zimmermann. Efficient disk replacement and data migration algorithms for large disk subsystems. Trans. Storage, Vol. 1, No. 3, pp. 316–345, 2005.

[8] Aameek Singh, Madhukar Korupolu, and Dushmanta Mohapatra. Server-storage virtualization: integration and load balancing in data centers. In SC '08: Proceedings of the 2008 ACM/IEEE conference on Supercomputing, pp. 1–12, Piscataway, NJ, USA, 2008. IEEE Press.

[9] Alexander Stage and Thomas Setzer. Networkaware migration control and scheduling of differentiated virtual machine workloads. In CLOUD'09: Proceedings of the 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing, pp. 9–14, Washington, DC, USA, 2009.IEEE Computer Society.

[10] Akshat Verma, Puneet Ahuja, and Anindya Neogi. pmapper: power and migration cost aware application placement in virtualized systems. In Middleware '08: Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware, pp. 243–264, New York, NY, USA, 2008. Springer-Verlag New York, Inc.

[11] OpenNebula
http://www.opennebula.org/