International Journal of Grid Distribution Computing
Vol.8, No.2 (2015), pp.177-190
http://dx.doi.org/10.14257/ijgdc.2015.8.2.17

Utility-based Virtual Cloud Resource Allocation Model and Algorithm in Cloud Computing

Zhu Jianrong, Li Jing and Zhuang Yi

School of Computer Science and Technology
Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China
zjr_nuaa@163.com, zy16@nuaa.edu.cn

Abstract

How to satisfy the users' QoS requirements while improving the resource utilization is one of the key technologies in cloud computing environment. In our work, a virtual cloud resource allocation model VCRAM-U (Utility-based Virtual Cloud Resource Allocation Model) is proposed. In our model, the problem of virtual cloud resources allocation is abstracted as a utility-maximization problem, taking tradeoffs between the utility of the data center and the performance of the applications into account, and maximizing the utility on the premise of meet user's performance. We design a local decision algorithm and a global decision algorithm to solve our model. Experimental results show that the virtual cloud resources can be managed and allocated efficiently by our model and algorithms. In addition, our model can get a higher utility of the data center compared with other models.

Keywords: *cloud computing, utility, virtualization, resource allocation, QoS*

1. Introduction

Cloud computing is a new computing model after parallel computing, grid computing and distributed computing [1]. The cloud environment contains a large number of decentralized, heterogeneous resources such as memory, CPU, network bandwidth and disk. The problem to be solved for cloud computing is how to manage and share the cloud resources effectively and securely, and how to provide the appropriate services. It emphasizes the comprehensive resource sharing and comprehensive application service [2]. Cloud computing has presented the highest goal of resource sharing within the full scope of the Internet, to make full use of the computing and storage resources. It is an effective platform for integration of high-performance computing resources in the whole society.

Virtualization is an important characteristic of cloud computing which has made the task no longer assigned to a physical device to perform like the grid, but made the user's requests to be mapped to a virtual host, while the virtual host can be mapped to different physical devices. Virtualization technology provides two key mechanisms to improve the resource utilization: (1) resource dynamic allocation mechanism, (2) virtual machine migration mechanism. In particular, the dynamic migration technique of the virtual machines (VMs) provides the possibility of running tasks without interruption in the process of virtual machine migration. However, it also increases the complexity of the virtual resource management. Therefore, to find a suitable method to allocate the virtual resources to the virtual machines dynamically while taking users' Quality of Service (QoS) requirements and the costs of the data center into account is a big challenge. Currently, most of researches about the resource allocation technology have focused on the performance of application and resource usage cost. However, most studies did not fully consider how to decrease the cost while improving the application's performance. An effective method to solve the tradeoff between the cost and performance is to use the

ISSN: 2005-4262 IJGDC

utility function. In economics, the utility function is usually used to represent the quantity relation between the obtained utility and a set of goods, to measure the consumers' satisfaction.

In our work, a virtual cloud resource allocation model VCRAM-U (Utility-based Virtual Cloud Resource Allocation Model) is proposed. In our model, the problem of virtual cloud resources allocation is abstracted as a utility-maximization problem, taking tradeoffs between the utility of the data center and the performance of the applications into account, and maximizing the utility on the premise of meet user's performance. We also design a decision algorithm and a global decision algorithm to solve our model. Experimental results show that the virtual cloud resources can be managed and allocated efficiently by our model and algorithms. In addition, our model can get a higher utility of the data center compared with other models.

The rest of this study is organized as follows. In Section 2 we discuss the related works. Section 3 describes our utility-based virtual resource allocation model and related algorithms. The simulation experiments and results are set out and analyzed in Section 4. Finally, the conclusions and future work are presented in Section 5.

2. Related Work

In recent years, the cloud computing resources allocation model and algorithm have been broadly studied. Typical resources allocation strategies include strategies based on the rent theory and dynamic multi-stage resource pool, economics-based resource allocation strategies, strategies based on general optimization algorithm and the optimal resource allocation strategy based on random integer programming, *etc*.

[4] Proposed a virtual resources management mechanism that achieves the allocation of resources through the division and resources reserve strategy, while ensuring the effectiveness for the users to use the virtual resources. The authors also proposed a borrowed/lend scheduling strategy to maximize the resource utilization [5]-[6] proposed the economics of cloud computing architecture from the perspective of economics and designed an economic model of cloud resource management on the basis of QoS. Based on the concept of economics, cloud computing environment can be seen as a cloud market, where resources are seen as goods, compute clouds and storage clouds are abstracted as resource providers, and the cloud users are seen as consumers. The strategy provides feedback about the economic incentives to improve the resource utilization, contributing to the efficient management and optimal configuration of resources in the cloud computing environment, while maximizing the users' request for QoS. But this strategy was proposed from an economic point of view, without the consideration of either saving money for consumers or dynamics of the resources and price.

The random integer programming based virtual resource optimization scheduling method is also a hot topic [7]-[10]. In cloud computing, cloud providers can provide resources to users in two ways: booking and pay as you go [11]. Booking methods can effectively reduce the cost of the user. However, it is difficult to meet the needs of users through the way of booking completely, because of the uncertainty of the users' demand and the resources' price. Heuristic method [12] or k-nearby algorithm [13] can forecast the resources needed by the user. This strategy provides a dynamic solution for providing resources to meet the customers' demand, considering the resource cost at every stage, and minimizing the users' cost. Although this model could reduce the resource cost for the cloud consumer, it didn't consider the utility of the data center.

[14] Utilized constraint satisfaction problem to solve the problem of virtual resource allocation, in which the constraint denotes the users' service level agreement (SLA) aiming to maximize the energy savings. Its core idea is to maximize the free physical machine number and save energy by closing free physical machine.[15][16] used a virtual machine deployment solution in which the constraint also denotes the users' SLA. Moreover,

some additional application constraints can be easily introduced into the constraint by using the constraint satisfaction strategy.

In summary, although researchers have made certain achievements in the field of virtual resources allocation, the existing works still have some shortcomings. For example, the existing works do not give a full consideration to the diversity of cloud resources, the utilization of resources is low and the utility is not maximized, *etc*. According to the above problems, a virtual cloud resource allocation model, VCRAM-U (Utility-based Virtual Cloud Resource Allocation Model), is proposed. In our work, the virtual cloud resources allocation problem is modeled as a utility-maximization problem. Compared with the model ARAP-U, our model can get higher utility of the data center whiling guaranteeing user's quality of service.

3. Utility-based Virtual Resource Allocation Model and Algorithm

In this section, we introduce the framework of our model, including the utility function, local node controller and global controller, as well as local decision algorithm and global decision algorithm for resource allocation.

3.1. Model Framework

References [3] proposed a virtual resource allocation model based on utility, but the paper only considered one dimension as the CPU. In our work we have carried on the expansion, modeling the problem of virtual cloud resource allocation mainly from the CPU, memory, and network bandwidth three dimensional.

Assume that all physical machines (PM) are homogeneous, and can migrate VMs from one PM to another. The framework of VCRAM-U model is composed of the local node controller and the global controller as shown in Figure 1. The local node controller maximizes the local utility by allocating of CPU, memory, and network bandwidth dynamically. And the global controller maximizes the global utility through virtual machine migrations. In VCRAM-U, we assume that a running virtual machine is associated with only one application.

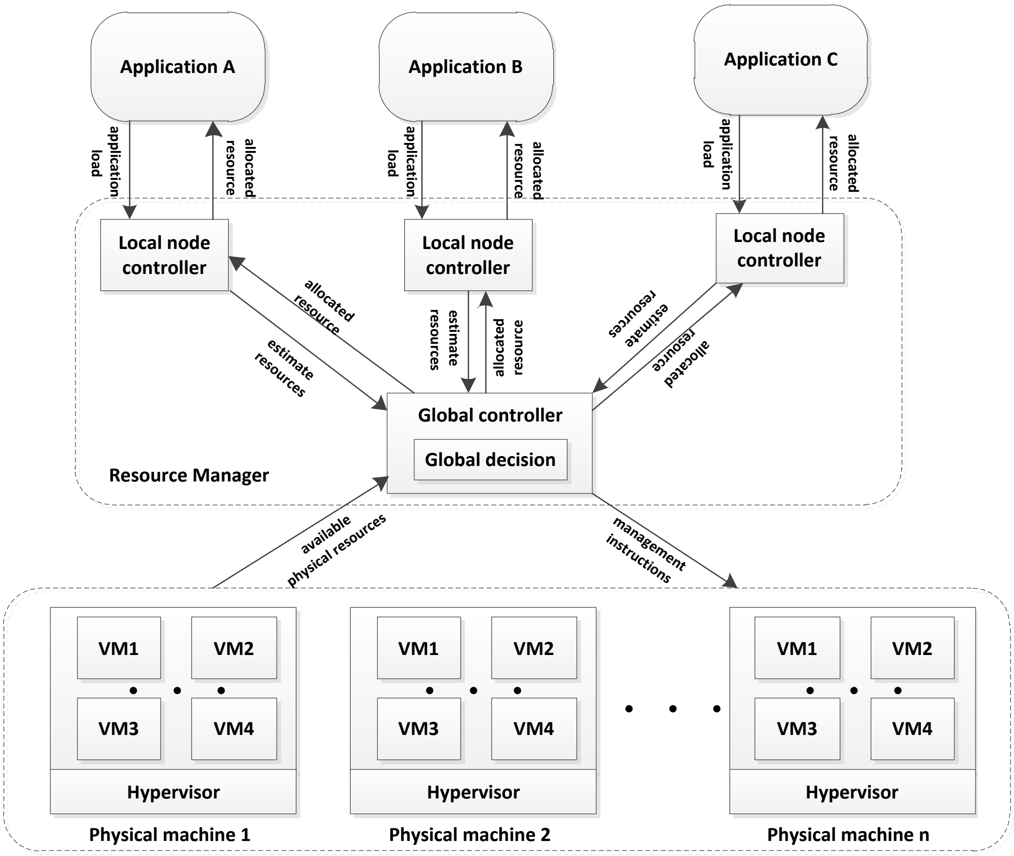

Figure 1. VCRAM-U Mode

3.2. Utility Function Model

Cloud users deploy their applications on a virtual machine service provided by cloud service providers to run, and paid according to the virtual resources they got. We defined a utility function of the virtual machine which was used to represent the costs of using the virtual machine by the user. We also defined a utility function of the local node to express the profit of generated through a single node. In addition to these, the global utility function was defined to express the profit generated by the entire system.

Definition 1. The utility function of the virtual machine is a linear function as shown in Formula (1).

$$U_{vm}^{i} = \alpha_i * C_i + \beta_i * M_i + \gamma_i * B_i \qquad (1)$$

In this paper CPU, memory and network bandwidth are all taken into account. In Formula (1), α_i , β_i and γ_i is the account of money paid unit of CPU, memory and network bandwidth resource allocated (*e.g.*, α_i = 8￥/1%/memory), C_i , M_i and B_i is the CPU, memory and network bandwidth allocated to VM_i .

Definition 2. The utility function of the local node is difference between the sum of the utility of all VMs running on the node and the node costs are as shown in Formula (2).

$$U_{node}^{j} = \sum_{i=1}^{m} U_{vm}^{ji} - C_{cost}^{j} \qquad (2)$$

In Formula (2), m is the number of virtual machines running on the node, U_{vm}^{ji} is the utility of VM_i on node j , and C_{cost}^{j} is the cost of node j , where $1 \le i \le m$.

Definition 3. The global utility function is the sum of the utility of all local nodes as shown in Formula (3).

$$U_{global} = \sum_{j=1}^{n} U_{node}^{j} \quad (3)$$

In (3), n is the number of local nodes, and U_{node}^{j} is the utility of node j, where $1 \le j \le n$ .We can get (4) by replacing Formula (3) with Formula (1) and Formula (2).

$$U_{global} = \sum_{j=1}^{n} U_{node}^{j} = \left(U_{vm}^{11} + U_{vm}^{12} + \cdots + U_{vm}^{21} + U_{vm}^{22} + \cdots\right) - \left(C_{cost}^{1} + C_{cost}^{2} + \cdots\right) = U_{vm} - C_{cost} \quad (4)$$

From Formula (4), it is observed that the global utility is the difference between the utility of all VMs and the costs of entire system.

Definition 4. Let S_a and D_a be the utility of the source node and target node before VM migration, S_b and D_b be the utility of the source node and target node after VM migration. $[NodeCost]$ is added if there is a new node turned on during the migration. Then Formula (5) gives the increased utility generated by a VM migration, where I is the increased utility generated by a VM migration.

$$I = (D_a - D_b) - (S_b - S_a) - [NodeCost] \quad (5)$$

The increased global utility obtained by all VM migrations can be calculated by Formula (6).

$$I_{gobal} = I_1 + I_2 + \cdots I_i + \cdots + I_m + \cdots + NodeCost \quad (6)$$

In (6), I_i is the increased utility for VM migration VM_i, and $NodeCost$ is the reduced cost of the local node turned off through VM migration.

3.3. Local Node Controller and Local Decision Algorithm

As shown in Figure 2, the local node controller is used to dynamically allocate memory, CPU, and network bandwidth resources to different virtual machines to maximize the local node utility. The local node controller is composed of an arbitrator, a virtual machine monitor and a feedback controller. The virtual machine monitor measures the average CPU utilization, memory utilization and network bandwidth of virtual machines in every control interval. Users can set this control according to how long the local node controller reflects the load information. The feedback controller determines the memory, CPU, and network bandwidth resources to be given to each VM to make the memory, CPU and network bandwidth utilization meet the QoS objectives. Then the arbitrator takes memory, CPU and network bandwidth requirements for each virtual machine form feedback controller, and allocates the resources according to the application of every virtual machine using the local decision algorithm descried below to achieve the goal of maximizing the local node utility.

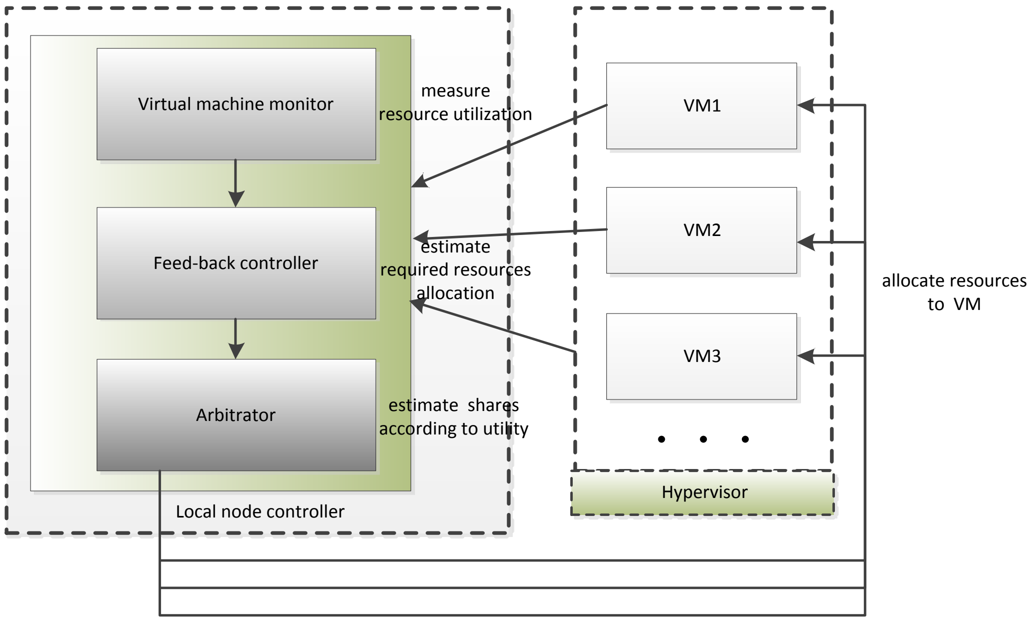

Figure 2. Local Node Controller

3.3.1. Local Decision Algorithm

Local node controller runs on the physical machine which running the virtual machine, estimates the resources required for virtual machines and allocates resources to VMs by local decision algorithm to achieve the goal of maximizing the utility of local node. The decision algorithm maximizes the utility of local node by first meeting the requests of those VMs that provide higher utility (the utility of per unit of CPU resource is higher), and then satisfying the requests of other VMs in decreasing order of VM utility. But, if there are not enough resources to meet all requests, then those VMs that provide lower utility cannot get resources for a long time. To solve the problem, we usually set a limit value of resources they can get for every VM to minimally meet the requirement, in our case 5% of the CPU. Then the remaining resources are allocated to VMs by decision algorithm in decreasing order of utility.

Local decision algorithm is described as follows:

Step 1 Firstly, estimate the resources required for every VM by the feedback controller;

Step 2 Allocate corresponding resources to VMs according to the lower limit set for the VMs;

Step 3 According to the resources required for VMs from Step 1 and the lower limit from Step 2, calculate the remaining resources required for every VM;

Step 4 Arbitrator sets resource shares allocated to VMs equal to their remaining required resource allocation;

Step 5 Check whether all the resources required for all the virtual machines exceed the total resource capacity, then go to Step6 and if not go to Step 7;

Step 6 Arbitrator first meets the requests of those VMs that provide higher utility, sets resources allocated to them equal to their required resource allocation and then satisfies the requests of other VMs in decreasing order of utility;

Step 7 If all the resources required for all the virtual machines do not exceed the total resource capacity, the arbitrator sets resource shares allocated to all VMs equal to their required resource allocation.

In spite of this, there are some virtual machines don’t get enough resources sometimes. So we will introduce the global decision algorithm in the following section. Through the global decision algorithm, the global system utility can be increased by VM migrations which were determined by the global decision algorithm.

The pseudo code of local decision algorithm is shown in Figure 3:

```
Local decision algorithm
Input: total resources provided by physical nodes(ResourceTotal),limit value of resources for VM(Resourcelimit)
Output: resources allocated to VMs(VMRes)
Steps:
1)For the PhysicalNode:                 //physical machines
2)    Begin
3          ResVM← GetResources(VM)//get required resources of VMs
4)         Allocation(Resourcelimit)//allocated limit resources to VMs
5)         ResVM←Compute(ResVM,Resourcelimit)//calculate the remaining resources required for every VM
6)         if isExceed(ResVM,ResourceTotal) then
7)              AllocationVM(ResVM) //allocate resources in decreasing order of utility
8)         else AllocationVM2(ResVM)//sets resource shares allocated to all VMs equal to their required resource allocation.
9)         end
10)   end
11)return VMRes;
```

Figure 3. Pseudo Code of Local Decision Algorithm

3.4. Global Controller and Global Decision Algorithm

The global controller runs on a single node which does not host any virtual machine. It gets the resources requirements of all VMs and resources shares VMs have got by querying the local controllers in every control interval. Then the global controller determines the list of VM migrations by global decision algorithm, and maximizes the global utility by VM migrations. There are two kinds of VM migration which can improve the global system utility. One is to migrate a virtual machine from a node which was overloaded and where its resources are not enough to satisfy all VMs to an under-loaded node or a node which was newly turned on, and the other is to migrate all virtual machines which run on an under-loaded node to other nodes, and then turn off the node to reduce the cost. For finding mapping relationship between the virtual machines and the physical machines and maximizing the global system utility, we design a global decision algorithm. Its main purpose is maximizing the global utility by the list of VM migrations which was determined through the global algorithm.

3.4.1. Global Decision Algorithm

The global decision algorithm gets the resource allocation of the VMs and the nodes mainly by querying the local controllers, and then migrates the VMs which offer lower utility running on an overloaded node or all the VMs running on an under-loaded node to some other nodes. By a similar series of VM migrations, the global system utility can be maximized. First, we divided the nodes into three groups based on the required allocations of all virtual machines and CPU shares: overloaded nodes, under-loaded nodes, and normal nodes. A node is considered as an overloaded node if the CPU resources required for all VMs running on the node exceed the total CPU capacity, and the number of VMs running on the node is more than one. If the CPU resources required for all VMs running on a node don't exceed 50% of the total CPU capacity, then we consider the node as an under-loaded node. If the CPU resources required for all VMs running on a node exceed 50% of the total CPU capacity, but don't exceed all the CPU resources of the node, we consider the node as a normal node. If a node does not meet above three kinds of circumstances, this is to say, the node is unstable for a long time, and then we consider the node as an unstable node. In order to guarantee the stability of the algorithm, we don't consider the unstable nodes during the VM migrations.

Global decision algorithm is described as follows:

Step 1 Get the resource allocation of VMs and the node by querying the local controllers;

Step 2 Check whether there is any overloaded node exists in the system. If there are overloaded nodes, then go to Step3; otherwise, go to Step9;

Step 3 Calculate the utility of all VMs on the node with Formula (1), and choose a VM which offers the lowest utility;

Step 4 Choose a normal node as the target node of the VM migration;

Step 5 Calculate the utility of the target node D_b and the source node utility S_b which runs the VM before VM migration with Formula (2);

Step 6 Simulate this VM migration, calculate the utility of all VMs on the source node and the utility of all VMs on the target node with Formula (1), and calculate the utility of the target node D_a and the source node utility S_a which runs the VM after VM migration with Formula (2);

Step 7 Calculate the utility increased by this VM migration with Formula (5);

Step 8 Repeat Step4~Step7, choose a node which provides the highest increase in global utility as the target node for this VM migration, and put this VM migration into the list of VM migrations. If all of the target nodes cannot increase the global utility, then this VM migration will not be put into the list of VM migrations;

Step 9 There is no overloaded node, check if there are any under-loaded nodes in the system. If there are under-loaded nodes, then go to Step 10 and migrate all VMs running on the node; otherwise, go to Step11;

Step 10 Repeat Step 4-8, and determine the list of VM migrations for this under-loaded node;

Step 11 Simulate all the VM migrations in the list of VM migrations, and calculate the global system utility improved by the list of VM migrations with Formula (6). If the global utility is improved, then the list of VM migrations will be done; otherwise, these VM migrations will be canceled.

Step 12 Repeat the whole algorithm until there are no overloaded nodes and under-loaded nodes in the system.

The pseudo code of global decision algorithm is shown in Figure 4:

Global decision algorithm

```
Input: resources information of nodes(NodeInfo)
Output: the list of VM migrations(migList)
Steps:
1)For the PhysicalNode:                //physical machines
2)    Begin
3)      if isExistOverloaded(NodeInfo) then//If there are any overloaded nodes
4)          UilityVM←Calculate(NodeID)//calculate the utility of VMs
5)          VMId←ChooseVM(UilityVM)//choose a VM which offers the lowest utility
6)          foreach node in nodeList do
7)              NodeTarget←ChooseNode(NodeInfo)//choose destination node
8)              Uillist←CalUility(NodeS,NodeD)//Calculate the utility increased by this VM migration
9)          end
10)         mig←ChooseNode(Uillist)
11)         migList←AddMigList(mig)//put this VM migration into the list of VM migrations
12)     else if isExistUnderloaded(NodeInfo) then//if  there are any underloaded nodes
13)         foreach node in nodeList do
14)             NodeTarget←ChooseNode(NodeInfo)///choose destination node
15)             Uillist←CalUility(NodeS,NodeD)
16)         end
17)         mig←ChooseNode(Uillist)
18)         migList←AddMigList(mig)//put this VM migration into the list of VM migrations
19)         UilGlobal←Cal(migList)
20)         if isIncrease(UilGlobal)then
21)             doMigration()
22)         else Empty(migList)//empty the list of VM migrations
23)     end
24)return migList;
```

Figure 4. Pseudo Code of Global Decision Algorithm

4. Experiment and Analysis

In order to verify our model and algorithm, we conduct three experiments. The purpose of experiment 1 is to verify the performance in two different cases: (1) when the resource manager is enabled (2) when the resource manager is disabled; In order to show that our resource manager can provide the tradeoffs between the performance of applications and the utility of data center, we conduct experiment 2; Experiment 3 shows that our proposed model VCRAM-U can get higher system utility compared with other similar model.

Experiment 1 is the foundation of the other two experiments, and the data center is composed of 5 physical machines. Table 1 gives the configuration of each PM, and Table 2 shows the configuration of each virtual machine in the experiments. Physical machine PM1, PM2, PM3 hosts the virtual machine VM1, VM2, VM3, respectively. VM1 and VM2 run large-scale computing applications, and VM3 runs a web application. The global controller runs on PM4 and PM5 runs a generator which is responsible for generating the web request running on VM3. Table 3 shows the utility parameter of resources in experiment 1, and Table 4 gives the utility parameter of resources in experiment 2.

Table 1. PM Parameter in Experiment

PM	RAM	CPU	Network Bandwidth
PM1	4096MB	4096 MIPS	1000 Mbps
PM2	4096MB	4096 MIPS	1000 Mbps
PM3	4096MB	4096 MIPS	1000 Mbps
PM4	4096MB	4096 MIPS	1000 Mbps
PM5	4096MB	4096 MIPS	1000 Mbps

Table 2. VM Parameter in Experiment

VM	RAM	CPU	Network Bandwidth
VM1	512MB	512MIPS	100Mbps
VM2	1024MB	1024MIPS	100Mbps
VM3	2048MB	2048MIPS	100Mbps

Table 3. Utility Parameter Setting in Experiment 1

VM types	α_i	β_i	γ_i	C_{cost}
VM1	4	4	4	60
VM2	4	4	4	60
VM3	8	8	8	60

Table 4. Utility Parameter Setting in Experiment 2

VM types	α_i	β_i	γ_i	C_{cost}
VM1	2	2	2	60
VM2	4	4	4	60
VM3	8	8	8	60

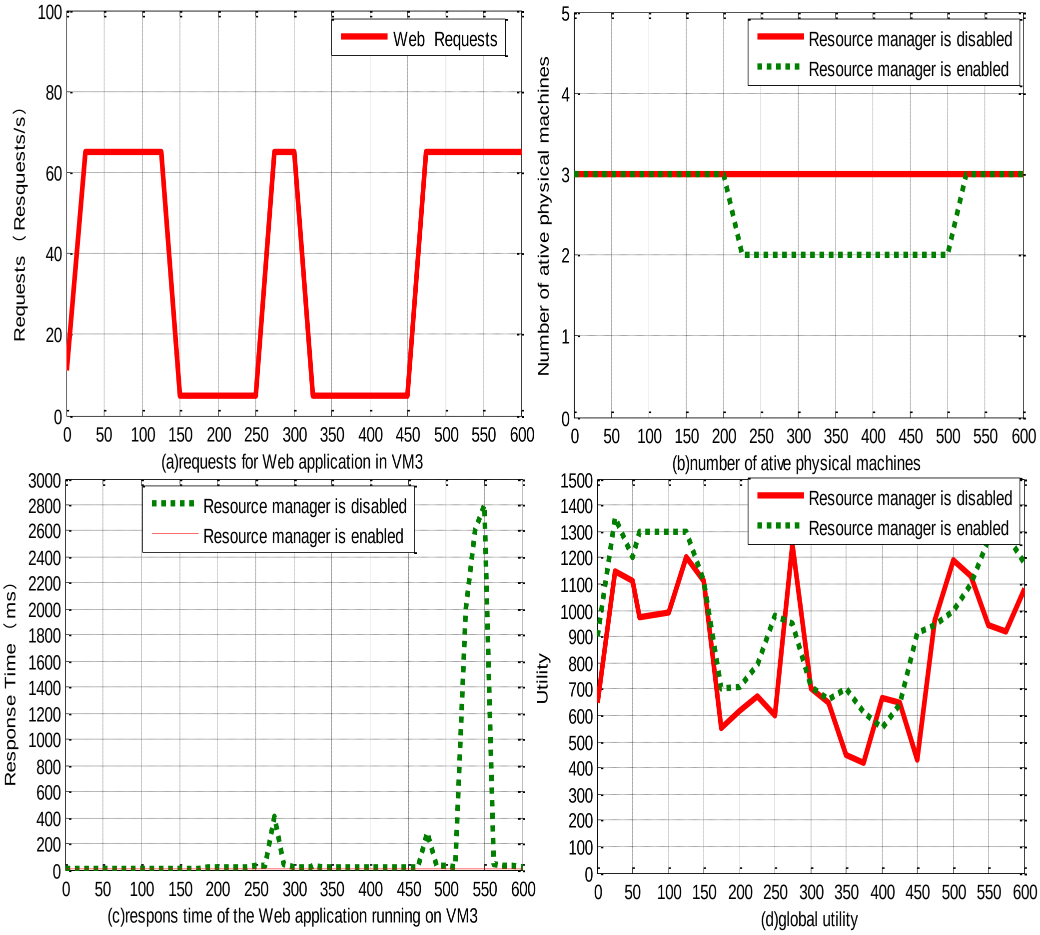

Figure 5. Simulation Experiment Results

In experiment 1, we conduct simulation experiments in two cases when the resource manager is disabled and enabled. The experiment 1 shows that each virtual machine runs on its own PM in the whole experiment when the resource manager is disabled. Figure 5 is the results of simulation experiment 1, and the horizontal coordinate is the time seconds. Figure 5(a) shows the request of the Web application in a period of time. Figure 5(b) shows the number of active physical machines in data center. From Figure 5(a) we can see that between second 150 and 450, as request of the Web application is low, the resources allocated is less, so there are only two physical machines launched(here, we only consider the physical machine which hosts VMs). As Figure 5(c) shows, between second 250 and 300 and second 500 and 550, the response time of the Web application is significantly increased when the resource manager is enabled. This is because that there is a VM being migrated from a physical machine to another physical machine during this period of time. Figure 5(d) gives the global system utility for both cases, and we can see that when the resource manager is enabled we can achieve a higher global utility.

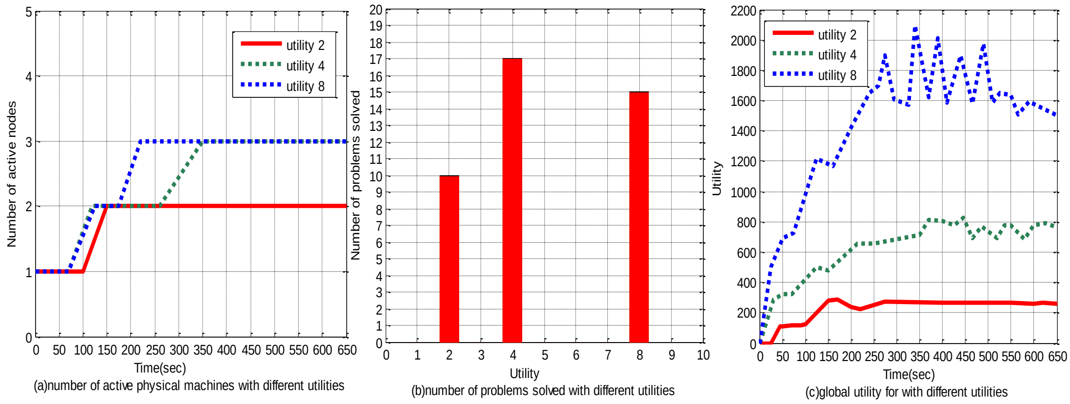

Figure 6. Simulation Experiment Results

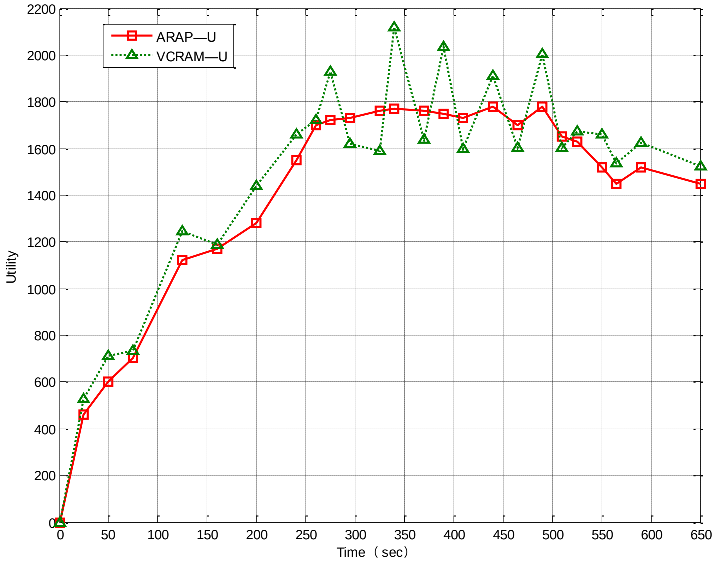

Figure 7. Utility Comparison of Different Models

In experiment 2, we start the experiment by running three VMs on one physical machine, and every VM runs a computing application as shown in Figure 6(a). Figure 6(a) shows the number of active physical machines in the whole process of experiment. From Figure 6(a), we can see that when the utility of VM is 2, the number of active physical machines is 2. This is due to turning on a new physical machine will reduce the global utility when the utility of VM is low. As shown in Figure 6(b), the problem instances solved is less in the case of utility 2 due to using less physical machines to solve problem. Figure 6(c) shows the global utility in three cases, and from Figure 6(c) we can see that the resource manager can improve global utility through VMs migrations.

In experiment 3, we compare our proposed model VCRAM-U with the utility-based adaptive resource allocation policy ARAP-U [17] as shown in Figure 7. From the experiment results we can see that our model VCRAM-U can achieve higher utility.

5. Conclusion

In our work, a virtual cloud resource allocation model VCRAM-U (Utility-based Virtual Cloud Resource Allocation Model) is proposed. In our model, the problem of virtual cloud resources allocation is abstracted as a utility-maximization problem, taking tradeoffs between the utility of the data center and the performance of the applications into account, and maximizing the utility on the premise of meet user's performance. A local decision algorithm and a global decision algorithm are also designed to solve our proposed model. Experimental results show that the virtual cloud resources can be managed and allocated efficiently by our model and algorithm. In addition, our model can get a higher utility of the data center compared with other models. But, when the size of the cloud computing environment becomes larger and larger, there will be some problems with our model and algorithm such as performance bottlenecks. Therefore, in the future work, we will be interested in studying a distributed virtual cloud resource manager for large-scale cloud computing environment.

References

[1] R. Buyya, C. S. Yeo and S. Venugopal, *et al*., "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility", Future Generation computer systems, vol. 25, no. 6, **(2009)**, pp. 599-616.

[2] Ke S H I X L X U, "Utility Maximization Model of Virtual Machine Scheduling in Cloud Environment", Chinese Journal of Computers, vol. 2, no. 004, **(2013)**.

[3] D. Minarolli and B. Freisleben, "Utility-based resource allocation for virtual machines in Cloud computing", Computers and Communications (ISCC), **(2011)** IEEE Symposium on. IEEE, **(2011)**, pp. 410-417.

[4] W.C. Yuan, Y. A. ZHU and W. Lu, "Exploring Virtualized Resource Management Mechanism for Cloud Computing", Journal of North western Polytechnic University, vol. 28, no. 5, **(2010)**, pp. 704-708.

[5] H. Q. Gao and Y. Xing, "Research on Cloud Resource Management Model Based on Economics", Computer Engineering and Design, vol. 31, no. 19, **(2010)**, pp. 4139-4142.

[6] H. N. Van, F. D. Tran and J. M. Menaud, "SLA-aware virtual resource management for cloud infrastructures", //Computer and Information Technology, 2009. CIT'09. Ninth IEEE International Conference on. IEEE, vol. 1, **(2009)**, pp. 357-362.

[7] S. Chaisiri, B. S. Lee and D. Niyato, "Optimal virtual machine placement across multiple cloud providers", Services Computing Conference, 2009. APSCC 2009. IEEE Asia-Pacific. IEEE, **(2009)**, pp. 103-110.

[8] R. N. Calheiros, R. Ranjan and A. Beloglazov, *et al*., "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms", Software: Practice and Experience, vol. 41, no. 1, **(2011)**, pp. 23-50.

[9] S. Chaisiri, B. S. Lee and D. Niyato, "Optimization of resource provisioning cost in cloud computing", Services Computing, IEEE Transactions on, vol. 5, no. 2, **(2012)**, pp. 164-177.

[10] J Wang, X. Yang and T. Ma, *et al*., "An energy-efficient competitive clustering algorithm for wireless sensor networks using mobile sink", International Journal of Grid and Distributed Computing, vol. 5, no. 4, **(2012)**, pp. 79-92.

[11] N. Bobroff, A. Kochut and K. Beaty, "Dynamic placement of virtual machines for managing sla violations", Integrated Network Management,. IM'07. 10th IFIP/IEEE International Symposium on IEEE, **(2007)**, pp. 119-128.

[12] K. Miyashita, K. Masuda and F. Higashitani, "Coordinated Service Allocation through Flexible Reservation", Services Computing, IEEE Transactions on, vol. 1, no. 2, **(2008)**, pp. 117-128.

[13] J. Chen, G. Soundararajan and C. Amza, "Autonomic provisioning of backend databases in dynamic content web servers", Autonomic Computing, 2006 ICAC'06, IEEE International on. IEEE, **(2006)**, pp. 231-242.

[14] H. N. Van, F. D. Tran and J. M. Menaud, "Performance and power management for cloud infrastructures", //Cloud Computing (CLOUD), **(2010)** IEEE 3rd International Conference on IEEE, **(2010)**, pp. 329-336.

[15] C. Hyser, B. Mckee and R. Gardner, *et al*., "Autonomic virtual machine placement in the data center", Hewlett Packard Laboratories, Tech. Rep. HPL-2007-189, **(2007)**, pp. 2007-189.

[16] N. Bobroff, A. Kochut and K. Beaty, "Dynamic placement of virtual machines for managing sla violations", //Integrated Network Management, IM'07 10th IFIP/IEEE International Symposium on IEEE, **(2007)**, pp. 119-128.

[17] J. Wan, P. Shan and X. Xu, "A Utility-Based Adaptive Resource Allocation Policy in Virtualized Environment", //Dependable, Autonomic and Secure Computing, DASC'09, Eighth IEEE International Conference on IEEE, **(2009)**, pp. 63-68.

Authors

Zhu Jianrong, received the B.S. degree from Nanjing University of Aeronautics & Astronautics in 2012. Now he is studying for the master degree at Nanjing University of Aeronautics & Astronautics. His research interest is Grid and Distributed Computing.

Li Jing, was born in 1976. She received the Ph.D. degree in Computer Science and Technology from Nanjing University in 2004. Currently, she is an associate professor of Nanjing University of Aeronautics & Astronautics. Her research interests include Distributed Computing and data mining.

Zhuang Yi, was born in 1957. She is a professor and Ph. D. supervisor of Nanjing University of Aeronautics & Astronautics. Her research interests include Distributed Computing and Information Security.