A Task Scheduling Algorithm Based on Load Balancing in Cloud Computing

Yiqiu Fang[1], Fei Wang[1], and Junwei Ge[2]

[1] College of Computer Science and Technology, Chongqing University of Posts and Telecommunications, 400065 Chongqing, China
[2] College of Software, Chongqing University of Posts and Telecommunications, 400065, Chongqing, China
fangyq@cqupt.edu.cn, wangfei___zxc@163.com, gejw@cqupt.edu.cn

Abstract. Efficient task scheduling mechanism can meet users' requirements, and improve the resource utilization, thereby enhancing the overall performance of the cloud computing environment. But the task scheduling in grid computing is often about the static task requirements, and the resources utilization rate is also low. According to the new features of cloud computing, such as flexibility, virtualization and etc, this paper discusses a two levels task scheduling mechanism based on load balancing in cloud computing. This task scheduling mechanism can not only meet user's requirements, but also get high resource utilization, which was proved by the simulation results in the CloudSim toolkit.

Keywords: cloud computing; task scheduling; virtualization; load balancing.

1 Introduction

As the key and frontier field of the current domestic and international computer technology, cloud computing is a computing paradigm that can provide dynamic and scalable virtual resources through the Internet service to users on demand, and also it is further development of distributed computing, parallel computing and grid computing [1]. Its main advantage is that it can quickly reduce the hardware costs and increase computational power and storage capacity; users can access high quality of service by low cost. And it is unnecessary to purchase expensive hardware equipment, as well as upgrade and maintain frequently.

Task scheduling is an important part of cloud computing, which is a mechanism that maps users' tasks to appropriate resources to execute, its efficiency will directly affect the performance of the whole cloud computing environment. Compared with grid computing, there are many unique properties and the mainly include virtualization and flexibility for cloud computing [2]. By using virtualization technology, all physical resources are virtualized and are transparent to user. All user has their own virtual machine and don't affect each other, which is created according to the user's requirement. Furthermore, one or multiple virtual machines can be run on a single host, and the utilization of resources is improved effectively, and the running independency of users' application is ensured as well as the information security of system and service availability is improved. Flexibility is the resource

F.L. Wang et al. (Eds.): WISM 2010, LNCS 6318, pp. 271–277, 2010.

provided by cloud computing environment which can be increased or reduced dynamically according to users' demand for tasks. Owing to these new features, the task scheduling mechanism for grid computing can not work effectively in the cloud computing environment.

In this paper, a task scheduling mechanism based on the two levels of load balance is discussed, which consider the flexibility and virtualization in cloud computing to meet the dynamic task requirements of users and improve the utilization of resources.

2 Scheduling Model

Cloud Computing Architecture includes three layers, application layer, platform layer and infrastructure layer [3].The application layer is oriented to users, it implements the interaction mechanism between user and service provider with the support of platform layer. Users can submit tasks and receive the results through the application layer in the task scheduling process. The infrastructure layer is a set of virtual hardware resources and related management function, in which the implementation of internal process automation and optimization of resource management can provide optimal dynamic and flexible external infrastructure services. Furthermore, the platform layer is a set of software resources with versatility and reusability, which can provide an environment for cloud application to develop, run, manage and monitor. According to the above architecture, two levels scheduling model [4] are adopted in this paper as shown in Fig.1.

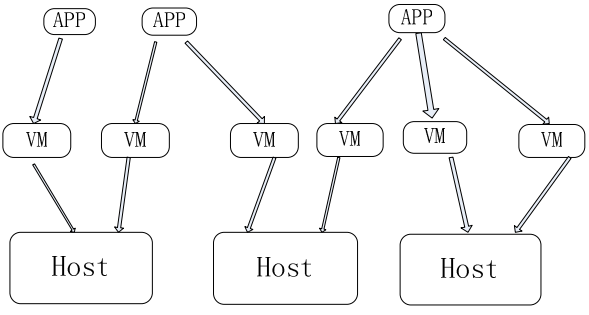

Fig. 1. Two levels scheduling Model in cloud computing

As shown in Figure 1, the first level scheduling is from the users' application to the virtual machine, and the second is from the virtual machine to host resources. In this two levels scheduling model, the first scheduler create the task description of a virtual machine, including the task of computing resources, network resources, storage resources, and other configuration information, according to resource demand of tasks. Then the second scheduler find appropriate resources for the virtual machine in the host resources under certain rules, based on each task description of virtual machine. Via the two levels scheduling, the task can obtain the required resources, and it would not lead to the resource allocated to some tasks is less than requirement and increase the implemental time while others are more than their requirements and lead to the waste of resources.

In order to describe the two levels scheduling model, task model and host resource model is established. All tasks in this paper are computational ones, only the Meta task is considered, and the tasks are independent each other, and the execution of the task replication is also not considered. The task model is described as follows:

The set of tasks is $T=\{t_0,t_2,\cdots,t_{n-1}\}$, and the number of tasks is $n=|T|$.The T, $t_i(i\in[0,n-1])$ indicates task i, $t_i=\{tId,tRr,tSta,tData,tVmch,tVm\}$, each property is defined as follows:

1) The task identification is defended as tId.

2) The required resource of one task is defined as $tRr=\{tC_0,tC_2,\cdots,tC_{k-1}\}$, where k is the number of resource type, $tC_j=\{j\in[0,k-1]\}$ is the ability of each resource.

3) The state of task is defined as $tSta$, where $tSta=\{tFree,tAllo,tSche,tWait,tExec,tComp\}$.

4) The relative data of task is defined as $tData$, including the computational amount tC, the input data tI and the output data tO, etc.

5) The description of virtual machine for a task is defined as $tVmch=\{tId,tRr,tData\}$, including the task identification tId, the required resource tRr and the related data $tData$.

6) The virtual machine of task is defined as $tVm=\{tId,thId\}$, where tId is the identification of virtual machine and $thId$ is the host identification allocated by the virtual machine.

The host holds all the physical resources for the task implementation, including computational resources, storage resources, network resources, and other hardware devices and the model is described as follows:

The set of host resources is defined as $H=\{h_0,h_1,\cdots,h_{m-1}\}$, and the size of set is $m=|H|$. In the H, $h_j(j\in[0,m-1])$ indicates Host j, $h_j=\{hId,hTcap,hFcap,hData\}$, each property is defined as follows:

1) The identification of host resource is defined as hId.

2) The total service ability that the host resource can provide is defined as $hTcap=\{hTc_0,hTc_1,\cdots,hTc_k\}$, where k is the number of resource type, $hTc_j(j\in[0,k-1])$ is the service ability of each resource.

3) The available used resource of host is defined as $hFcap=\{hFc_0,hFc_1,\cdots,hFc_k\}$.

4) The relative data of host is defined as $hData$, including the input bandwidth hIB and the output bandwidth hOB.

In this scheduling model, because the task can change dynamically, including the task arrival time and the demand for resources, which requires the corresponding virtual machine can change dynamically according to the change of task to deploy the resource in order to meet the dynamic demand. In this process, if the virtual machine requires additional resources and they can not be provided by the host. Then the virtual machine migration technology will be used. Two migration strategies can be taken: the first one is to migrate the virtual machine to a host and allocate the additional resources for the machine, and another one is to migrate the other virtual machines out of this host to vacate the free resources for the virtual machine.

3 Scheduling Algorithm

The load of virtual machine discussed in this paper is expressed by the predicted executing time of tasks running in the virtual machine, named as VL_i [5].And the load of host is expressed by the average load of the virtual machine that run on it, named as HL_i, it is defined as: $HL_i = \frac{\sum_{j=1}^{n} VL_j}{n}$, where the n is the number of virtual machines that run on the host.

From the HL_i, the average load value $avgl$ and load balancing evaluating value B of cloud computing environment can be defined as follows:

$$avgl = \frac{\sum_{i=1}^{m} HL_i}{m} \tag{1}$$

$$B = \frac{\sqrt{\sum_{i=1}^{m} (L_i - avgl)^2}}{m} \tag{2}$$

In the equations above, the number of hosts is m, the smaller value B the better load balancing and the bigger value B the worse load balancing.

In order to meet users' requirements and increase the utilization of resources, a scheduling algorithm based on load balancing is proposed in this paper. The algorithm is based on the former scheduling model discussed, considering the flexibility and virtualization features of cloud computing, it is divided into two levels scheduling, one is the mapping from task to a virtual machine, another is mapping from the virtual machine to host resources. Generally, for the requirement of the task, users want to get the best response time. Therefore, only task response time and the demand for resources are considered in this paper. At the same time, because tasks are dynamic, they may arrive randomly. If the tasks arrive at same time, they will be sorted ascending according to the resource applied by users. And if they arrive at different time, they will be sorted according to the time sequence arrived. The steps of this algorithm are described as follows:

Step1: According to the host resource model, establish the host resource set $H = \{h_0, h_2, \cdots, h_{m-1}\}$ and sort ascending order of their processing power.

Step2: According to the task model, establish the task set $T = \{t_0, t_2, \cdots, t_{n-1}\}$ .In this process, the first level scheduler establish the virtual machine description according to the properties of task ,providing configuration information for allocation of resources and creation of the virtual machine.

Step3: According to the virtual machine description of Task $t_i \in T$, select a host resource h_j that can meet the required resources and the load is lightest. If the host exists, create the virtual machine and allocate the required resource for it, then update the available resources $hFcap$ of Host h_j, otherwise take the Task t_i to the tail of the task queue and waiting for the next scheduling.

Step4: If the resource requirements of the Task t_i increase, find whether the host whose virtual machine of Task t_i run on can meet the additional required resources, if it exists, allocate the additional required resources for it, reconfigure the virtual machine, and then update the host's available resources. Otherwise, the virtual machine is migrated to the host with lightest load and the additional required resources to execute continuously.

Step5: If the resource requirements of the Task t_i reduce, release the excess resources that the virtual machine occupied, and update the available resources hold by the host.

Step6: If Task t_i has been completed, then destroy the virtual machine of Task t_i and release the occupied resources for the other unfinished tasks.

Step7: Calculate the load balancing evaluating value B in current environment, if B is greater than the threshold value B_0, that indicates the load balancing state is worse, select a virtual machine with lightest load and migrate it to the host which can meet the resource requirement with the lightest load.

Step8: Repeat step3 to 7 until all tasks are completed.

In the above algorithm, the virtual machine is scheduled to the host with lightest load each time. The advantage is to avoid overloading for the host hold more resources. If the current virtual machine is scheduled to a host, as the computational amount increase, leading to the virtual machine's load is heavy, resulting in load imbalance, then take the dynamic migration operation, keeping load balance in current environment.

4 Simulation Experiment and Result Analysis

In this paper, the CloudSim toolkit is used to simulate the scheduling algorithm discussed above [6]. In order to compared with the performance under different environments, a cloud computing environment and a grid computing environment with one hundred host node is set up to implement scheduling of one hundred to one thousand independent tasks, which demand can be dynamically changed during execution. This simulation mainly validates the advantage of the makespan and the resource utilization between this scheduling mechanism in cloud computing and the scheduling mechanism in paper [7] in grid computing, the expression of resource utilization are defined as follows:

$$RUsed = \sum_{j=0}^{m-1} \frac{Rt_j}{Rc_j} \Big/ m \tag{3}$$

In equation (3), the total execution time in host h_j is Rt_i, the actual occupied time in host h_j is Rc_j, and the number of host is m .The simulation result are described as Fig.2 and Fig.3:

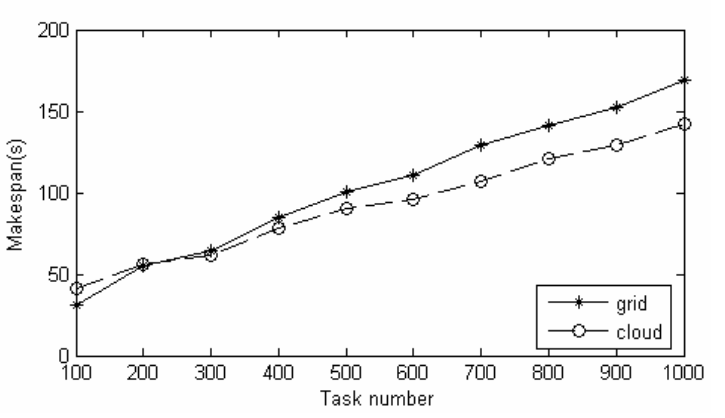

Fig. 2. Makespan Comparation

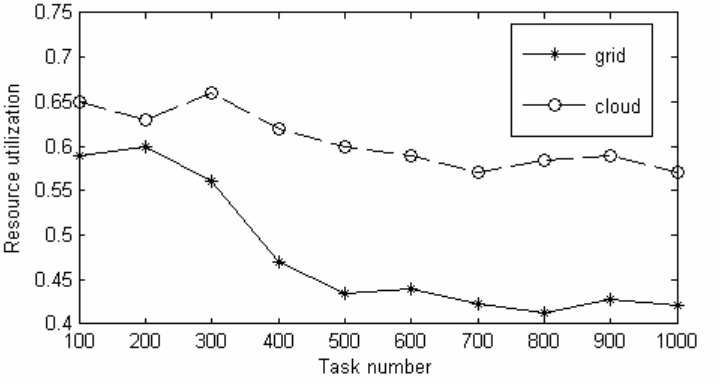

Fig. 3. Resource utilization Comparation

As shown in Figure 2, with the task number increasing, the makespan in cloud environment and grid environment are also increased. When the number of tasks and hosts difference is not large, the makespan in grid environment is lower than cloud environment. It is because the resources are allocated to a task is just it required in cloud computing, while the whole host resources are allocated to a task in grid computing. However, with the increase of ratio of number of tasks and hosts, the makespan in cloud environment is significantly less than it in grid environment, which is due to the flexibility in cloud computing, in which the resource can be allocate to the task dynamically stretching. As indicates in Figure 3, the resource utilization in cloud environment is significantly higher than grid environment and maintains a relatively stable level. The virtualization in cloud computing is considered fully in the scheduling mechanism for this paper, in which multiple virtual machines can be run in one host, when the load is imbalance, the dynamic migration approach is used to achieve a relatively load balancing and improve resource utilization.

5 Conclusions and Future Work

The traditional task scheduling in grid computing is to schedule the task directly to the host resources to execute, and it is not well to meet the dynamic requirements of users. A two-level scheduling mechanism based on load balancing is discussed in this paper. The scheduling mechanism take into account the dynamic requirements of users and the load balancing in cloud environment, therefore, it meets the dynamic requirements of users and increases the resource utilization. Through the CloudSim toolkit, we simulate this scheduling mechanism, proving the well scheduling effect. In the future work, we will consider more users' requirements, such as bandwidth, cost, etc., to establish a precise description of model for the users' requirements, to further improve the scheduling mechanism.

Acknowledgment

This work was supported by Chongqing Municipal Education Commission, NO.KJ090519.

References

1. Rimal, B.P., Choi, E., Lumb, I.: A Taxonomy and Survey of Cloud Computing Systems. In: Fifth International Joint Conference on INC,IMS and IDC, vol. 218, pp. 44–51 (2009)
2. Foster, I., Zhao, Y., Raicu, I., Lu, S.: Cloud Computing and Grid Computing 360-Degree Compared. In: Grid Computing Enviroments Workshop, pp. 1–10 (2008)
3. Armbrust, M.: Above the Clouds: A Berkeley View of Cloud Computing. In: EECS Department, University of California, Berkeley (2009)
4. Sadhasivam Dr., S., Jayarani, R.: Design and Implementation of an efficient Two-level Scheduler for Cloud Computing Environment. In: International Conference on Advances in Recent Technologies in Communication and Computing, vol. 148, pp. 884–886 (2009)
5. Xiang-hui, P., Er-hu, Z., Xue-yi, W., Guang-feng, L.: Load balancing algorithm of multi-cluster grid. Computer Enineering and Applications 45(35), 107–110 (2009)
6. The CLOUDS Lab.CloudSim: A Novel Framework for Modeling and Simulation of Cloud Computing Infrastructures and Services [EB/OL], `http://www.gridbus.org/cloudsim/`
7. Bing-han, L., Jun, H., Xiang, H., Qi, L.: Grid Load Schedule Algorithm Based on QoS. Computer Engineering 35(24), 96–98 (2009)