A New Load Balance Scheme for Heterogeneous Entities in Cloud Network Convergence

Jiaji Liu, Zhiwei Zhang(✉), Wangzhe Xu, Xinghui Zhu, and Xuewen Dong

School of Computer Science and Technology, Xidian University, Xi'an Shaanxi 710071, China
zwzhang@xidian.edu.cn

Abstract. For future Internet and next-generation network, the cloud networking convergence is one of the most popular research directions, and it has attracted widespread attention from academia as well as industry. Network adapting cloud and network cloudification are two dimensions in cloud network convergence that can break the closeness and independence between cloud and network. However, the techniques related to the network adapting cloud and network cloudification unavoidably introduce more heterogeneous devices, services and users. That disables the existing load balance schemes which are almost proposed for data centers in cloud computing environments, where the entities are typically standard hardware and software modules. As a result, the overhead and cost of load balance shcemes would be raised significantly in the progress of cloud network convergence. Therefore, in this paper, to make the most usage of heterogeneous entities and encourage the development of future Internet as well as next-generation networking, we propose a model and the requirements of load balance for heterogeneous entities in the convergence of cloud and network, then we present a concrete load balance scheme. Finally, we discuss the abilities and applications of our proposed model and scheme.

Keywords: Load balance · Cloud network convergence · Heterogeneous entities · Next-generation network · Private cloud computing

1 Introduction

Nowadays, the convergence of cloud and network is the development trend for future Internet and next-generation networking. Among the existing researches and applications in the convergence of cloud and network, two techniques, the network adapting cloud and the network cloudification, are playing key roles in realizing the convergence. However, these two techniques introduce heterogeneousness unavoidably. It is well known that the more different entities, the more complex to deal with. Furthermore, the previous load balance schemes have been designed for data centers in cloud computing environments or communication network operators where the entities are typically standard hardware and software modules. Therefore, few of the previous cloud and network load balance schemes are still feasible in convergence environments.

J. Xiong et al. (Eds.): MobiMedia 2021, LNICST 394, pp. 22–32, 2021.
https://doi.org/10.1007/978-3-030-89814-4_2

1.1 Related Works

Many methods have been proposed to reduce energy consumptions in both academic and industrial fields. At the hardware level, there are techniques such as optimized memory and dynamic frequency conversion [1]. At the software level, the most important method to reduce energy consumption is resource scheduling, which can achieve load balancing while ensuring that energy consumption is as low as possible.

Load scheduling algorithms can be divided into two categories: one is the load balancing algorithm implemented on the physical machine with tasks as the scheduling unit [2, 3]; another is the load balancing algorithm implemented on the physical machine with resources as the scheduling unit [4]. Besides, many researchers have conducted research on energy conservation. Online migration of virtual machines is an important method for server consolidation. However, migration has many negative effects, such as service interruption, network congestion and additional migration costs [5]. Therefore, it is also important to improve resource utilization and reduce energy consumption by migrating virtual machines. The static integration method [6–8] gives a mapping plan for virtual machines and physical servers, minimizing the number of servers or the overall cost. The dynamic integration method [9, 10] dynamically reconfigures the cluster through virtual machine migration to run on fewer nodes.

Our work in this paper is to analyze the status of services in the cloud and combine the ideas of load balancing and server integration in the environment of cloud networking integration to schedule resources in the cloud under different conditions. Therefore, the cloud network integration environment can be seen as a private cloud computing platform with a group of physical computers, the software-defined network and other technologies are used to manage the communication among the physical and virtual machines.

2 Contributions

In this paper, we focus on the problem of how to minimize the overhead and cost of load balance in cloud network convergence environment when taking the heterogeneousness into account. Our main contribution is that we propose the model and requirements of load balance for heterogeneous entities in the environment of cloud network convergence, and we present a concrete load balance scheme based on the proposed model.

2.1 Organization

In Sect. 2 we present the model and requirements of load balance for heterogeneous entities in cloud network convergence, then we describe the architecture of the load balance system in Sect. 3, with which we propose a new concrete load balance scheme in Sect. 4. In Sect. 5, we analyze and discuss the proposed scheme. Finally, Sect. 6 concludes this paper and discusses our future work on load balance in the environments of future Internet and next-generation networking.

3 The Model and Requirements of Load Balance for Heterogeneous Entities

In this section, we present the model and requirements of load balance for heterogeneous entities.

3.1 Entities Model

In an enterprise's private cloud, assume there are N physical machines (PM), and they can be represented as

$$\{\{PMa1, ..., PMan\}, \{PMb1, ..., PMbn\}, ..., \{PMx1, ..., PMxn\}\}, \tag{1}$$

where every physical machine is classified into one category represented as a set. In each set, machines have the same configuration. The existing researches [11] have found that power consumption and CPU utilization have a linear relationship. Therefore, the previous work uses the average power consumption measured when the load of a physical machine is 80% to 90%, called PPM. In short, PPM means the power consumption of PM. Based on the classification set of physical machines, the power consumption is divided into sets

$$\{\{PPMa2, ..., PPMa\mathrm{n}\}, \{PPMb1, ..., PPMbn\}, ..., \{PPMx1, .., PPMxn\}\}. \tag{2}$$

The previous work uses the average performance efficiency of physical machines measured when the load of a physical machine is 80% to 90%, called FPM. In short, FPM means the performance efficiency of PM.

$$\{\{FPMa2, ..., FPMan\}, \{FPMb1, ..., FPMbn\}, ..., \{FPMx1, ..., FPMxn\}\}. \tag{3}$$

Note that we should arrange the values in each set from small to large. Besides, we classify the services or requests. Assume there are N services (requests) in a private cloud, and they can be represented as

$$\{\{U\mathrm{a1},...,U\mathrm{an}\},\{\ U\mathrm{b1},..,U\mathrm{bn}\},...,\{\ U\mathrm{x1},...,U\mathrm{xn}\}\ \}\ . \tag{4}$$

Note that the sets in expression (4) and the sets in expression (1) have a one-to-one mapping relationship. Figure 1 illustrates the one-to-one correspondence between request sets and physical machine sets. Furthermore, there is a many-to-many relationship between requests and physical in the corresponding sets.

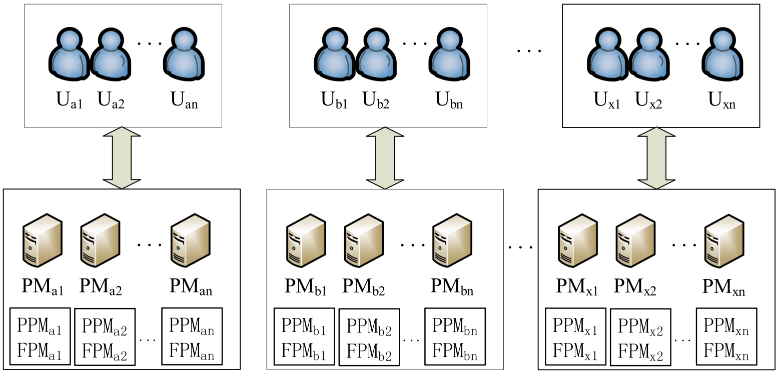


Fig. 1. Correspondence between requests and physical machines.

3.2 Load Balance Requirements

In private clouds, the actions of users, services and other entities could present kinds of regulations according to some contexts.

3.2.1 Service State Transition

In order to quantify the periodic and regular changes in services or requests, we introduce the strategy of service state transition (SST) in this section, in which we divide the service cycle into several time segments. Typically, a private cloud data center can also be divided into three states: idle (T0), normal (T1) and busy (T2). Change between states is determined by the results of historical value and cloud data center monitoring module, shown as Fig. 2.

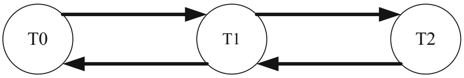


Fig. 2. Service state transition.

3.2.2 Scheduling Strategy

We apply resource scheduling method in SST to realize the load balance of cloud data center resources and the least energy consumption in each state. The core of scheduling strategy is a load balancing resource scheduling method and the underlying technology of load balancing resource scheduling method is resource migration. Considering green computing, virtual machines should be migrated to the lowest energy-consuming server. The technique of live or online migration is being used for efficient resource allocation in cloud computing. In order to select the virtual machine to be migrated, various threshold-based methods are introduced in. These thresholds are set to determine whether the server is over-utilized or under-utilized. In this paper, with the idea of Maximum Correlation

Policy (MC), we propose a new virtual machine migration strategy according to the previous SST.

3.2.3 The Minimum Energy Consumption Strategy

In this paper, we characterize the energy consumption of different servers as a fixed value according to the load value. We introduce the idea of server integration into our proposed SST strategy. So that, the problem of server integration can be saved by two steps: (1) choose the right service for scheduling and turn off the right physical machines; (2) choose the right physical machines for integrated services.

4 Design of Load Balance System Architecture

In this section, we present the system architecture of the resource scheduling model.

4.1 Service Request Learner

When users submit their service requests, service request learner (SRL) records the number of times each service accesses the cloud data center on a time scale, so as to judge the service status and the overall status of the data center. Further, SRL can judge the status of the data center in advance with the learned periodic data values, so that relevant strategies can be selected faster.

4.2 Local Resource Monitor

The local resource monitor (LRM) collects information about allocated VMS and containers running on physical machines. The LRM can save the resource utilization history of its host, and the local resource monitor monitors physical machine resource usage in real-time. At the last, LRM will transmit its data to the data center physical machine.

4.3 Global Resource Scheduler

The global resource scheduler (GRS), shown as Fig. 3, works at the central physical machine. The GRS is responsible for detecting the current load conditions (peak and nonpeak) by using the host utilization information stored in the LRM. The GRS also stores the load threshold, energy consumption value, and performance value of each physical machine. The communications between LRM and GRS are enabled by the sharing of common data structures and by using regular information polling.

4.4 Physical Machine

In the cloud, PM provides the hardware infrastructure for creating virtual machines, it can also be used to hold multiple containers. In other words, PM is a physical platform for virtual machines (VM) and containers. So, in this paper, we see PM as the basic building block in architecture.

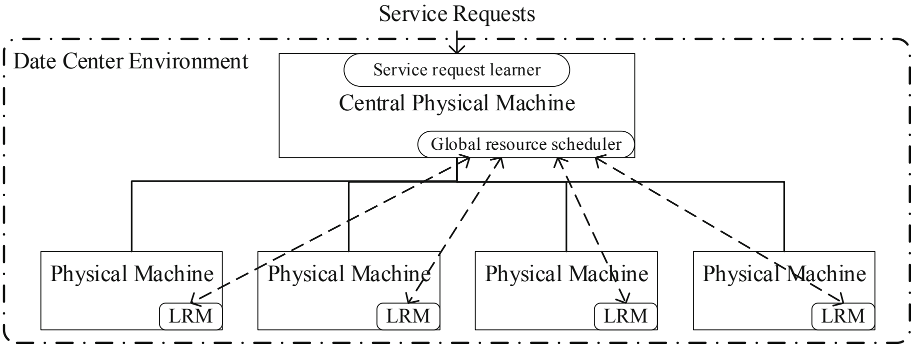


Fig. 3. Model Architecture Diagram.

5 Load Balance Scheme in Cloud Network Convergence

In this section, we present the details of the model and the design process.

5.1 From Idle State to Normal State

When SRL finds that the service volume of the entire system begins to exceed the set threshold, or when GRS finds that system resource usage starts to rise to a certain threshold the system state changes from T0 to T1. Thresholds are generated by periodic SRL and GRS records. When data center status changes from idle (T0) to normal (T1). In each PM cluster, the currently active PM set is called *SL*, and the currently dormant PM set is called *Si*. *Rsi* is used as the resource threshold of the set *SL*.

LRM gets the number of instances in each PM and gets the resource usage of each PM, called *Rpu*. LRM obtains the remaining resources of each PM, called *Rpi*. The energy consumption of each PM is represented by *PPM*. The performance efficiency of each PM is represented by *FPM*. When a service or request enters the data center, match its corresponding virtual machine instance, called VM. The amount of resources used by VM is called *RVm*.

When data center status changes from idle (T0) to normal (T1). And when a request corresponds to an instance on an inactive PM in the cluster. Moreover, the resources of active physical machines in the current cluster meet the resource requirements of the instance, the instance can be migrated to the active cluster. the problem of the location of instance can be saved by two steps: (1) selection of the active physical machine (2) choose the right new physical machine. For the selection of active PM, we introduce the idea of improving the minimum number of connections as the criterion. For each PM in the cluster, LRM obtains the amount of resource used on the PM, called *RVmi*; the total amount of resources in PM, called *Rpi*; the number of VM instances in PM, called *Gi*. Based on *(Rvmi/Rpi) *Gi*, select the PM with the smallest value,

$$Min(RVMi/RPi) * Gi \ (i = 1, 2, 3, ..., n). \tag{5}$$

Figure 4 illustrates the situation when the service corresponding instance is on the PM, this PM can be in a dormant cluster or in an active cluster.

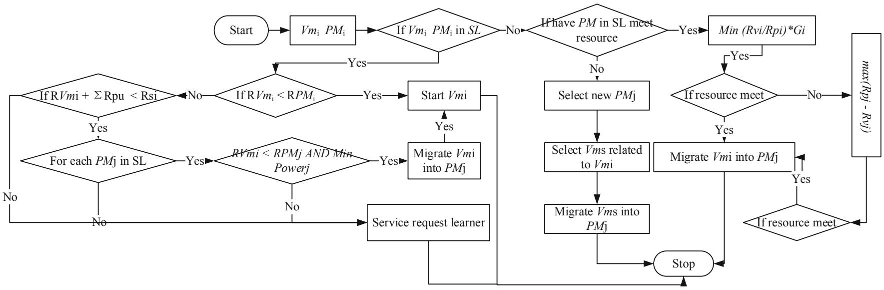


Fig. 4. Flowchart for VM instance in sleeping PM cluster.

5.2 From Normal State to Busy State

When SRL detects the status of the cloud data center from normal (T1) to busy (T2) through historical data, or when GRS detects that the used resources of the data center are approaching the threshold resource value, we judge that the status of the data center has changed. We have done the following to balance the load while reducing energy consumption as much as possible. (1) Detect whether the VM carried by the active PM in each existing cluster can be migrated. (2) If no PM is detected, add the dormant PM to the active cluster and perform load balancing. (3) If a PM is detected, the data center can shut down the existing PM and select a new PM cluster to ensure the load while reducing energy consumption.

We judge whether the instance in PM can be migrated through the dual-threshold method and prediction method. The level corresponding to each instance is recorded in GRS, and the instances that handle special services have a higher level which cannot be easily moved. SRL records the number of times a user accesses the cloud center by time. Only instances whose threshold is lower than the threshold set in SRL and GRS can be scheduled. LRM records the resource consumption of an instance on a time scale. According to the recorded value of LRM, it is found that the corresponding value of the instance will be higher than the threshold after a short period of time, so the instance will not be migrated.

When all instances on the PM can be migrated, a new PM collection will be obtained. The new PM needs to meet the following,

$$\sum PPMi < \sum PPMj + Power, \tag{6}$$

$$\sum Ri > \sum Rj + \nabla Rt * \alpha + Rp * \beta. \tag{7}$$

Table 1. Description of the parameter in the formula (6) and (7).

Parameter	Description
$\sum PPMi$	Total energy consumption value generated by the selected new PM set
$\sum PPMj$	Total energy consumption generated by the selected PM set by step (1)
Power	The value of the minimum energy consumption in the dormant PM
$\sum Ri$	The total resource value generated by the selected new PM set
$\sum Rj$	The total resource generated by the selected PM set by step (1)
∇Rt	Maximum used resource difference between in T1 state and in T2 state
α	When GRS has no recorded value, it is set to 0. Otherwise, it is set to 1
β	When GRS has no recorded value, it is set to 1. Otherwise, it is set to 0

Figure 5 illustrates the process of changing the cloud data center from T1 state to T2 state, where the instance migration strategy is a migration strategy based on the relevance idea mentioned above.

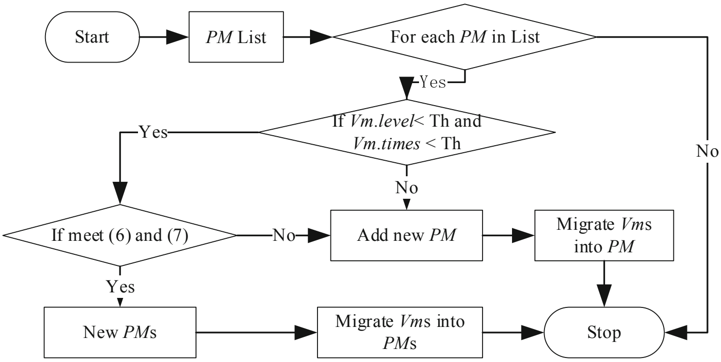


Fig. 5. Flowchart for the cluster status from T1 to T2.

5.3 From Busy State to Normal State or Form Normal State to Idle State

When SRL detects the status of the cloud data center from busy (T2) to normal (T1) or from normal (T1) to idle (T0) through historical data, or when GRS detects that the used resources of the data center below the threshold resource value, we judge that the status of the data center has changed. In the process of state transition, as business processing decreases, server idle resources increase, and servers can be integrated by five steps: (1) LRM obtains the used resource *Ru* and idle resource *Rd* of the active PM in the cluster; (2) eliminate PMs that cannot be integrated through the dual-threshold method; (3) close the instances that are no longer in use, and close the PM when all instances are closed; (4) find PM pairs with *Ru* less than *Rd*. Otherwise, find PM with *Ru* less than the sum of

$Rd * \alpha$;(5) migrate all instances in the PM to other PM, and use the improved minimum number of connections to ensure load balance during migration, and the resources used by other PMs must not exceed the threshold. Then put the corresponding PM to sleep. Figure 6 illustrates the processing flow of reducing energy consumption by combining server integration ideas when the cluster status changes.

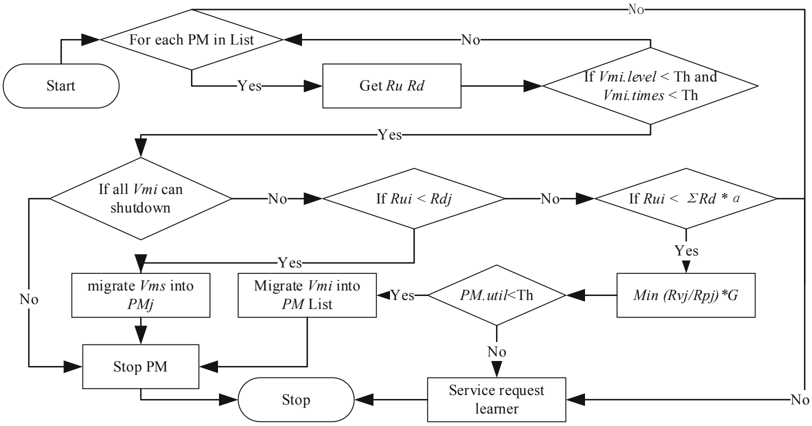


Fig. 6. Flowchart for the cluster status from T2 to T1 or T1 to T0.

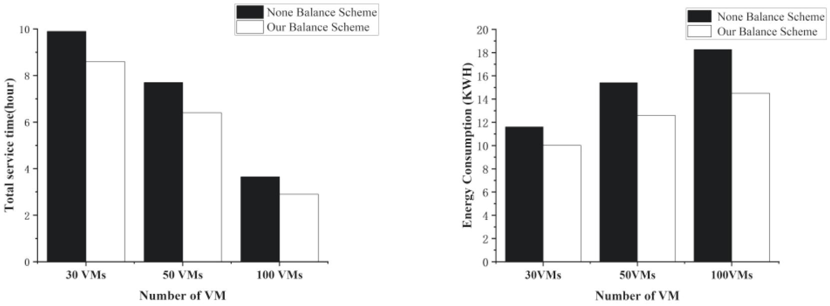
To this end, the above content fully demonstrates the process of resource scheduling in the data center according to state changes in a cycle of time.

6 Result and Discussion

In order to analyze the performance of the service state-based scheduling strategy proposed in this paper, the simulation experiment of this paper is carried out in the *CloudSim-Toolki*t [12] environment, and the *bindCloudletToVm* method in the *DatacenterBroke*r class is rewritten to map the services and instances accordingly. Figure 7 illustrates that compared with the ordinary private cloud data center, our proposed strategy can shorten processing time under the same workload.

We simulated a data center comprising 30, 50 and 100 VMs respectively. Each VM is modeled to have one CPU core with the performance equivalent to 1,000 MIPS, 2 GB of RAM and 10,000 Mb storage. Figure 7(a) illustrates that when the number of VMs is 30, the total service time of Our Balance Schedule (OBS) is 13.1% better than that of the None Balanced Scheme (NBS); when the number of VMs reaches 50, the total service time of Our Balance Schedule (OBS) is 16.8% better than the None Balanced Scheme (NBS). Notably, when the number of VMs is about 100, the total service time of OBS is 20.5% better than that of NBS.

An improved instance migration strategy based on correlation is introduced in the instance migration, and instance scheduling is carried out according to the correlation between services, which reduces the scheduling overhead to the greatest extent and



(a) Task processing time variance.

(b) Power consumption variance.

Fig. 7. Illustration of task processing time variance and power consumption variance

effectively reduces the number of instance migrations in the integrated cloud center. Figure 7(b) illustrates the change in data center energy consumption as the number of VMs changes.

7 Conclusion

Cloud networking convergence is the development trend of future Internet and next-generation networking. In this paper, we propose the model and requirements of load balance for heterogeneous entities in the convergence of cloud and network, and we present a concrete load balance scheme with a set of detailed algorithms. With the improvement of cloud network convergence, there should be paid more attention in the study of load balance.

Acknowledgment. This work is supported by National Key R&D Program of China (No.2017YFB1400700), Key R&D Program of Hainan Provincial (ZDYF2019202) and the Fundamental Research Funds for the Central Universities (JB210301).

References

1. Mezmaz, M., Melab, N., Kessaci, Y., et al.: A parallel bi-objective hybrid metaheuristic for energy-aware scheduling for cloud computing systems. J. Parallel Distrib. Comput. **71**(11), 1497–1508 (2011)
2. Panda, S.K., Jana, P.K.: Load balanced task scheduling for cloud computing: a probabilistic approach. Knowl. Inf. Syst. **61**(3), 1607–1631 (2019)
3. Singh, R.M., Paul, S., Kumar, A.: Task scheduling in cloud computing. Int. J. Comp. Sci. Inf. Technol. **5**(6), 7940–7944 (2014)
4. Nashaat, H., Ashry, N., Rizk, R.: Smart elastic scheduling algorithm for virtual machine migration in cloud computing. J. Supercomput. **75**(7), 3842–3865 (2019)
5. Bari, M.F., Zhani, M.F., Zhang, Q., et al.: CQNCR: optimal VM migration planning in cloud data centers. In: 2014 IFIP Networking Conference, pp,:1–9. IEEE (2014)

6. Zhang, S., Qian, Z., Luo, Z., et al.: Burstiness-aware resource reservation for server consolidation in computing clouds. IEEE Trans. Parallel Distrib. Syst. **27**(4), 964–977 (2015)
7. He, L., Zou, D., Zhang, Z., et al.: Developing resource consolidation frameworks for moldable virtual machines in clouds. Futur. Gener. Comput. Syst. **32**, 69–81 (2014)
8. Mohamadi Bahram Abadi, R., Rahmani, A.M., Alizadeh, S.H.: Server consolidation techniques in virtualized data centers of cloud environments: a systematic literature review. Software Pract. Exp. **48**(9), 1688–1726 (2018)
9. Biswas, J., Ray, M., Sondur, S., et al.: Coordinated power management in data center networks. Sustain. Comput. Inf. Syst. **22**, 1–12 (2019)
10. Choudhary, A., Govil, M.C., Singh, G., et al.: A critical survey of live virtual machine migration techniques. J. Cloud Comput. **6**(1), 1–41 (2017)
11. Kusic, D., Kephart, J.O., Hanson, J.E., et al.: Power and performance management of virtualized computing environments via lookahead control. Clust. Comput. **12**(1), 1–15 (2009)
12. Calheiros, R.N., Ranjan, R., De Rose, C.A.F., et al.: Cloudsim: a novel framework for modeling and simulation of cloud computing infrastructures and services. arXiv preprint arXiv:0903.2525 (2009)