Send Orders for Reprints to reprints@benthamscience.ae

The Open Cybernetics & Systemics Journal, 2014, *8*, 115-121

Open Access

A Load Balancing Strategy with Bandwidth Constraint in Cloud Computing

Jing Deng[1,*], Ping Guo[2], Qi Li[3], Haizhu Chen[1]

[1]Department of Computer, Chongqing College of Electronic Engineering, Chongqing, 401331, China; [2]College of Computer Science, Chongqing University, Chongqing, 400044, China; [3]China Telecom Corporation Limited Chongqing Branch, Chongqing, 401122, China

Abstract: The scheduling strategy on load balancing, used by data center in cloud computing, plays an important role on computing performance, and it is a key technique for the high-performance service. It directly controls the total performance and the efficiency of resource in cloud computing. In this paper, we introduce seven recent patents in the area of load balancing of cloud computing and discuss some classical load balancing algorithms (especially Min-Min algorithm). Based on Min-Min algorithm, a new improved algorithm named BCLL-Min-Min is proposed. It can satisfy the bandwidth constraint and implement the relative load balancing scheduling. The simulated experiments show that BCLL-Min-Min algorithm is widely available for the diverse and uncertain tasks in cloud computing. It improves the load balance in data center and enhances the throughput in the cluster.

Keywords: Cloud computing, load balance, Min-Min algorithm, BCLL-Min-Min algorithm.

1. INTRODUCTION

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (*e.g.*, networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction [1]. Cloud computing enables data centers to operate like the Internet by the process of making computing and storage resources to be accessed and shared as virtual resources in a secure and scalable manner. Cloud data centers accommodate a lot of computing and storage equipments, and they provide services by combining multi-node servers as a cluster [2]. In the cluster, the balance is dispatched to each node for promoting resource utilization and reducing the waiting-time of users. The advanced load balancing strategy is one of the key techniques for high-performance service, cost savings and improving the cluster throughput [3].

Up to now, several load balance strategies have been presented into cloud computing. Patent US 0,031,550, title "Method for improving the performance of high performance computing applications on cloud using integrated load balancing" [4], provides an expected cost set associated with an application-specific task of an application executing on a processing resource in a cloud computing environment, and communicating the expected cost set from the processing resource to a cloud management system.

Then a task to VM (virtual machine) assignment is determined based on the assignment of the application-specific task to the specific computational resource. Patent US 0,217,100, title "Method and system for load balancing content delivery servers" [5], presents a cloud computing content delivery system and it includes multiple content delivery servers (CDSs) configured to deliver content to multiple client devices. The client devices send requests to the CDS for the mirror list. Then they send content requests to the first entry in the list. Each CDS can update the mirror list by applying a load balancing algorithm and provide the mirror list to a client device in the event that one or more CDSs are unavailable. Patent US 0,166,645, title "Method and apparatus for load balancing in multi-level distributed computations" [6], provides load balancing in multi-level distributed computations. A distributed computation control platform determines closure capability data associated with respective levels of a computational architecture, wherein the respective levels include a device level, an infrastructure level, and a cloud computing level. The distributed computation control platform further determines to cause processing the closure capability data. Patent CN2013156525, title "Cloud computing load balancing method based on layering multiple agents" [7], relates to a cloud computing load balancing method based on layering multiple agents. Two of a plurality of nodes which are connected with a cloud computing platform through the network are used as a task monitoring agent and a resource monitoring agent, agents are joint management nodes of the cloud computing platform, each management node performs task allocation according to load conditions, different management nodes respectively take charge of monitoring, resource allocating and the like. So that a plurality of cloud computing tasks can be concurrently and effectively processed, and task processing capacity of the cloud computing platform is improved. Patent CN20101199455, title "Cloud computing load balancing method and equipment" [8], provides a cloud computing load balancing method and equipment. The load capacity of the server can be changed adaptively; and the proper service copy is selected according to the load capacity of the server to ensure comparatively uniform load distribution on each copy and realize adaptive load balance of the server.

The current scheduling algorithms takes short time, high degree of satisfaction and load balancing as the optimization goals and their algorithms can be classified into two types; static and dynamic. The static type, which is based on Poisson distribution of the tasks and exponential distribution of response time, includes RR [9], WRR [10]. They are available for the static server systems with small size and simple configuration since the resources utilization is described by a little static feature information and the resources are allocated to the coming task in proper sequence. When the tasks becomes diverse and uncertain, the effect of load balancing will be undesirable [11] with a difference of 10 to 100 times. The type of dynamic algorithms is presented subsequently, such as LC, WLC [10]. They will compute the load of the servers in a period of time according to different factors before the tasks arrive, so the coming task will be assigned to a suitable sever. The most difficult problem in this type of algorithm is how to choose an appropriate coefficient to compute the load of the servers. Patent CN 20131210315, title "Cloud calculating load balancing scheduling algorithm based on double-weighted least-connection algorithm" [12], discloses a cloud calculating load balancing scheduling algorithm based on a double-weighted least-connection algorithm. In the algorithm, no relatively large inclination of a load of each node is guaranteed when a system is in a long time operation state, summation of all task weights of each server and weight ratio of performances of the servers are calculated by a scheduler before task distribution, new tasks are distributed to the servers with smallest ratios. The algorithm shortens average accomplishment time of the cloud calculating service system, improves system efficiency and further improves a load balancing degree of each resource server in a cloud data center.

Min-Min and TD-Min-Min are heuristic scheduling algorithms [13, 14], and shortening the total completion time of all tasks is their optimization goal: Min-Min algorithm assigns the task according to the mapping between virtual machine and the task whose completion time is the shortest; while TD-Min-Min algorithm uses the mapping between virtual machine and the task whose completion time is the longest. Comparing with the static algorithms, the performance of the dynamic ones is better, for they know the real-time load of the servers. But their performance will be poor in cloud computing because they are not available for the tasks with different size and the resources with big diversity.

The remainder of this paper is organized as follows: Section 2 introduces the scheduling modes in cloud computing, Min-Min and LL-Min-Min algorithms [15]; in section 3, we present BCLL-Min-Min that can satisfy the bandwidth constraint and implement the relative load balancing scheduling; simulated experiments are analyzed in Section 4; the last section concludes this paper.

2. CLASSIC SCHEDULING MODEL AND ALGORITHM IN CLOUD COMPUTING

The architecture of cloud computing consists of three layers: application, platform and infrastructure layers [16]. This architecture makes the load scheduling in cloud computing a two-level mode [17]. The first level is the assignment between user application and virtual machine, and the task will be assigned to a suitable virtual machine according to the performance requirements or other limited factors of the task. While the second one is the assignment between virtual machine and host, it means that the second level selects the appropriate one from the physical hosts according to resource request of each task in the virtual machine and dynamically balances the loads of the physical machines. This model can satisfy all performance requirements of different tasks and the limited factors. What's more, it can avoid increasing the execution time for short of computing resource or wasting the resources since the distributed resource is more than the task needs.

Thanks to the virtualization technology, we can only concern ourselves with the first level scheduling of the model in clouding computing.

2.1. The Task Parameter Model in Cloud Computing

Let $C=\{C_1,C_2,C_3,\dots C_n\}$ be a given task sets of cloud computing. And each task C_i can be represented as:

$$C_i=\{ID,length,ExpB,Bwsat\} \tag{1}$$

where, ID uniquely identifies the task, and $length$ is given as its estimated completion time. $ExpB$ is the expected bandwidth and it is the unique measure of the users' satisfaction. Obviously, a task is assigned to a virtual machine that can perform the task.

In particular, $Bwsat$ is the measure of the satisfying bandwidth and it can be calculated by:

$$Bwsat=\begin{cases}1 & if \quad ExpB <= Bw\\ 0 & otherwise\end{cases} \tag{2}$$

where, Bw is the bandwidth of the virtual machine. Eq.(2) means, if the resource of a virtual machine can be used to perform the task, $Bwsat$ is set by 1. Otherwise, it set by 0.

2.2. The Virtual Machine Resource Model in Cloud Computing

Let $V=\{V_1,V_2,V_3,\dots V_m\}$ be the set of virtual machines in cloud computing data center, and each virtual machine V_i can be represented as:

$$V_i=\{ID,PE,mips,Ram,Bw,VL\} \tag{3}$$

where, ID uniquely identifies the virtual machine. PE is the number of executive unit in the virtual machine. $mips$ is the execution speed of the virtual machine. Ram is the memory size. Bw is the bandwidth of the virtual machine and it is compared with $ExpB$. VL is the workloads of the virtual machine. While some resource, such as external storage, are not considered, because they have little effect on the workloads of the virtual machine.

Based on the parameters of the virtual machines and the task parameter model, our new algorithm presented bellow will select the virtual machine in Eq.(3) for the tasks in Eq.(1) by a suitable load scheduling strategy.

2.3. The Load Model in Cloud Computing

Assume that an accurate estimation of the expected completion time for each task on each machine is known before the execution and contained within an ETC (namely, expected time to compute) matrix. $\mathrm{ETC}[i][j]$ is the estimated completion time of task i on virtual machine j. $\mathrm{ETC}[i][j]$ is influenced by many factors. In this paper, we assume that $\mathrm{ETC}[i][j]$ is only influenced by $length$, the length of task, and $mips$ is the execution speed of the virtual machine. To simplify the model, we have two conventions as follows:

1) The finishibg time of all tasks is fixed when many tasks are executed in a virtual machine simultaneously.

2) If executing time of a task is the minimum on a virtual machine, the task's executing time would be the minimum on other virtual machine.

3) If the execution speed of a virtual machine is the maximum, no matter which task, its execution speed is the maximum. Namely, the execution speed is not related to the task.

$\mathrm{ETC}[i][j]$ can be calculated by:

$$T_{ij} = length_i / mips_j \tag{4}$$

$$\mathrm{ETC}[i][j] = T_{ij} \tag{5}$$

Assume that there are n tasks and m virtual machines. The estimated completion time of n tasks can be calculated by Eqs. (4) and (5) and the result can be expressed by a matrix:

$$\mathrm{ETC} = \begin{bmatrix} T_{11} & \cdots & T_{1m} \\ T_{21} & \cdots & T_{2m} \\ \cdots & \cdots & \cdots \\ T_{n1} & \cdots & T_{nm} \end{bmatrix} \tag{6}$$

The tasks of different users have nothing in common with the processing ability of virtual machines in cloud computing data centers and they are differ from one another. All these increase the difficulties of defining the workload. In this paper, two assumptions are given as following shows.

The first assumption, the workload of a task is only related to $length$ of the task. Namely, the longer the length is, the heavier the workload is.

The second assumption, the processing ability of a virtual machine is only related to $mips$, the execution speed of virtual machine, without considering the bandwidth, the memory size and so on. $mips$ can be determined to weight according to its historical record.

The workload of virtual machine can be defined as VL and the workload caused by task i of virtual machine j can be calculated by:

$$VL_j = length_i / mips_j \tag{7}$$

We know from Eq.(4) , the $length_i / mips_j$ is equal to the T_{ij}. So, the workload of virtual machine is equal to the total sum of all tasks processed in the virtual machine:

$$VL_j = \sum T_{ij} \tag{8}$$

So the load balancing scheduling can be described as all the virtual machines have equivalent executing time of tasks. And a standard measuring load balancing is defined as:

$$\rho = T_{\min} / T_{\max} \quad (9)$$

where, $T_{\min}$ is the minimal and $T_{\max}$ is the maximal completion time of all virtual machines.

The results were obtained as follows according to Eq.(9).

1) The tasks are not scheduled when $T_{\max}$ is equal to 0.

2) There are idle virtual machines when ρ is equal to 0 and $T_{\max}$ is not equal to 0.

3) There is the highest load balance degree when ρ is equal to one, namely, the maximum is equal to minimum of workload. Moreover, the closer to 1 ρ is, the higher the load balance degree is.

2.4. Min-Min [13] and LL-Min-Min [15]

Up till now, many scheduling algorithm derived from Min-Min algorithm. Min_Min heuristic begins with the set U of all unmapped tasks, and then the set of minimum completion times is found. Next, the task with the overall minimum completion time is selected and assigned to the corresponding machine (hence the name Min_Min). Last, the newly mapped task is removed from U, and the process repeats until all tasks are mapped (*i.e.*, U is empty). Min_Min maps the tasks in the order that changes the machine availability status by the least amount that any assignment could. The expectation is that a smaller makespan can be obtained if more tasks are assigned to the machines that complete them the earliest and also execute them the fastest.

LL-Min-Min is one of the improved versions of Min-Min and it implements the relative load balancing scheduling. In this algorithm, the task with the relatively largest load and the minimum completion time is chosen, so each task has an opportunity to be performed in different virtual machine parallel and the utilization percentage of the virtual machines can be promoted.

The relative load can describe the difference of the completion times of each task in different virtual machines, so choosing the maximum different one from the current task can avoid increasing the load or the makespan.

3. BCLL-MIN-MIN ALGORITHM

When users use cloud computing service and ask for virtual machine resource, they raise some virtual machine performance requests. For example, bandwidth request of tasks, obviously service provider should satisfy its users and meet the user demands by distributing sufficient virtual machine resource. Simultaneously, the completion time must be short

Table 1. An example of BCETC matrix (unit: ms).

	M_1	M_2	M_3	M_1
C_1	10	11	17	7
C_2	N	13	15	24
C_3	N	N	N	22
C_4	N	17	12	27
C_5	N	N	18	12

enough, because it relates to the user's feeling directly. In conclusion, basing on LL-Min-Min algorithm and bandwidth constraint, we propose a load balancing strategy with bandwidth constraint in cloud computing, named as BCLL-Min-Min.

BCLL-Min-Min algorithm's optimistic goal is: loading balance; satisfy different task bandwidth request; short completion time.

For simplicity, we appoint task scheduling that meet two conditions:

1) There is no relation between two tasks;

2) When a task is distributed to the virtual machine, other tasks must wait until it is accomplished;

Base on the above scheduling, we raise scheduling discipline to realize BCLL-Min-Min algorithms optimize goal:

1) If a virtual machine satisfies the bandwidth requirements, the task will assign to the virtual machine.

2) If there are multiple virtual machines that meet bandwidth requirements, the task will be assigned to the one whose execution time is minimal.

3.1. The idea of BCLL-Min-Min

BCLL-Min-Min algorithm should meet user's demand for bandwidth and guarantee a certain load balance. The bandwidth requirements can be clarified by the user's request or by querying the historical information, and in BCLL-Min-Min algorithm, the choice of the bandwidth is based on the user's consideration.

For the bandwidth-demand tasks, ETC that will meet the bandwidth constraint matrices are named BCETC, and the core of the BCLL-Min-Min algorithm is BCETC. The estimated execution time in BCETC matrix is the same as before when virtual machine resource meet the user's demand. However, the task cannot be assigned to the virtual machine and the value of estimated execution time will be set N in the same place of BCETC matrix if it does not meet user's need

BCETC matrix as shown in the Table **1**, the bandwidth in virtual machine M_1, M_2, M_3, M_4 increases successively. Due to the lower bandwidth requirements of the task C_1, these 4 virtual machines satisfy the condition, the first line unchanged. But in the second line, M_1 does not meet the requirement, the value in the $[C_1, M_1]$ will be set to N, that is not mapped. Similarly, we can conclude the map between the other tasks and virtual machines.

Meanwhile, in order to achieve a certain level of load balance, the fewer virtual machines will be given Priority scheduling if they satisfy the bandwidth requirements, the procedure of this sort of scheduling is given below:

Step1: Only one virtual machine that meet their requirements.

Step2: At least two virtual machines that meet their requirements.

Step3: All virtual machines that meet their requirements.

Step4: All virtual machines that do not meet their requirements.

If we never adopt these scheduling, the tasks of greater coverage will be performed in the virtual machine with a low bandwidth, and it will result in a seriously uneven load. In order to ensure the task and virtual machines corresponding to the N in matrix BCETC never be assigned, we set N with a large value (greater than the sum of all the task completion time). Therefore, a task with largest gap in average and minimum executed time will be executed firstly. The less virtual machines meet the requirement, the greater the D-value is. This algorithm will preferentially select the task and meet these characteristics.

3.2. Algorithm Description

Algorithm: BCLL-Min-Min

Input: virtual machine parameter, tasks parameter

Output: mapping scheme

Procedure:

```
Step1: generate $C$ to record all tasks; modify $V$ to record all virtual machine; definite CTOV to record current mapping scheme; generate VMLoad to record all virtual machines' loading; Los is an array which has relatively loading of each task;
Step2: according to tasks' expected instruction length, virtual machines' processing speed and bandwidth-demand tasks, we can get the $ETC[i][j]$;
Step3: foreach $C_i$ in $C$;
    add VMLoad[j] to BCETC [i][j];
    foreach $C_i$ in $C$;
        foreach $V_j$ in $V$;
```

```
find minimum completion time of each task;
calculate average time for each task that remove the minimum time;
    get LL by calculating the D-value from average time and
    minimum time;
  end foreach
end foreach
choose the task-resource mapping of biggest D-value, and keep in
    CTOV;
According to ETC array refresh VMLoad, delete task that have
    done from $C$.
end foreach
End.
```

BCLL-Min-Min algorithm not only distributes tasks preferentially to whose relative loading is heavy but also adds user satisfaction. It solves Min-min algorithm's problems for optimize goal unicity and load imbalance.

4. SIMULATION EXPERIMENTAL RESULTS AND ANALYSIS

In order to verify the performance of the BCLL-Min-Min algorithm, this section uses cloud computing simulation tools CloudSim [9] to do a comparative experiment among BCLL-Min-Min algorithm, Round-Robin (CloudSim platform comes with round-robin scheduling algorithm, referred to as RR algorithm) and Min-Min scheduling algorithm (referred to as MM algorithm) in the completion time of task, load of the virtual machine and completion of the task bandwidth requirement classes. Patent CN2012194359, title "Cloud computing load balancing evaluation system and evaluation method" [18], invents a cloud computing load balancing evaluation system. In the system, a signal output end of a cloud cluster load information collection module is connected with a signal input end of an image conversion module, and the signal output end of the image conversion module is connected with a signal input end of a cluster balance analysis module. According to the invention, the load conversion picture can be generated by adopting the method for mapping the node comprehensive load value to the grayscale map and is analyzed according to the load image analysis method, thus achieving the purpose of accurately and wholly evaluating a load.

In our previous work, we take experiment and verification in terms of LL-Min-Min algorithm for the case of heterogeneous task and unfixed length of the tasks in heterogeneous environments. This paper compares and verifies the case of tasks with unfixed length based on the strength of Min-Min algorithm, observing the implementation of tasks in the bandwidth constraints.

This paper selects the following two sets of experimental data based on the standard of the total execution time of the task and load balancing of the virtual machine:

The first group: 100, 200, 300 tasks and 10 virtual machines' load scheduling. The main references are task span, virtual machine load and completion of the task in bandwidth requirements. All parameters are randomly generated, difference between tasks and virtual machine performance heterogeneous is little, and the ratio of bandwidth requirement task is 50%.

The second group: adjusting the ratio of the bandwidth requirement task, 40%, 60% and 80% respectively; observing the execution time of three algorithms with the effect of different ratio bandwidth requirement task.

As shown in Fig. (**1**), RR algorithm takes the longest completion time, BCLL-min algorithm followed; Min-min algorithm is slightly better than the BCLL-min algorithm in time span. RR algorithm is the clearly worst because it does not consider the characteristics of the virtual machine and tasks, and it assigns tasks to a virtual machine sequentially. Min-min algorithm has good results in the neat tasks; and it does not consider the bandwidth needs of the users, which means that it assigns the unsatisfied resources to users; as a result, Min-min algorithm is better than BCLL-min algorithm in execution time span. Overall, BCLL-min algorithm meets the basic need in bandwidth requirement, and it guarantees the completion time.

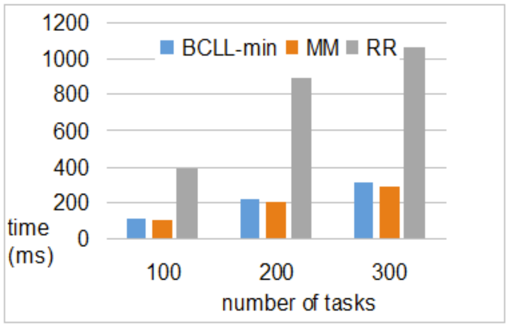

Fig. (1). Comparison of completion time .

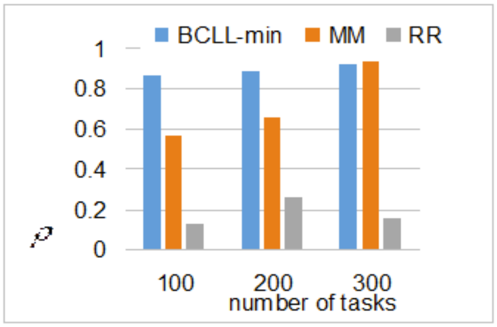

Fig. (2). Comparison of loading balance ρ .

The comparison of load balancing of the virtual machine is shown in Fig. (**2**). It can be seen that BCLL-min algorithm has the highest load balancing virtual machine under the bandwidth requirements at 50%. Because the bandwidth requirements classes of the task takes only 50%, and the remaining tasks make up for load inequality caused by bandwidth requirements to a certain extent based on the algorithm characteristics, achieving optimization goal based on load balancing. Performance of Min-Min algorithm is good under tasks in certain conditions. But Min-Min algorithm is still worse than BCLL-min algorithm, while load balance of these two algorithms perform well when the task increases. However, RR algorithm is the worst.

If tasks with bandwidth constraints can be performed by some virtual machines, the value of Eq.(2) is set by 1. 50% of the bandwidth requirement classes tasks, which the maximum is 50, all meet under the case of 100 tasks. Similarly, 200 tasks are 100 and 300 tasks are 150. The comparison of the completion of the bandwidth tasks is shown in Fig. (**3**). As Fig. (**3**) shows, BCLL-Min algorithm can satisfy bandwidth requirement classes. While Min-Min algorithm just can meet a little since it has not the optimization goal of bandwidth requirement. However, the RR algorithm will definitely have some tasks to meet the needs because of the allocation of a fixed rotation, and it equally matches Min-Min algorithm under the satisfaction of bandwidth requirement. Therefore, compared with a slight time difference in BCLL-min algorithm, BCLL-min algorithm achieves a big satisfaction difference forthe user tasks.

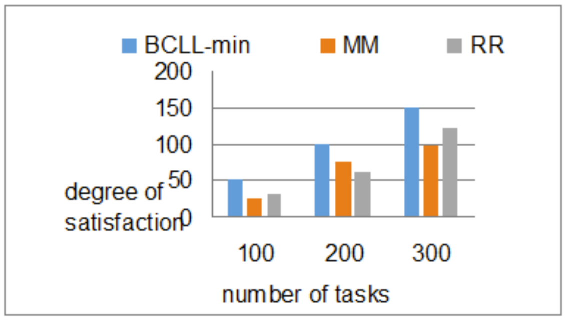

Fig. (3). Comparing the bandwidth satisfaction of the tasks.

In order to observe tasks with different ratio bandwidth constraints impact on load scheduling tasks, we adjust the ratio of bandwidth requirement tasks and increase bandwidth for tasks requirement in the second experiments. We design 200 heterogeneous and little different tasks and 10 heterogeneous virtual machine resources to reflect the comparison between BCLL-min algorithm and Min-Min algorithm based on Min-Min algorithm strengths. All parameters are still randomly generated. The experiments results of bandwidth satisfaction with three ratios are shown in Fig. (**4**).

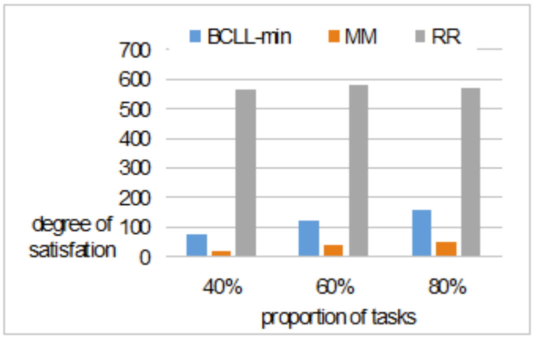

Fig. (4). Comparison of 200 different proportions tasks satisfaction.

From Fig. (**4**), we can see that BCLL-min algorithm can basically meet the bandwidth requirement of users. The growth rate of Min-Min algorithm and the RR algorithm is not significant with the increase in bandwidth ratio, and they do not meet bandwidth requirement well. However, with the ratio of bandwidth requirement tasks increasing, requirement of bandwidth constraints and limiting increases, meeting the bandwidth requirements will inevitably lose some time span.

Fig. (**5**) shows that BCLL-min algorithm makespan is increased with the limit of bandwidth requirements constraints, because the satisfied machine lessens after increasing the bandwidth limiting, and it sacrifices some completion time to meet the tasks. In contrast, Min-Min algorithm does not consider bandwidth requirements, so completion time with different ratio is not much different. However, in general, there will not be so much tasks of type of bandwidth requirement.

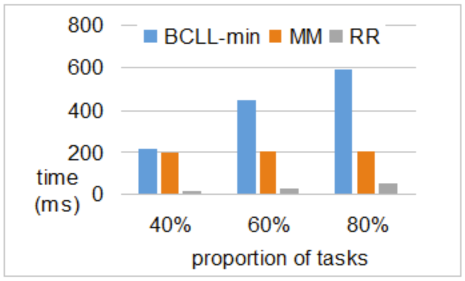

Fig. (5). Comparison of 3 proportions kinds of the time span.

Overall, BCLL-min algorithm is the best under various conditions, load balance and task completion satisfaction. But task completion time of BCLL-min algorithm will be slightly reduced with ratio of bandwidth requirement tasks increasing.

As clearly obtained from the experiments, BCLL-min algorithm is better than the other two algorithms in heterogeneous task and heterogeneous environments, and it meets bandwidth requirement. The following conclusions can be drawn from simulation experiments:

1) Under the number of tasks with little difference, the load balance of BCLL-min algorithm is the best. In order to meet bandwidth requirement, the total completion time of BCLL-min algorithm is not better than that of Min-Min algorithm, not considering the bandwidth requirement, but much better than that of RR algorithm.

2) Under the number of tasks with little difference, BCLL-min algorithm is the best in optimization goals and guarantees the load balancing, since adding bandwidth limitations.

3) BCLL-min algorithm will have some impact on task execution time with bandwidth constraints condition increasing, but it can meet bandwidth requirements tasks at any ratio. Meanwhile, BCLL-min algorithm will be as efficient as LL-Min-Min algorithm if there is no bandwidth requirement.

5. CONCLUSION

The computing tasks in cloud computing is not only complicated but also diverse. Considering on the heterogeneity of the virtual machines and diversity of the tasks in cloud computing, we improve classical Min-Min algorithm and LL-Min-Min algorithm and present BCLL-Min-Min algorithm, which can satisfy the bandwidth constraints and implement the relative load balancing scheduling. The experiments performed on simulation platform CloudSim show that BCLL-Min-Min algorithm is more available for scheduling the tasks in cloud computing and it can improve the load balance in data center and enhances the throughput in the cluster.

CONFLICT OF INTEREST

The authors confirm that this article content has no conflict of interest.

ACKNOWLEDGEMENTS

This work is supported by Natural Science Foundation Project of CQ CSTC 2012jjA40022) and the National Science Foundation for Young Scholars of China (Grant No. 61201347).

REFERENCES

[1] P. Mell and T. Grance, "The NIST definition of cloud computing", *Commun. ACM*, vol. 53, pp. 50-50, 2010.

[2] H. L. Zeng, B. F. Zhang and L. H. Zhang, "Virtual cluster constructing based on cloud computing platform", *Microelectron. Comp.*, vol. 27, pp. 31-35, 2010 (in Chinese).

[3] R. Buyya, "High performance cluster computing: Systems and architectures", *Michigan: Prentice Hall PTR*, 1999.

[4] A. R. Choudhury, T. George, M. Kedia, Y. Sabharwal and V.Saxena, "*Method for improving the performance of high performance computing applications on cloud using integrated load balancing*," U.S. Patent 0,031,550, January 31, 2013

[5] P. Goerner, G. Breen, R. H. Smith, S. C. Dandy and A. H. Bridge, "*Method and system for load balancing content delivery servers*," U. S. Patent 0,217,100, August 22, 2013

[6] S. Boldyrev, H. E. Laine, J. H. Kaaja, J. Honkola, V. Luukkala and I. J. Oliver, "*Method and apparatus for load balancing in multi-level distributed computations*," U. S. Patent 0,166,645, June 28, 2012.

[7] X. L. Tao, Y. Wang, Y. Pei, P. H. Li and Q. L. Zhou, "*Cloud computing load balancing method based on layering multiple agents*," C. N. Patent CN2013156525, May 22, 2013.

[8] C. Jin and X. Zhang, "*Cloud computing load balancing method and equipment*," C. N. Patent CN20101199455, October 27, 2010

[9] The clouds lab, "CloudSim: a novel framework for model-ing and simulation of cloud computing infrastructures and services", http://www.gridbus.org/ cloudsim/. September 2, 2011.

[10] E. Choi, "Performance test and analysis for an adaptive load balancing mechanism on distributed server cluster systems", *Future Generation Computer Systems*, vol. 20, pp. 237-247, 2004.

[11] E. Casslicchio and S. Tucci, "Static and Dynamic scheduling algorithm for scalable Web server farm", In the IEEE 9th *Euromicro Workshop on Parallel and Distributed Processing*, IEEE Conference Publications, 2001, pp. 369-376.

[12] L. Y. Zhou, X. P. Cui and J. Zheng, "*Cloud calculating load balancing scheduling algorithm based on double-weighted least-connection algorithm*," C. N. Patent CN 20131210315, October 2, 2013

[13] T. D. Braun, H. J. Siegel, N. Beck, "A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems", *Journal of Parallel and Distributed computing*, Vol. 61, pp. 810-837, 2001.

[14] A. Iyengar, E. MacNair and T. Nguyen, "An analysis of Web server performance", In *Global Telecommunications Conference*, IEEE Conference Publications, 1997, pp. 1943~1947.

[15] P. Guo, T. Li and Q. Li, "A scheduling strategy on load balancing in cloud computing" (in Chinese) (Accepted).

[16] S. Sadhasivam, N. Nagaveni, R. Jayarani, R.V. Ram, "Design and implementation of an efficient two-level scheduler for cloud computing environment", In *Advances in Recent Technologies in Communication and Computing*, IEEE Conference Publications, 2009, pp. 884-886.

[17] G. Liu, J. Li and J. C. Xu, "An improved min-min algorithm in cloud computing", In *Proceedings of the 2012 International Conference of Modern Computer Science and Applications*. Berlin, Springer, 2013, pp. 47-52.

[18] H. F. Huang, P. Wang, K. Cao, J. Y. Dong, H. Tang, L. Chen, C. Ren and Y.Huang, "*Cloud computing load balancing evaluation system and evaluation method*," C. N. Patent CN2012194359, August 22, 2012.

Received: September 22, 2014 Revised: November 30, 2014 Accepted: December 02, 2014