Send Orders for Reprints to reprints@benthamscience.ae

The Open Cybernetics & Systemics Journal, **2014,** *8,* **111-114**

Open Access

Task Scheduling Algorithm based on Greedy Strategy in Cloud Computing

Zhou Zhou and Hu Zhigang*

School of Software, Central South University, Changsha, 410004, P.R. China

Abstract: In view of Min-Min algorithm prefers scheduling small tasks and Max-Min algorithm prefers scheduling big tasks led to the problem of load imbalance in cloud computing, a new algorithm named Min-Max is proposed. Min-Max makes good use of time of greedy strategy, small tasks and big tasks are put together for scheduling in order to solve the problem of load imbalance. Experimental results show that the Min-Max improves the utilization rate of the entire system and saves 9% of the overall execution time compared with Min-Min. As compared to Max-Min, Min-Max improves the utilization rate of the entire system and the total completion time and average response time are saved by 7% and 9%, respectively.

Keywords: Cloud computing, greedy strategy, load balancing, Min-Min, Max-Min.

1. INTRODUCTION

Cloud computing [1, 2] is derived from grid computing, and now it has been widely used in various fields. In cloud computing, the problem of task scheduling has become a hot issue. At the same time, it is also a NP hard problem [3, 4].

Resources in cloud computing costs, and it depends mainly on the use of time. Therefore, the main goal of task schedule is to decrease total completion time, overall execution time, average response time and to improve the utilization rate of the entire system under the condition of meeting the QoS (Quality of Service). At present, many researches proposed different algorithms to solve this problem. In paper [5], authors modify standard heuristics for task assignment in perfectly predictable environments, and put forward an extension of Sufferage called XSufferage. Experimental results from simulation show that XSufferage can achieve better performance than other heuristics. However, XSufferage can lead to the problem of load imbalance. To better use the tremendous capabilities of the distributed system, authors present a new scheduling algorithm named Min-Min [6]. The experimental results show that Min-Min can lead to significant performance gain for a variety of scenarios. But, the new heuristic prefers scheduling small tasks. Moreno *et al.* presents a new algorithm called Max-Min [7], and it prefers scheduling big tasks. In paper [8, 9], authors use General Algorithm to solve the problem of task scheduling. In addition, there are some improved algorithms based on QoS such as [10, 11]. These algorithms work well in some aspects, but in other aspects there are some shortages. In this paper, we propose a new scheduling algorithm named Min-Max, which is based on two conventional scheduling algorithms, Min-Min and Max-Min, to use their cons and at the same time, cover their pros.

Experimental results show that the Min-Max improves the utilization rate of the entire system and saves up to 9% of the overall execution time compared with Min-Min. As compared with Max-Min, Min-Max improves the utilization rate of the entire system. The total completion time and average response time is saved by 7% and 9%, respectively.

2. BASIC CONCEPT AND GOAL OF TASK SCHEDULE

2.1. Basic Concept

It is necessary to introduce some basic concepts in order to enhance description:

1) Total completion time: It represents the sum of completion time for each task. Assuming that completion time of the first task costs 10 ms, the second task 20 ms and the third task 30 ms. Therefore, the total completion time of the three tasks is 60 ms (10+20+30=60).

2) Overall execution time: It refers to the completion time of all tasks.

3) Task response time: It represents the time taken for the task to be completed after entering cloud computing system.

4) Utilization rate of the entire system: It refers to the level of free resources. At the same time, it is also an important indicator for measuring whether a schedule algorithm is good or bad.

2.2. Goal of Task Schedule

Formal description of task scheduling algorithm is as follows: In a cloud environment, there are n tasks (T_1, T_2, …, T_n) and r resources (M_1, M_2, …, M_r). The goal of task scheduling is that n tasks are assigned to r resources in some manner, meeting the following requirements.

1) Optimal makespan: It refers to the minimum overall execution time.

1874-110X/14

2) Load balancing: It means that the majority of resources have tasks to be run.

3) Service requirements: The tasks being processed have to meet the QoS requirements.

3. THE ANALYSIS OF MIN-MIN AND MAX-MIN ALGORITHM

3.1. The Analysis of Min-Min Algorithm

Min-Min algorithm prefers assigning small tasks to fast resources to run so that the total completion time is a minimum. Min-Min, firstly, calculates the minimum completion time for each task which is assigned to the related resources, and then chooses a minimum value from minimum completion time. In other words, Min-Min selects minimum value twice. The description of Min-Min algorithm is as follows:

1) Calculate the minimum completion time for each task which is assigned to the related resources.

2) Choose a minimum value from the minimum completion time.

3) Finish task scheduling and update related variables.

4) Repeat the above steps until all tasks are assigned.

Min-Min can ensure the total completion time of tasks is a minimum. But there is a shortage that Min-Min leads to fast resources with heavy load and slow resources with light load. That is to say Min-Min causes lower utilization rate of the entire system.

3.2. The Analysis of Max-Min Algorithm

Difference from Min-Min, Max-Min prefers scheduling big tasks. The description of Max-Min is as follows.

1) Calculate the minimum completion time for each task which is assigned to the related resources.

2) Choose a maximum value from these minimum completion time.

3) Finish task scheduling and update related variables.

4) Repeat the above steps until all tasks are assigned.

Max-Min is better than Min-Min in most cases. However, Max-Min also causes lower utilization rate of the entire system.

4. MIN-MAX ALGORITHM

4.1. The Main Idea of Min-Max

In view that the Min-Min algorithm prefers scheduling small tasks and Max-Min algorithm prefers scheduling big tasks led to the problem of load imbalance in cloud computing, a new algorithm named Min-Max is proposed. Min-Max makes good use of time for greedy strategy, small tasks and big tasks are put together for scheduling. The main idea is as follows.

1) Calculate the minimum completion time for each task which is assigned to the related resources.

2) Choose a minimum value and a maximum value from the minimum completion time to make up a pair of tasks.

3) Finish the pair of tasks scheduling and update related variables.

4) Repeat above steps until all tasks are assigned.

4.2. The Pseudo-code of Min-Max

Before presenting the pseudo-code of Min-Max, some explanations are as follows.

RT(j): It refers to prepare time of resource M_j.

ETC(i, j): It represents prediction execute time of task T_i which is assigned to resource M_j.

CT(i, j) : It represents prediction finish time of task T_i which is assigned to resource M_j, satisfying CT(i, j)= ETC(i, j) + RT(j).

MCT(i): It refers to the minimum finish time of task T_i.

host _MCT (i): It means that Task T_i is assigned to resource host _MCT (i).

The pseudo-code of Min-Max is as follows by Fig. (**1**):

```
1 for each task Ti in task collection T
2 for j=1, 2, …, n
3 initialize RT(j)=0
4 calculate prediction finish time of task Ti which is assigned to resource
  Mj, CT(i, j)=ETC(i, j)+RT(j)
5 end for
6 end for
7 while T is not null do
8 for each task Ti in task collection T
9 calculate MCT(i) and record host number host _MCT (i)
10 end for
11 choose a minimum value Ta and a maximum value Tb from the
   MCT(T)
12 assign Ta and Tb to host_MCT (Ta +Tb)
13 delete Ta and Tb from the task collection T
14 update RT(host_MCT (Ta +Tb))=MCT(Ta)+ MCT(Tb)
15 update CT matrix
16 end while
```

Fig. (1). Min-Max.

5. PERFORMANCE EVALUATION

We achieve the Min-Max, Min-Min and Max-Min with JAVA programming language. Experiments were run on a Pentium(R) Dual-Core 2.8 GHz PC-compatible machine with 4G of RAM and 320 G of disk storage running Windows XP. The simulation software is Cloudsim [12]. There are 100 resources (the computing power of each resource is generated randomly, and its' scope belongs to [100, 500]) in the cloud computing environment. In the cloud computing environment, the task numbers are set to 300, 500 and 1000 respectively (the size of task is generated randomly, and its' scope belongs to [4000, 440000]). The following four experiments are set to evaluate: (1) The total completion time. (2) The overall execution time. (3) The utilization rate of the entire system. (4) The average response time.

Table 1. The total completion time (1-1).

Task Number	Min-Min	Max-Min	Min-Max
300	17752	21937	20114
500	31791	39078	35802
1000	65524	75681	70330

5.1. The Total Completion Time

As the task numbers are set to 300, 500 and 1000 respectively. The total completion time for the three kindsof algorithms (Min-Min, Max-Min and Min-Max) are as follows.

Table **1** shows that the total completion time increases due to the increase of task number. As for the aspect of total completion time, Min-Min is the best, Max-Min the worst and Min-Max the average. The reason is as follows: Min-Min schedules small tasks to the fast resource to run each time. Therefore, the total completion time for Min-Min is the least. Difference from Min-Min, Max-Min schedules big tasks to the source each time, which leads to a higher completion time. During Min-Max, small tasks and big tasks are put together for scheduling which lead to average completion time.

5.2. The Overall Execution Time

As the task numbers are set to 300, 500 and 1000, respectively, the overall execution time for the three kinds of algorithms (Min-Min, Max-Min and Min-Max) are shown in Fig. (**2**).

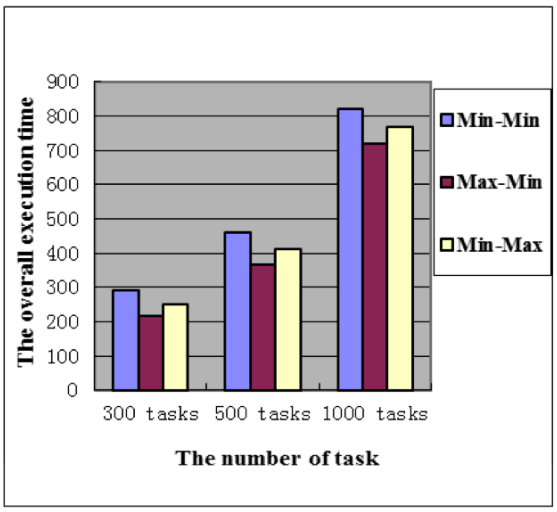

Fig. (2). The overall execution time for Min-Min, Max-Min and Min-Max.

Fig. (**2**) shows that the overall execution time increases due to the increase of task number. As for the aspect of the overall execution time, Max-Min is the best, Min-Min the worst and Min-Max the average. The reason is as follows: Min-Min schedules small tasks to fast resource to run each time. Therefore, the fast resources are full of tasks while the slow resources are idle. A lot of tasks wait for the fast resources in order to be executed and the result is that Min-Min leads to the higher execution time. Difference from Min-Min is that Max-Min schedules big tasks to resource each time, which leads to the least execution time. During Min-Max, small tasks and big tasks are put together for scheduling which leads to average execution time.

5.3. The Utilization Rate of Entire System

For the task numbers set to 300, 500 and 1000 respectively, the utilization rate of entire system for the three kinds of algorithms (Min-Min, Max-Min and Min-Max) is expressed in Fig. (**3**). Fig. (**3**) illustrates that the utilization rate of the entire system increases due to the increase of task number. As for the aspect of the utilization rate of the entire system, Min-Max is the best, Min-Min the worst and Max-Min the mediocre. The reason is as follows: Min-Min prefers scheduling small tasks which leads occupies fast resources while the slow resources are idle. The result is that Min-Min leads to imbalance of the entire system. Max-Min prefers scheduling big tasks that leads to the imbalance of the entire system. Min-Max schedules both small tasks and big tasks that leads to the balance of the entire system.

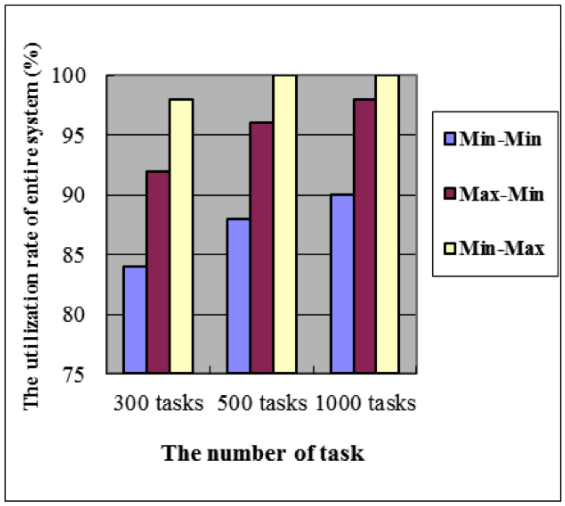

Fig. (3). The utilization rate of entire system for Min-Min, Max-Min and Min-Max.

5.4. The Average Response Time

For the task numbers set to 300, 500 and 1000 respectively, the average response time for the three kinds of algorithms (Min-Min, Max-Min and Min-Max) is as follows.

Fig. (**4**) shows that the average response time increases due to the increase of task number. As for the aspect of the average response time, Min-Min is the best, Max-Min the worst and Min-Max the middle. The reason is as follows: Min-Min prefers scheduling small tasks to be run first, which

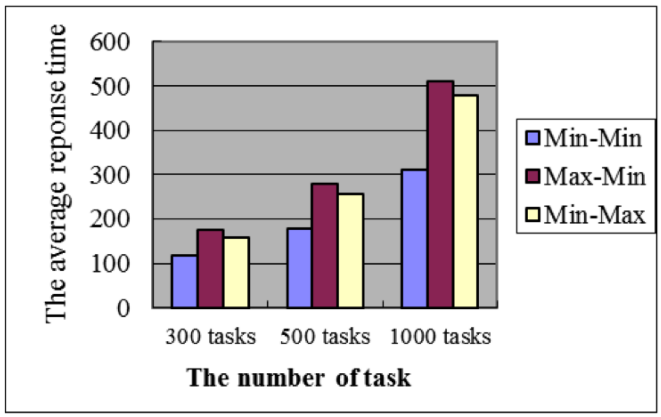

Fig. (4). The average response time for Min-Min, Max-Min and Min-Max.

leads to the shorter wait time of big tasks. Therefore, the average response time of Min-Min is the least. Contrary to Min-Min, Max-Min prefers scheduling big tasks which leads to the longer wait time of small tasks. Min-Max schedules both small tasks and big tasks which lead to the average response time.

Experimental results show that the Min-Max improves the utilization rate of the entire system and saves up to 9% overall execution time compared with Min-Min. As compared with Max-Min, Min-Max improves the utilization rate of the entire system and the total completion time and average response time are reduced by 7% and 9% respectively.

6. SUMMARY AND PROSPECT

In this paper, we propose Min-Max algorithm and verify its validity. In cloud computing environment, we should consider various factors such as the price of resources and the energy consumption of resources except the total completion time, the overall execution time, the utilization rate of the entire system and the average response time. In the future work, we will do more researches on this aspect.

CONFLICT OF INTEREST

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

ACKNOWLEDGEMENTS

This work was supported by a Grant from the National Natural Science Foundation (Nos. 61272148, 60970038), the Ph.D. Programs Foundation of Ministry of Education of China (Nos. 20120162110061), the Fundamental Research Funds for the Central Universities of Central South University (Nos. 2014zzts044) and the Hunan Provincial Innovation Foundation For Postgraduate (Nos. CX2014B066). Also, the authors greatly thank the reviewers' valuable comments on this paper.

REFERENCES

[1] M. Sedaghat, F. Hernández, and E. Elmroth, "Unifying cloud management: towards overall governance of business level objectives", In: *11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid),* Beach: California, 2011, pp. 591-597.

[2] T. V. T. Duy, Y. Sato, and Y. Inoguchi, "A prediction-based green scheduler for datacenters in clouds," *IEICE Transactions on Information and Systems,* vol. 94, pp. 1731-1741, Sept. 2011.

[3] A. A. Ahmadi, A. Olshevsky, P. A. Parrilo, and J. N. Tsitsiklis, "NP-hardness of deciding convexity of quartic polynomials and related problems," *Mathematical Programming,* vol. 137, pp. 453-476, Nov. 2013.

[4] C. S. Calude, E. Calude, and M. S. Queen, "Inductive complexity of the P versus NP problem," *Parallel Processing Letters,* Vol. 23, pp. 2-9, Mar. 2013.

[5] H. Casanova, A. Legrand, D. Zagorodnov, and F. Berman, "Heuristics for scheduling parameter sweep applications in grid environments", In: *9th Heterogeneous Computing Workshop,* Cancun: Mexico, 2000, pp. 349-363.

[6] K. Etminani, and M. Naghibzadeh, "A min-min max-min selective algorithm for grid task scheduling," In: *3rd IEEE/IFIP International Conference in Central Asia on Internet*, Tashkent, 2007, pp. 1-7.

[7] R. Moreno, and A. B. Alonso-Conde, "Job scheduling and resource management techniques in economic grid environments," In: *Grid Computing, Santiago de Compostela*, Spain, 2004, pp. 25-32.

[8] O. Engin, G. Ceran, and M. K. Yilmaz, "An efficient genetic algorithm for hybrid flow shop scheduling with multiprocessor task problems," *Applied Soft Computing*, vol. 11, pp. 3056-3065, April 2011.

[9] O. L. Sathappan, P. Chitra, P. Venkatesh, and M. Prabhu, "Modified genetic algorithm for multiobjective task scheduling on heterogeneous computing system," *International Journal of Information Technology, Communications and Convergence*, vol. 1, pp. 146-158, March 2011.

[10] T. Kokilavani, and D. I. G. Amalarethinam, "Load balanced min-min algorithm for static meta-task scheduling in grid computing", *International Journal of Computer Applications,* vol. 20, pp. 43-49, Apr. 2011.

[11] A. Agarwal, and P. Kumar, "Multidimensional Qos oriented task scheduling in grid environments," *International Journal of Grid Computing and Applications (IJGCA),* vol. 2, pp. 28-37, Mar. 2011.

[12] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, and R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and Experience,* vol. 41, pp. 23-50, August 2011.

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