

# Tasks Scheduling and Resource Allocation for High Data Management in Scientific Cloud Computing Environment

Esma Insaf Djebbar<sup>(✉)</sup> and Ghalem Belalem

Department of Computer Science, University of Oran1,  
Ahmed Ben Bella, Oran, Algeria  
esma.djebbar@gmail.com,  
ghalemldz@yahoo.fr

**Abstract.** Cloud computing refers to the use of computing, platform, software, as a service. It's a form of utility computing where the customer need not own the necessary infrastructure and pay for only what they use. Computing resources are delivered as virtual machines. In such a scenario, data management in virtual machines in Cloud Computing is a new challenge and task scheduling algorithms play an important role where the aim is to schedule the tasks effectively so as to reduce the turnaround time and improve resource utilization and Data Management.

In this work, we propose two strategies for task scheduling and resource allocation for high data in Cloud computing. The main objective is to improve data management in virtual machine in Cloud computing and optimize the total execution time of all tasks.

**Keywords:** Task scheduling · Resource allocation · High data management · Scientific cloud computing

## 1 Introduction

Cloud computing can be best described as a highly automated, readily scalable, on-demand computing platform of virtually unlimited processing, storage and ubiquitous connectivity, always available to carry out a task of any size and charged based on usage. While early in its evolution, Cloud computing is fast becoming as pervasive a platform as the internet [11].

Cloud computing is the large-scale Data center resources which are more concentrated [1]. In addition, virtualization technology hides the heterogeneity of the resources in Cloud computing [2], Cloud computing is user-oriented design which provides varied services to meet the needs of different users. It is more commercialized, and the resources in Cloud computing are packed into virtual resources by using virtualization technology [3]. This causes its resource allocation process, the interaction with user tasks and so on are different with grid computation.

Cloud computing adoption continues to gain momentum across a broad range of industries including financial services. Once organizations manage to filter through the

noise surrounding the cloud, there are actually some very pragmatic ways in which the IT organizations of banks, insurers and similar institutions can leverage Cloud computing to directly benefit their daily operations and most significantly, have an impact on the business bottom line. These gains can be achieved without incurring large capital expenditure or exposing sensitive business data.

Cloud task scheduling and virtual machine (VM) resource allocation optimization is an important, challenging and core component in the cloud application services and Cloud computing systems. Scheduling refers to the appropriate assignment of tasks to the resources available like CPU, memory and storage, such that there is a maximum utilization of resources. Efficient scheduling is a necessary for both cloud service requesters as well as providers.

In this work, we propose two strategies for tasks scheduling and resource allocation. The first strategy for tasks scheduling and resources allocation is based on a deadline, length of cloudlets and the speed of execution of the virtual machine. The second strategy of task scheduling and resource allocation uses tree based data structure called Virtual Machine Tree (VMT) for efficient execution of tasks.

The remaining parts of this paper are organized as follows. The next section briefly describes the generality of tasks scheduling in a Cloud computing environment. In Sect. 3, the related works are presented for a task scheduling and resource allocation in a Cloud computing environment. A proposed model is illustrated in Sect. 4 and experimentation and results is given in Sect. 5. Section 6 concludes the paper and discusses future research directions.

## 2 Problem of Task Scheduling

Task scheduling algorithm is a method by which tasks are matched, or allocated to Data center resources. Due to conflicting scheduling objectives generally no absolutely perfect scheduling algorithm exists. A good scheduler implements a suitable compromise, or applies combination of scheduling algorithms according to different applications [12]. A problem can be solved in seconds, hours or even years depending on the algorithm applied. The efficiency of an algorithm is evaluated by the amount of time necessary to execute it. The execution time of an algorithm is stated as a time complexity function relating the input. There are several kinds of time complexity algorithms that appear in the literature. If a problem has a polynomial time algorithm, the problem is tractable, feasible, efficient or fast enough to be executed on a computational machine. In computational complexity theory, set of problems can be treated as complexity class based on a certain resource.

Class NP is the set of decision problems that are solvable on a nondeterministic Turing machine in polynomial time, but a candidate solution of the problem of Class NP can be confirmed by a polynomial time algorithm, which means that the problem can be verified quickly.

Class NP-complete is the set of decision problems, to which all other NP problems can be polynomial transformable, and a NP-complete problem must be in class NP. Generally speaking, NP-complete problems are more difficult than NP problems.

Class NP-hard is the set of optimization problems, to which all NP problems can be polynomial transformable, but a NP-hard problem is not necessarily in class NP.

Task scheduling problem is the problem of matching tasks to different sets of resources which is formally expressed as a triple  $(T, S, O)$  where 'T' is the set of tasks, each of which is an instance of problem, the set of feasible solutions is 'S' and the objective of the problem is 'O'.

Scheduling problem can be further classified into two types as optimization problem and decision problem based on objective O. An optimization problem requires finding the best solution among all the feasible solutions in set S. Different from optimization; the aim of decision problem is relatively easy. For a specified feasible solution  $s \in S$ , problem needs a positive or negative answer to whether the objective is achieved. Clearly, optimization problem is harder than decision problem.

Scheduling theory for Cloud computing is receiving growing attention with increase in cloud popularity. In general, scheduling is the process of mapping tasks to available resources on the basis of tasks' characteristics and requirements. It is an important aspect in efficient working of cloud as various task parameters need to be taken into account for appropriate scheduling. The available resources should be utilized efficiently without affecting the service parameters of cloud. Scheduling process in cloud can be generalized into three stages namely:

- a. **Resource discovering and filtering:** Datacenter Broker discovers the resources present in the network system and collects status information related to them.
- b. **Resource selection:** Target resource is selected based on certain parameters of task and resource.
- c. **Task submission:** Task is submitted to resource selected. This is deciding stage.

### 3 Related Works

The difficulty of task scheduling in distributed computing system is to handle dependent or independent tasks is a well studied area. In this section, we explained different existing task scheduling methods in a heterogeneous computing environment. By using dynamic allocation methods applied on large sets of real-world applications that are able to be formulated in a way which allows for deterministic execution. But the dynamic technique doesn't have any prior knowledge about tasks to be executing compare to static techniques.

The task scheduling algorithms currently prevalent in clouds are summarized in Table 1. All the algorithms are implemented in cloud computing environment.

Target resources in a cloud environment can be selected in various ways. The selection of resources can be either random, round robin, greedy (resource processing power and waiting time based) or by any other means. The selection of jobs to be scheduled can be based on FCFS, SJF, priority based, coarse grained task grouping etc. Scheduling algorithm selects job to be executed and the corresponding resource where

**Table 1.** The related works in the literature

| N | Scheduling algorithms  | Scheduling parameters                                | Skeleton outline  | Tools                  |
|---|--|--|---|------------------------|
| 1 | Cloud task and virtual machine allocation strategy [4]   | The total execution time of all tasks                | Use Speed of VM, cloudlets length and load balancing is taken in to account   | Cloudsim               |
| 2 | Optimal scheduling of computational task [5]   | The execution time                                   | Use Tree based data structure called Virtual Machine Tree (VMT) for efficient execution of tasks  | Cloudsim               |
| 3 | A deadline scheduler for jobs [6]  | The missed deadlines, execution time and cost        | Cloud Least Laxity First (CLLF), minimizes the extra-cost implied from tasks that are executed over a cloud setting by ordering each of which using its laxity and locality | Cloudsim               |
| 4 | Job scheduling algorithm based on Berger [7]   | The execution time, user satisfaction and CPU number | The algorithm establishes dual fairness constraint  | Cloudsim               |
| 5 | Online optimization for scheduling preemptable tasks on IaaS cloud systems [8]   | The computational power, execution time, bandwidth   | Algorithms adjust the resource allocation dynamically based on the updated information of the actual task executions  | Simulation environment |
| 6 | A priority based job scheduling algorithm [9]  | The complexity, consistency, makespan                | The proposed algorithm is based on multiple criteria decision making model  | Cloud environment      |
| 7 | Performance and cost evaluation of Gang Scheduling in a Cloud Computing system with job migrations and starvation handling | The response time and Bounded Slowdown               | The study takes into consideration both performance and cost while integrating mechanisms for job migration and handling of job starvation                                  | Cloud environment      |

the job will be executed. As each selection strategy is having certain flaws work could be done in this direction to extract the advantageous points of these algorithms and come up with a better solution that tries to minimize the drawbacks of resultant algorithm.

New scheduling strategy need to be proposed to overcome the problem posed by network properties and user requirements. The new strategies may use some of the conventional scheduling concepts to merge them with some network and requirement aware strategies to provide solution for better and more efficient task scheduling.

### 4 Proposed Strategy of Task Scheduling

In this section, two strategies of task scheduling and resources allocation are presented:

#### 4.1 The First Strategy

We propose a strategy for tasks scheduling and resources allocation based on a deadline, length of cloudlets and the speed of execution of the virtual machine. Our proposition are different from [9] because we add in the algorithm, in the second step,

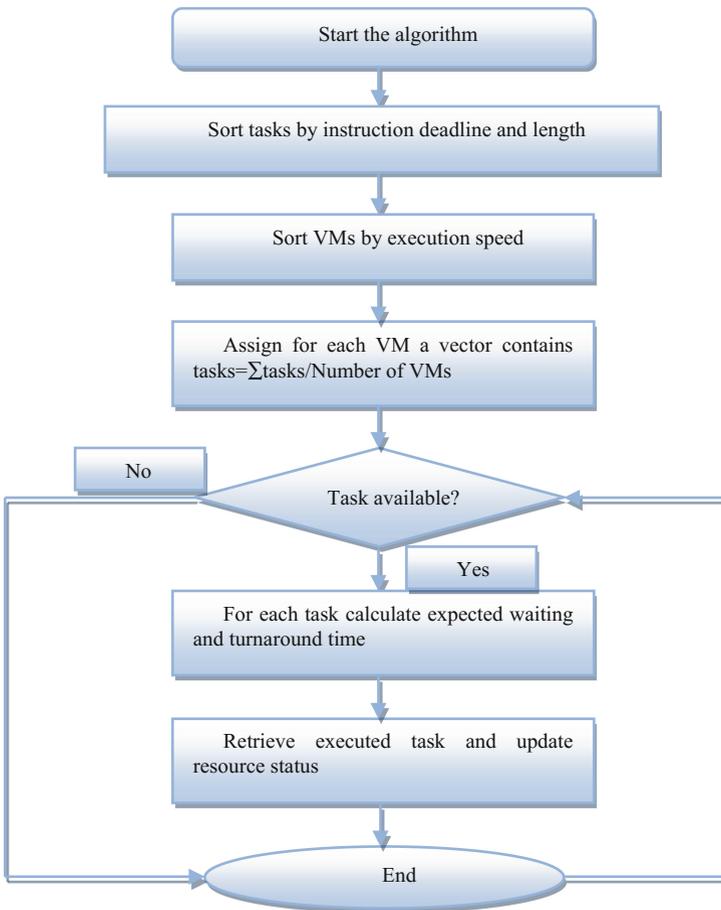


Fig. 1. First proposed strategy

a division of number of cloudlets by the number of virtual machines to minimize an average time execution of all tasks. The major process lines of strategy are as follows, and the flow chart is shown in Fig. 1.

The different steps of the algorithm are as follows:

- Step 1: sort the cloud task by instruction deadline and length in ascending order;
- Step 2: sort the virtual machine by execution speed in ascending order;
- Step 3: Assign for each VM a vector, a number of cases equals to M (number of cloudlets) divide by N (number of VM); so that the first group of the first tasks are executed by the first VM, the second one are executed by the second VM,...

### 4.2 The Second Strategy

The second strategy of task scheduling and resource allocation uses tree based data structure called Virtual Machine Tree (VMT) for efficient execution of tasks. Our algorithm is an amelioration of [10] but it provides a better load balancing.

A Virtual Machine Tree (VMT) is a binary tree with N nodes. Each node represents a Virtual Machine containing Virtual Machine Id and MIPS. N represents total number of computational specific Virtual Machines in a cloud. The special property of VMT is that node value (MIPS) at level L is greater than or equal to node value at level L + 1 where  $L \geq 0$ . Each node contains zero, one or two child nodes. A node with no child node is called as a leaf node and the node with child nodes is referred as internal nodes. Consider a 5 computational specific Virtual Machines represented by their Id and MIPS as  $V = \{ \{0, 250\}, \{1, 1000\}, \{2, 250\}, \{3, 500\}, \{4, 250\} \}$ . Figure below shows the VMT. The VMT is constructed based on the prioritized order of Virtual Machines from left to right, such that Virtual Machine with highest MIPS becomes the root (Fig. 2).

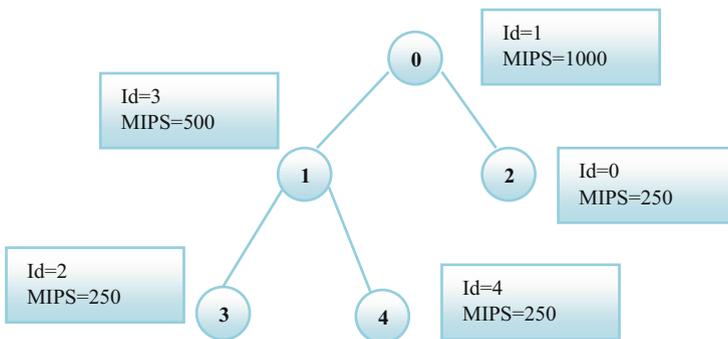


Fig. 2. An example of a Virtual Machine Tree (VMT)

Here VMT with the root node representing the Virtual Machine with Id 1 and MIPS 1000. The root node has two children. The left child node represents the Virtual Machine with Id 3 and MIPS 500. The right child node represents the Virtual Machine with Id 0 and MIPS 250. Similarly node which represents the Virtual Machine with Id 3 and MIPS 500 has 2 children. The left child of this node represents the Virtual Machines with Id 2 and MIPS 250, right child represents the Virtual Machine with Id 4 and MIPS 250 respectively.

Here we present a grouping mechanism for the set of task submitted to the cloud. Let T COUNT be the total number of tasks submitted and L COUNT be the total number of leaf nodes in VMT. The total number of groups G COUNT for the submitted tasks are computed as follows: G COUNT = L COUNT.

If VMT constructed with 5 Virtual Machines, then total number of group is the number of level and it's equal to 3 in our example. The number of tasks in each group G is computed as follows. G = Number of level.

Each group contains the maximum number of tasks that their sum does not exceed a value which is calculated by the following formula, and each group is assigned in this level, the first level in the top level (root), the second group in the second level, and the last one in the third level.

$$\sum \text{length of tasks} * \text{VM Mips in level} / \sum \text{Mips of VMs}$$

Consider 12 tasks represented by their Id and size as  $G = \{\{0, 20000\}, \{1, 20000\}, \{2, 20000\}, \{3, 10000\}, \{4, 10000\}, \{5, 20000\}, \{6, 10000\}, \{7, 20000\}, \{8, 10000\}, \{9, 10000\}, \{10, 20000\}, \{11, 10000\}\}$ .

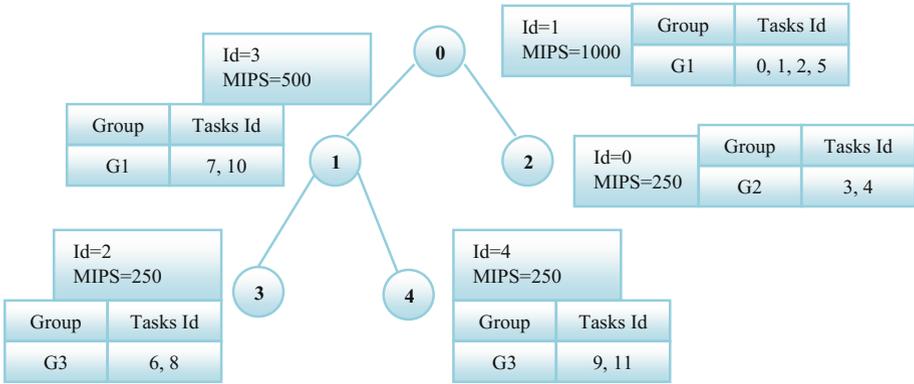
After prioritizing and grouping, each group contains following tasks.

$$G1 = \{\{0, 20000\}, \{1, 20000\}, \{2, 20000\}, \{5, 20000\}\}$$

$$G2 = \{\{7, 20000\}, \{10, 20000\}, \{3, 10000\}, \{4, 10000\}\}$$

$$G3 = \{\{6, 10000\}, \{8, 10000\}, \{9, 10000\}, \{11, 10000\}\}$$

Once the grouping of the tasks are done, suitable Virtual Machines are selected for execution from VMT. The tasks in each group is selected sequentially and submitted to the Virtual Machine. The order is as follows. The first task in the group G1 is executed by the Virtual Machine represented by the root node of the VMT. The second task will be executed by its child, third task will be executed by grand child and so on. Once it reaches the Virtual Machine represented by the leaf node, the next task will be submitted once again to root node and so on. Same procedure is repeated for all the tasks in each group. Figure below shows the VMT for 5 Virtual Machines and total number of groups formed for the 12 submitted tasks (Fig. 3).



**Fig. 3.** The result of Tasks execution in the VMT tree

Here total number of tasks submitted will be in 3 groups namely G1, G2 and G3 respectively. Tasks with Id 0, 2, 5, 7 will be in group G1, tasks with Id 10, 1, 3, 4 will be in group G2 and tasks with Id 6, 8, 9, 11 will be in group G3 respectively.

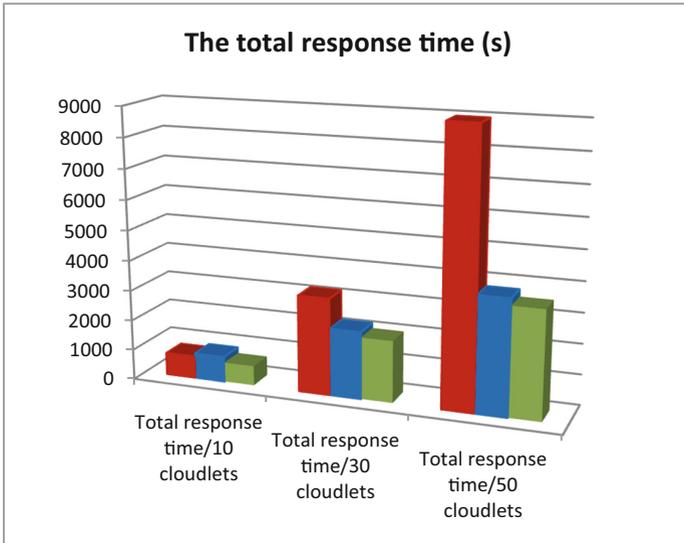
## 5 Experimentation Results

### 5.1 The Response Time for the First Strategy

The experiments are conducted on a simulated Cloud environment provided by CloudSim. The speed of each processing element is expressed in MIPS (Million Instructions Per Second) and the length of each cloudlet is expressed as the number of instructions to be executed. The simulation environment consists of two Data Center with two hosts having two Processing Elements respectively. Each Processing Element is assigned varying computing power (varying MIPS). The algorithms are tested by varying the number of cloudlets from 10 to 50 and also varying the length of cloudlets. Also, the number of VMs used to execute the cloudlets, are varied accordingly. The overall response time to execute the cloudlets is used as the metric to evaluate the performance of the first strategy. The results are shown in Table 2 and Fig. 4.

**Table 2.** The total response time results of execution tasks

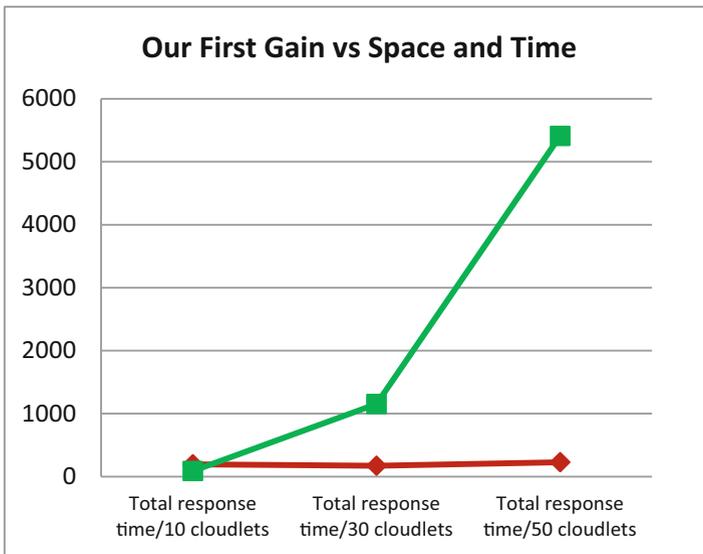
|                              | Time shared | Space shared | Proposed first strategy |
|------------------------------|-------------|--------------|-------------------------|
| Response time 10 Cloudlets/s | 734,92      | 840,75       | 646,63                  |
| Response time 30 Cloudlets/s | 3185,15     | 2204,34      | 2032,39                 |
| Response time 50 Cloudlets/s | 8959        | 3776,8       | 3548,35                 |



**Fig. 4.** The total response time results of execution of tasks.

The following Fig. 5 show the gain obtained:

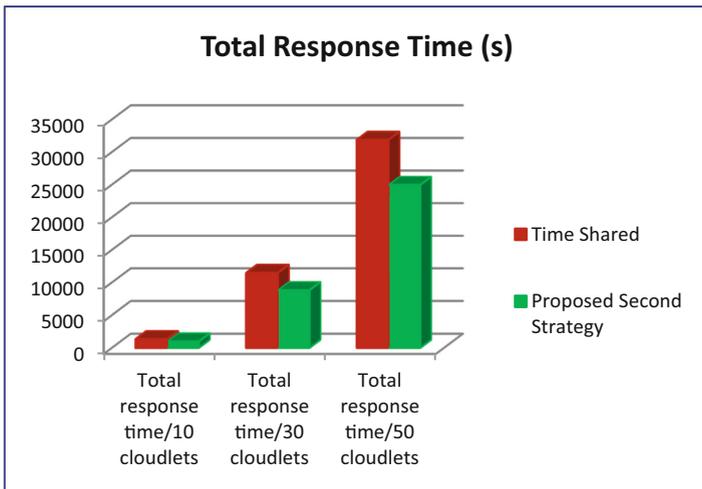
It has been observed that, for smaller number of tasks, all the three algorithms exhibit more or less similar performance since the length of the queued cloudlets is less. But as shown in Table 2 and Fig. 4, as the number of tasks increase, the first strategy exhibits better performance when compared to Space shared and Time shared since longer tasks complete faster thereby reducing the response time.



**Fig. 5.** The gain of the first strategy obtained vs. Space and Time.

### 5.2 The Response Time for the Second Strategy

For the second strategy, the experiments are conducted on a simulated Cloud environment provided by CloudSim. The speed of each processing element is expressed in MIPS (Million Instructions Per Second) and the length of each cloudlet is expressed as the number of instructions to be executed. The simulation environment consists of two Data Center with two hosts having two Processing Elements respectively. Each Processing Element is assigned varying computing power (varying MIPS). The algorithms are tested by varying the number of cloudlets from 10 to 50 and also varying the length of cloudlets. Also, the number of VMs used to execute the cloudlets, are varied accordingly. The overall response time to execute the cloudlets is used as the metric to evaluate the performance of the first strategy. The results are shown in Table 3 and Fig. 6.



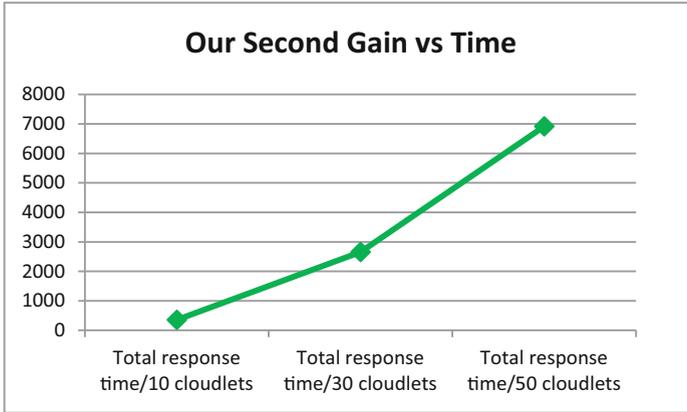
**Fig. 6.** The total response time results of execution of tasks.

The following Fig. 7 show the gain obtained:

It has been observed that, for smaller number of tasks, all the three algorithms exhibit more or less similar performance since the length of the queued cloudlets is less. But as shown in Table 3 and Fig. 6, as the number of tasks increase, the second strategy exhibits better performance when compared to Time shared since longer tasks complete faster thereby reducing the response time.

**Table 3.** The total response time results of execution tasks.

|                              | Time shared | Proposed second strategy |
|------------------------------|-------------|--------------------------|
| Response time 10 Cloudlets/s | 1334,94     | 980                      |
| Response time 30 Cloudlets/s | 11475,05    | 8819,98                  |
| Response time 50 Cloudlets/s | 31875,7     | 24959,88                 |



**Fig. 7.** The gain of the second strategy obtained vs. Time.

The two strategies can provide a better response time, a minimizing waiting time of tasks and better load balancing.

## 6 Conclusion and Perspectives

In this paper, we proposed two effective scheduling strategies and allocation resources in the environment of computing clouds. The first strategy is based on the length of tasks and the speed of execution of Virtual Machines, the second is based on VMT (Virtual Machine Tree).

The two strategies are tested in the CloudSim simulator and compared with Space Shared and Time Shared politics. The strategies can provide a better response time, a minimizing waiting time of tasks and better load balancing.

For a continuation of our work, we think to propose a third strategy of task scheduling and resource allocation takes into account the pre-emption of tasks to ensure better energy consumption and in the second way, The future work may group the cost based tasks before resource allocation according to resource capacity to reduce the communication overhead.

## References

1. Peng, L.: The definition of cloud computing and characteristics. <http://www.chinacloud.cn/2009-2-25>
2. Sutherland, I.E.: A future market in computer time. *Commun. ACM* **11**(6), 449–451 (1968)
3. Ferguson, D., Yemini, Y., Nikolaou, C.: Microeconomic for load balancing in distributed computer Systems. In: *Proceeding of the Eighth International Conference on Distributed Systems*. San Jose, pp. 491–499. IEEE Press (1988)

4. Xu, X., Hu, H., Hu, N., Ying, W.: Cloud task and virtual machine allocation strategy in cloud computing environment. In: Lei, J., Wang, F.L., Li, M., Luo, Y. (eds.) NCIS 2012. CCIS, vol. 345, pp. 113–120. Springer, Heidelberg (2012)
5. Achar, R., Thilagam, P.S., Shwetha, D., et al.: Optimal scheduling of computational task in cloud using virtual machine tree. In: 2012 Third International Conference on Emerging Applications of Information Technology (EAIT), pp. 143–146 (2012)
6. Perret, Q., Charlemagne, G., Sotiriadis, S., Bessis, N.: A deadline scheduler for jobs in distributed systems. In: 2013 27th International Conference on Advanced Information Networking and Applications Workshops (WAINA), pp. 757–764 (2013)
7. Baomin, X., Zhao, C., Enzhao, H., Bin, H.: Job scheduling algorithm based on Berger model in cloud environment. *Adv. Eng. Softw.* **42**, 419–425 (2011)
8. Li, J., Qiu, M., Ming, Z., Quan, G., Qin, X., Gu, Z.: Online optimization for scheduling preemptable tasks on IaaS cloud systems. *J. Parallel Distrib. Comput.* **72**(5), 666–677 (2012)
9. Ghanbaria, S., Othmana, M.: A priority based job scheduling algorithm in cloud computing. In: ICASCE 2012, pp. 778–785 (2012)
10. Moschakis, I.A., Karatza, H.D.: Performance and cost evaluation of Gang Scheduling in a Cloud Computing system with job migrations and starvation handling. In: IEEE Symposium on Computers and Communications (ISCC) (2012) and 2011 IEEE Symposium on Computers and Communications, pp. 418–423 (2011)
11. Sharma, A.: Data management and deployment of cloud applications in financial institutions and its adoption challenges. *Int. J. Sci. Technology Res.* **1**(1), 1–7 (2012)
12. Djebbar, E.L., Belalem, G.: Optimization of tasks scheduling by an efficacy data placement and replication in cloud computing. In: Aversa, R., Kołodziej, J., Zhang, J., Amato, F., Fortino, G. (eds.) ICA3PP 2013, Part II. LNCS, vol. 8286, pp. 22–29. Springer, Heidelberg (2013). doi:[10.1007/978-3-319-03889-6\\_3](https://doi.org/10.1007/978-3-319-03889-6_3)



<http://www.springer.com/978-3-319-50462-9>

Mobile, Secure, and Programmable Networking  
Second International Conference, MSPN 2016, Paris,  
France, June 1-3, 2016, Revised Selected Papers  
Boumerdassi, S.; Renault, E.; Bouzefrane, S. (Eds.)  
2016, X, 225 p. 100 illus., Softcover  
ISBN: 978-3-319-50462-9