An Improved SJF Scheduling Algorithm in Cloud Computing Environment

Mokhtar A. Alworafi
DoS in CS
Mysore University, India
mokhtar119@gmail.com

Atyaf Dhari
DoCS
Thi_Qar University, Iraq
atyafcomsinc@gmail.com

Asma A. Al-Hashmi
DoS in CS
Mysore University, India
asma.alhashmi@yahoo.com

A. Basit Darem
DoS in CS
Mysore University, India
Basit.darem@yahoo.com

Suresha
DoS in CS
Mysore University, India
sureshabm@yahoo.co.in

Abstract— **Today, the service provider has to serve many users and the increase of the requests or tasks from the users to the cloud providers has become one of the scalable techniques to proposed the services. Many scheduling algorithms have been proposed to schedule the tasks in cloud computing environment such as (SJF) and (FCFS) algorithms. This paper aims to improve the shortest job first scheduling algorithm in the cloud computing. In tasks scheduling (TS), the most important parameters are makespan and response time. Therefore, we have proposed a Modified Shortest Job First algorithm (MSJF) to minimize the completion time of the last task (Makespan) and minimize the average response time with maximizing the resources utilization. MSJF has two functions, one is calculating the average of tasks length and the other one is Load-Balancing between the virtual machines. One of the important advantages of MSJF is sending the longest tasks to the fastest Machine. We compared the result of our proposed algorithm MSJF with SJF and FCFS. The performance of MSJF is better compared to SJF and FCFS.**

Keywords—Cloud computing; Shortest Job First Scheduling; Load Balancing; SJF; Response time; Makespan.

I. INTRODUCTION

The cloud environment has a number of heterogeneous resources hosted in data centers in different locations. These data centers contain a large number of physical machines, which contain Virtual Machines (VM). Each VM has a specific configuration of processing power, RAM, communication bandwidth and storage associated with it. So customers do not require purchasing several hardware and software [1]. In cloud computing environment, the service provider provides three services - infrastructure as a service (IAAS), platform as a service (PAAS) and software as a service (SAAS) to the users on demand through the internet [19].

Cloud computing (CC) has two components: Provider and User. The user submits the task(s) to providerfor execution. The Provider receives the users' tasks, executes them at a specific data center, then sends the results back to the user [3].
Load Balancing is an important issue in cloud computing. The cloud platform has the ability to scale up and down any time. The nature of this dynamic environment required an efficient load balancing and task scheduling to reduce the time of response, completion time (makespan), energy consumption and the services interruption. It provides also high availability when the components are in non-responsive mode [4]. The balance scheduling algorithms were proposed to schedule the load between the resources in cloud computing system for the achievement and enhancement of the optimized results as a whole [5].
Moreover, tasks scheduling is a key issue in CC [6]. Scheduling can be defined as a set of policies that manage the arrangement of processing tasks that is represented in cloud computing systems [7]. Each task has an execution time based on the capability of the VM to which it is allocated. The tasks involved in cloud computing could be in large scales because of the huge number of users. The scheduling algorithm assigns the tasks to the suitable and available resources in the cloud environment without violating the precedence constraints. In other words, tasks scheduling's considered as one of the important components of cloud environment, as it maps the users' tasks to appropriate resources utilization, if scheduling techniques are working efficiently, they will affect the performance of the whole CC environment. One of the basic objectives of the scheduler is to utilize the use of CPU to facilitate the processes of the completion and response time, waiting and turnaround time, priorities and the system throughput [8]. One of the proposed tasks scheduling algorithms is SJF. The task in this algorithm has the lowest execution time, which its process having the smallest CPU burst gets a high priority and executes at the first by the CPU. The remaining tasks wait for the execution during this time, and finally the task that has the highest execution time will be executed at the end. The criteria of Shortest Job First (SJF) scheduling algorithm is that it process the smallest task first. In our proposed algorithm, if the task has the highest execution time, the broker will send the task to high configuration virtual machine to execute this task, and if the task has the lowest execution time, the broker will send it to the low configuration virtual machine to be executed.

This paper is divided into many sections. The motivation of enhanced and modified task scheduling algorithm is illustrated in the second section. Section 3 presents, the related work about task scheduling algorithm in CC environment. The architecture model of the proposed algorithm is illustrated in section 4.The result of the experiments is discussed in section 5. The performance evaluation is shown in section 6. Finally, in section 7, conclusion is provided.

II. Motivation

The scheduler is considered as one of the most critical service components of the cloud middleware that can be integrated to achieve a real cloud environment. The responsibility of task scheduling is selecting the best suitable VMs or computing resources with the consideration of maximizing resource utilization. We aim to reduce the total completion time (makespan) and the response time with maximizing the resources utilization. Load balancing and task scheduling are compatible together to achieve satisfaction for users and service providers. We attempt to manage the loads between the virtual machines and use the best algorithm to execute the tasks with taking into consideration maximizing the resource utilization and reducing the completion time.

III. RELATED WORK

Rashmi et al. proposed an efficient technique for load balancing for reducing the Response Time (RT) to execute the task that comes from different customers of cloudenvironment. This technique of SJFST provides an effective balancing technique for minimizing the RT to process the job requests received from different customers of the cloud [9]. JiaRu and Jacky Keung proposed a scheduling algorithm with group of tasks, aware of priority and SJF to reduce the makespan and waiting time, with maximizing resource utilization. They compared the existing scheduling algorithms with grouping scheduling algorithm. The results of the experiment showed that the waiting time and processing time in the proposed technique decreased significantly (over 30%) [13].
E. Iniya Nehru et al. proposed a priority-based algorithm, they checked each task according to different algorithms and calculated their waiting time. The waiting time average compared to First Come First Serve scheduling algorithm is lower. It means that Shortest Job First Scheduling algorithm is more efficient than the First Come First Serve algorithm [10].
Ranjan Kumar Mondal et al. proposed an efficient scheduling algorithm for load balancing, providing an improved techniques among efficient task scheduling to reduce the RT and increase the availability of virtual machines to assign new tasks from requesting users [11]. Mohammed Aboalama and Adil Yousif proposed a new task scheduling technique in cloud system using Shortest Remaining Job First algorithm. They evaluated using cloud simulator with a number of tasks varying from 5 to 25 and extended either VM-Scheduler or Cloudlet-Scheduler, then created the techniques for deciding sharing of processing element [12]. Thomas Yeboah et al. enhanced the version of Round Robin (RR) algorithm is according to the reintegrating of round-robin with SJF algorithm that are selected for processing according to shortest job first in RR fashion then select the optimal job. This Scheduling algorithm gives a better result compared to RR [2].

IV. Architecture Model of Proposed Algorithm

The Architecture model of the proposed algorithm contains three main sections: first section represents user side (tasks), second section contains the broker (balancing scheduling) and the third section represents the service provider. The main components are illustrated in Fig.1. Data center helps to manage several hosts in a CC system. Host is one of the important components which is a physical computing server in a Cloud. We can create any number of hosts in a data center based on datacenter capacity. Each host can has its own configuration and resources like CPU, bandwidth, RAM (in MB), and storage. The other component is Virtual Machine. It is possible to create any number of virtual machines on host with different computational capabilities according to host capacity. Each virtual machine has its own configuration and resources also [4]. Cloud provider is responsible for delivering computing and services to the users and managing the scheduling and balancing with the cloud broker based on the resources available. Cloud Broker is another component which is working as meditatorbetween users and providers. Task is the user request. It is an independent and computational one. Each task has its own configuration and resources like task length, id, and number of processing element (CPU) required.

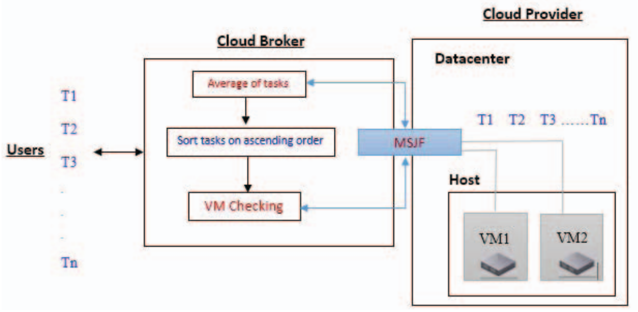

Fig. 1. Proposed TS algorithm in CC.

This paper aims to improve the shortest job first scheduling algorithm in CC. In order to get the best result of task scheduling to take less execution time and reduce the makespan. The main idea of the Modified Shortest Job First is to sort the tasks in an ascending order based on the task length and calculate the average of all the tasks length. Then for all tasks the algorithm checks each task length, if it is less than the average of tasks length and number of tasks in VM1 less than the number of tasks in VM2, then the task will be sent to VM1, else to VM2.

The following Parameters are used in *MSJF:*

Average of tasks length (AV) can be calculated using Eq. (1).

$$\textbf{AV}=\sum_{i=1}^{n} LT_i / n \qquad (1)$$

Where n is the number of tasks, ***LT*** is the length of the task.

Execution Time (ExT) is the exact time taken to execute the given tasks on VM as defined in the Eq. (2).

$$ExT = LT / VMP \qquad (2)$$

Where VMP is the power processing of ***VM***

CompletionTime (CT) isthe sum of the **ExT** of all the previous tasks and **ExT$_m$** of the present task allocated in the same VM as mentioned in the Eq. (3).

$$CT = ExT_m + \sum_{i=1}^{m-1} ExT \qquad (3)$$

Makespan is the total amount of time required to complete a group of tasks which indicates the completion time of last task. The scheduling task is to minimize the makespan as most of the user's desire fastest execution of their applications [14] it is illustrated in the Eq. (4).

$$Makespan = CT_n \qquad (4)$$

Where CT$_n$is the completion time of last task.

Response Time (RT)represents the amount of time required to respond by the load balancing algorithms in cloud computing and this metric must be reduced [15]. As mentioned in the Eq. (5).

$$\textbf{RT}= \sum_{i=1}^{n} CT_i + SB_i \qquad (5)$$

CT: completion time of task, **SB**: Submission time of task

Then calculate the average of response time for each VMs using Eq. (6).

$$Avg.RT=RT/N \qquad (6)$$

N: number of tasks in the VM.

The implementation steps of MSJF algorithm are shown below and illustrated in Fig. 2.

Step1: Create a heterogeneous cloud computing environment.

Step2: Input all the required tasks and calculate the average of tasks length for all the tasks.

Step3: Sort the tasks in an ascending order.

Step4: For each task, if the task length is less than the average and number of tasks in VM1 less than number of tasks in VM2 then pass the task to VM1 (less computing power), else passed it to VM2(high computing power).

Step5: Calculate the average response time and makespan.

V. Experimental Result

Cloudsim is a framework which helps for simulation and instantiation of enormous cloud infrastructure, also consists ofdatacenters on a single physical computing node and java virtual machine [20]. We used CloudSim toolkit to simulate the communication environment and the heterogeneous resource environment. The experimentation environment of emerging cloud computing management and infrastructure services to be accomplished in cloudsim platform [16] [17] [18]. The configuration of VM1 and VM2 is shown in Table 1. The configuration of the host is shown in Table 2.The tasks are allocated in the VMs in MSJF manner with tracking the proposed strategy. Each task will be sentto the VM based on the average and number of tasks as shown in Table 3, where: Actual Execution Time (sec) = completion time – start time. We conducted three experiments for each algorithm. Experiment 1 has 20 tasks, experiment 2 has 40 tasks and experiment 3 has 60 tasks. Table 3 shows the distribution of the tasks among the virtual machines VM0 and VM1. Table 4 shows the average response time and total completion time for longest task (makespan) given in those simulations based on the proposed algorithm.

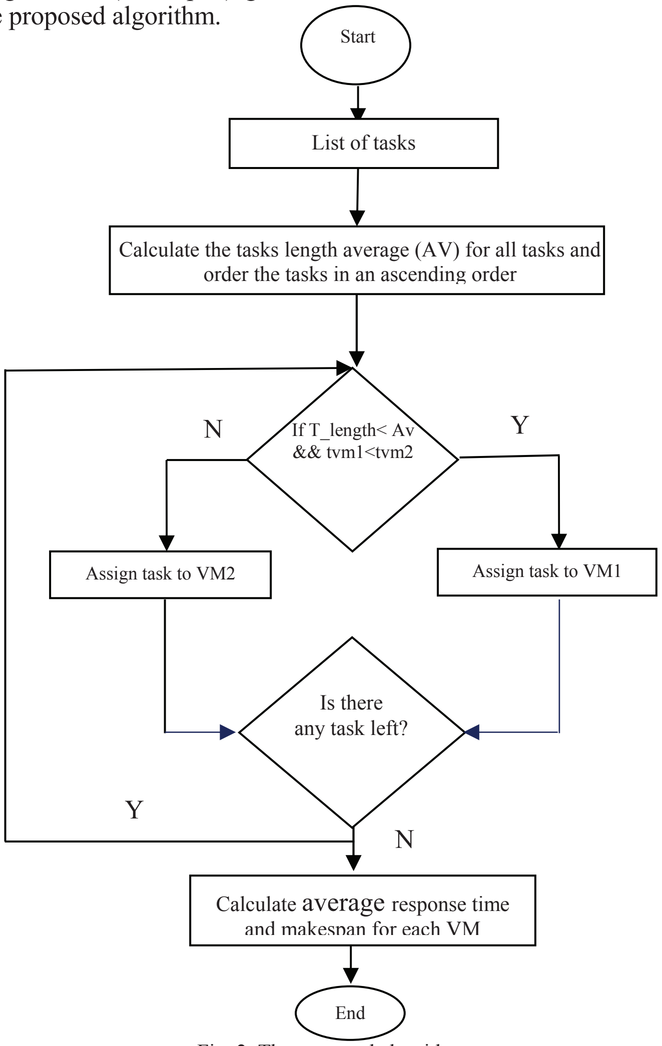
Start
List of tasks
Calculate the tasks length average (AV) for all tasks and order the tasks in an ascending order
N
If T_length< Av && tvm1<tvm2
Y
Assign task to VM2
Assign task to VM1
Is there any task left?
Y
N
Calculate average response time and makespan for each VM
End

Fig. 2. The proposed algorithm.

Table 1. Configuration of the VMs

VM No.	MIPS	Memory	Bandwidth	Storage	Scheduler Type
VM 1	1000	512 Mb.	1000 Mb.	10000 Mb.	SpaceShared
VM2	8000	512 Mb.	1000 Mb.	10000 Mb.	SpaceShared

Table2. Configuration of Host

MIPS	Memory	Bandwidth	Storage
10000	16384	10000	1000000

Table 3. Tasks allocation to VMs

No	Task ID	Start Time	Actual Execution Time	Completion Time	VM ID
1.	19	0.1	2.44	2.54	1
2.	17	2.54	4.88	7.42	1
3.	13	7.42	19.53	26.95	1
4.	11	26.95	39.06	66.02	1
5.	15	0.1	78.12	78.22	0
6.	9	66.02	78.12	144.14	1
7.	18	78.22	195.3	273.52	0
8.	7	144.14	156.25	300.39	1
9.	0	300.39	250	550.39	1
10.	16	273.52	390.6	664.12	0
11.	10	550.39	390.62	941.02	1
12.	14	664.12	781.25	1445.37	0
13.	8	941.02	781.25	1722.27	1
14.	1	1722.27	1250	2972.27	1
15.	12	1445.37	1562.5	3007.87	0
16.	6	2972.27	1562.5	4534.77	1
17.	5	3007.87	2500	5507.87	0
18.	4	4534.77	3125	7659.77	1
19.	3	5507.87	5000	10507.87	0
20.	2	7659.77	6250	13909.77	1

Table 4. Makespan and Average Response Time of MSJF

Algorithm	Average Response Time		Makespan	
	VM0	VM1	VM0	VM1
MSJF	3069	2525	10507	13909

VI. Performance Evaluation

We used two algorithms, SJF and FCFS algorithms to evaluate our proposed algorithm MSJF. For each algorithm, we used the same configuration. We repeated the experiment with different number of 20, 40 and 60 respectively. The result presented in Table 5.a, 5.b and5.c. The observation from Fig. 3, 4 and 5 shows significant improvement in performance of the proposed MSJF algorithm compared to SJF and FCFS algorithm.

Table 5.a Makespan and Average Response Time of MSJF algorithm

Task No.	MSJF			
	Average Response Time		Makespan	
	VM-0	VM-1	VM-0	VM-1
Experiment 1 (20 Tasks)	3069	2525	10507	13909
Experiment 2 (40 Tasks)	742	1571	5624	14547
Experiment 3 (60Tasks)	250	1165	3125	14860

Table 5.b Makespan and Average Response Time of SJF algorithm

Task No.	SJF			
	Average Response Time		Makespan	
	VM-0	VM-1	VM-0	VM-1
Experiment 1 (20 Tasks)	21278	497	101804	2497
Experiment 2 (40 Tasks)	10756	250	101999	2500
Experiment 3 (60Tasks)	7171	167	102000	2501

Table 5.c Makespan and Average Response Time of FCFS algorithm

Task No.	FCFS			
	Average Response Time		Makespan	
	VM-0	VM-1	VM-0	VM-1
Experiment 1 (20 Tasks)	82019	2250	101804	2497
Experiment 2 (40 Tasks)	91999	2375	101999	2500
Experiment 3 (60Tasks)	95333	2417	102000	2501

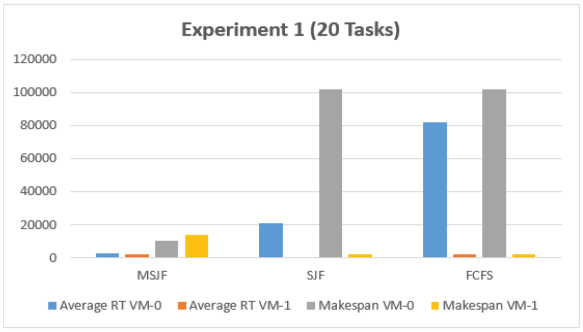

Fig. 3. Makespan and Average Response Time for 20 Tasks.

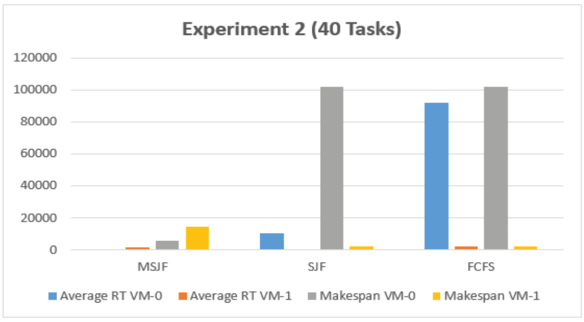

Fig. 4. Makespan and Average Response Time for 40 Tasks.

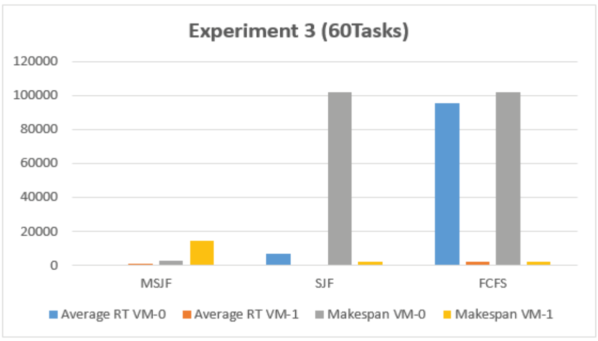

Fig.5. Makespan and Average Response Time for 60 Tasks.

From Fig. 3, 4 and 5, we can observe the performance improvement because the total task completion time for last task (makespan) and the average response time for MSJF are significantly faster than the SJF and FCFS algorithms of the existing cloud balancing.

VII. CONCLUSION AND FUTURE WORK

The work in this paper modified the SJF task scheduling algorithm to achieve tasks scheduling with least makespan. It does not only focus on the completion time of the task but also takes into account the total completion time of all tasks. We showed an empirical evidence that MSJF is better than SJF and FCFS. The empirical results showed that the MSJF is more efficient to improve the response time and makespan compared to SJF and FCFS. In future work, we will concentrate on enhancing the makespan based on QoS factors such as deadline constraint.

REFERENCES

[1] Karthick, A. V., and E. Ramaraj. "Reservation and On-Demand Priority based Queue Job Scheduling for Cloud Computing." *International Journal* 1.2 (2014).

[2] Thomas Yeboah, I Odabi and Kamal Kant Hiran, ‘An Integration of Round Robin with Shortest Job First Algorithm for Cloud Computing Environment’, *International Conference On Management, Communication and Technology,* III.1 (2015), 1–5.

[3] Aldulaimy, Auday, et al. "Job Classification in Cloud Computing: The Classification Effects on Energy Efficiency." *2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC)*. IEEE, 2015.

[4] Haidri, Raza Abbas, C. P. Katti, and P. C. Saxena. "A load balancing strategy for Cloud Computing environment." *Signal Propagation and IEEE*, 2014.

[5] Chaudhary, Divya, and Bijendra Kumar. "Analytical study of load scheduling algorithms in cloud computing." *Parallel, Distributed and Grid Computing (PDGC), 2014 International Conference on.* IEEE, 2014.

[6] Singh, Raja Manish, Sanchita Paul, and Abhishek Kumar. "Task Scheduling in Cloud Computing: Review." *International Journal of Computer Science and Information Technologies* 5.6 (2014): 7940-7944.

[7] Gahlawat, Monica, and Priyanka Sharma. "Analysis and Performance Assessment of CPU Scheduling Algorithms in Cloud using Cloud Sim."*Analysis* 5.9 (2013).

[8] Sharma, Mukesh K., and Kapil Kumar Bansal. "Fuzzy Analysis Of Shortest Job First." *International Journal of Engineering Research & Management Technology*. 2015. 125-128

[9] Srinivasan, Rashmi KrishnaIyengar, V. Suma, and Vaidehi Nedu. "An Enhanced Load Balancing Technique for Efficient Load Distribution in Cloud-Based IT Industries." *Intelligent Informatics. Springer Berlin Heidelberg,* 2013. 479-485.

[10] Nehru, E. Iniya, Saswati Mukherjee, and Abhishek Kumar. "Deadline-based Priority Management in Cloud." *Artificial Intelligence and Evolutionary Algorithms in Engineering Systems. Springer India,* 2015. 745-751.

[11] Mondal, Ranjan Kumar, Enakshmi Nandi, and Debabrata Sarddar. "Load Balancing Scheduling with Shortest Load First." *International Journal of Grid and Distributed Computing* 8.4 (2015): 171-178.

[12] Ibrahim, Mohammed, and Adil Yousif. "Enhanced Job Scheduling Algorithm for Cloud Computing Using Shortest Remaining Job First (SRJF." (2009).

[13] Ru, Jia, and Jacky Keung. "An Empirical investigation on the simulation of priority and shortest-job-first scheduling for cloud-based software systems."*Software Engineering Conference (ASWEC),* 2013 22nd Australian. IEEE, 2013.

[14] Banerjee, Sourav, et al. "Development and analysis of a new cloudlet allocation strategy for QoS improvement in cloud." *Arabian Journal for Science and Engineering* 40.5 (2015): 1409-1425.

[15] Kalra, Mala, and Sarbjeet Singh. "A review of metaheuristic scheduling techniques in cloud computing." *Egyptian Informatics Journal* 16.3 (2015): 275-295.

[16] Buyya, Rajkumar, et al. "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility." *Future Generation computer systems* 25.6 (2009): 599-616.

[17] Calheiros, Rodrigo N., et al. "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms." *Software: Practice and Experience* 41.1 (2011): 23-50.

[18] Calheiros, Rodrigo N., et al. "Cloudsim: A novel framework for modeling and simulation of cloud computing infrastructures and services." *arXiv preprint arXiv:*0903.2525 (2009).

[19] Zhao, Jia, et al. "A Heuristic Clustering-Based Task Deployment Approach for Load Balancing Using Bayes Theorem in Cloud Environment." *IEEE Transactions on Parallel and Distributed Systems* 27.2 (2016): 305-316.

[20] Singh, Umang and Sharma, Ayushi "CloudSim Simulator Used for Load balancing in Cloud Computing." *International Journal of Emerging Technology and Advanced Engineering*, (2016) 246-255.