ISSN: 0067-2904

ESJF Algorithm to Improve Cloud Environment

Manal F. Younis
Department of Computer, College of Engineer, University of Baghdad, Baghdad, Iraq

Received: 4/4/2021 Accepted: 11/6/2021

Abstract

Nowadays, after the technological development in societies, cloud computing has become one of the most important technologies. It provides users with software, hardware, and platform as remote services over the Internet. The increasing number of cloud users has caused a critical problem in how the clients receive cloud services when the cloud is in a state of instability, as it cannot provide required services and, thus, a delay occurs. Therefore, an algorithm was proposed to provide high efficiency and stability to work, because all existing tasks must operate without delay. The proposed system is an enhancement shortest job first algorithm (ESJF) using a time slice, which works by taking a task in the shortest time first and then the longest first from the queue. Through the experiments in decreasing the waiting and completion time of the task, the result of the proposed ESJF algorithm was compared with the traditional shortest job first (SJF) algorithm by taking into account reducing tasks starvation. These algorithms were applied when all tasks arrived at the same time, and it proved that the ESJF algorithm works more efficiently compared to SJF.

Keywords: Cloud computing; Shortest Job First Scheduling; Burst time; Waiting Time; ESJF Scheduling

خوارزمية ESJF لتحسين بيئة السحابة

منال فاضل يونس

قسم الحاسبات، كلية الهندسة، جامعة بغداد، بغداد، العراق

الخلاصة

في الوقت الحاضر و بعد التطورالتكنولوجي في المجتمعات، اصبحت الحوسبة السحابية واحدة من أهم التقنيات ، فهي توفر للمستخدمين (البرامج ،الأجهزة، والمنصة) كخدمات عن بُعد عبر الإنترنت. و ان زيادة عدد مستخدمي السحابة تسبب حدوث مشكلة حرجة في كيفية تلقي العملاء للخدمات السحابية عندما تكون السحابة في حالة عدم استقرار حيث لا يمكنها تقديم الخدمات المطلوبة ، لذا يحدث تأخير. لذلك، تم اقتراح خوارزمية لتوفير كفاءة عالية واستقرار للعمل ، لأن جميع المهام الحالية يجب أن تعمل دون تأخير. النظام المقترح هو تحسين جدولة الخوارزمية لأقصر مهمة أولاً (ESJF) باستخدام شريحة زمنية ، والتي تعمل عن طريق أخذ مهمة ذات أقصر وقت أولاً ثم اخذ مهمة الأطول وقت من قائمة الانتظار وهكذا بقية المهام. من خلال التجارب في تقليل وقت الانتظار والانتهاء للمهام ، تمت مقارنة نتيجة خوارزمية ESJF المقترحة مع خوارزمية SJF التقليدية. الاخذ بنظر الاعتبار تقليل starvation للمهام. حيث تم تطبيق هذه الخوارزميات عند وصول جميع المهام في نفس الوقت ، وثبت أن خوارزمية ESJF تعمل بكفاءة أكبر مقارنة بخوارزمية SJF.

Email: Manal.fadhel@coeng.uobaghdad.edu.iq

Introduction

Cloud computing system is a revolution in new information technology (IT) that requires an efficient and robust architecture to be used in various systems that need sophisticated and large-scale computing [1]. Cloud computing enables users to access resources, such as software and hardware, through the Internet. In cloud computing, there are a variety of services that can be rented on a pay-per-use basis. Cloud providers, such as Amazon Web Services (AWS), Google, Microsoft Azure, and others offer these services to customers. The cloud is a multi-tenant computing system in which users can share resources [1]. Since the resources on distributed cloud servers are sophisticated and congested, resource allocation for cloud computing systems can become very complicated [2]. To devote these resources to client requests, they must be scheduled with effective task scheduling [1]. As shown in Figure 1 [3], various models (private, public, hybrid, and community) are used:

1. Private cloud: This type of cloud is designated for a specific organization or enterprise.
2. Public cloud: Any company may use a public cloud to obtain technology and resources. Amazon, Google, Microsoft, and other companies provide public cloud services.
3. Community cloud: Services and infrastructure are supplied to organizations with same interests.
4. Hybrid cloud: This type of cloud is a combination of the private and public clouds. Even though clouds are mixed up, each has its own special identity and thus helps with multiple deployments.

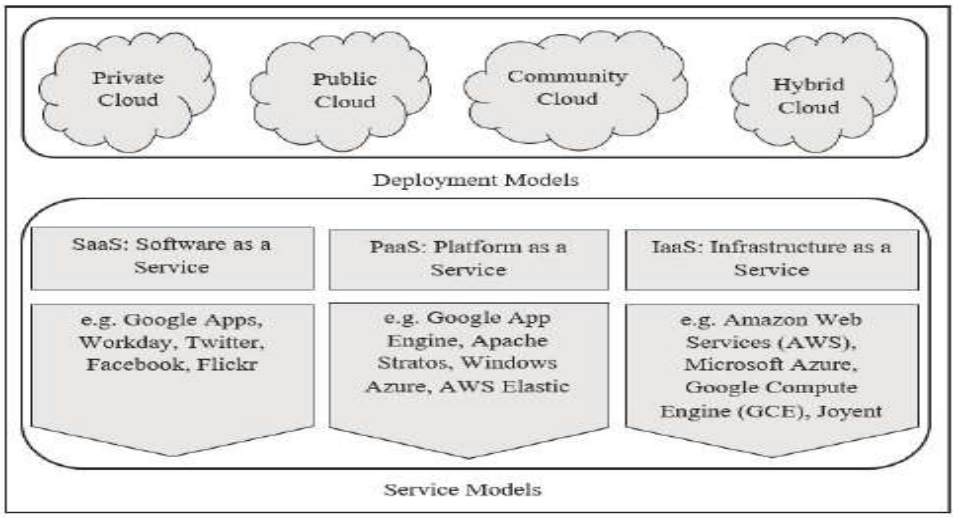

Figure 1- Cloud computing service model [4]

Various services (software, platform, and infrastructure) are provided geographically around the world; these services are described as in the following [5]:

1. Software as a Service (SaaS): This is a model in which a vendor's computer releases software for customers to use over the Internet. The payment for this service is determined by the length of time or the value of the service request. Yahoo Mail, Dropbox, and Gmail are examples of SaaS.
2. Platform as a Service (PaaS): This is a platform and environment that allow developers to build services and applications. Users can access this service through the internet, since it is hosted in the cloud. PaaS offerings include AWS, Salesforce, Microsoft Azure, and Heroku.

3. Infrastructure as a Service (IaaS): This is one of the most popular cloud computing service models. It provides links to computing resources in a virtual "cloud" network over the internet.

Cloud system resources are shared by large numbers of users; thus, massive task scheduling is an issue for cloud computing, which is a dynamic environment. The main imperative of a task scheduling algorithm in a cloud computing is to reach an optimal scheduling between user tasks and virtual machines (VMs). VMs, such as processing cores and memory, demand implementing each user's tasks. A huge amount of tasks are coming from different locations in the world. Thus, each cloud requires a good scheduling approach to determine the execution arrangement of the tasks [3]. Scheduling is the process that is accountable for locating the tasks sent to provide appropriate resources. These algorithms consider different metrics in which cloud computing system performance is measured. The most commonly used performance metrics are shown in Table 1.

Table 1- Performance Metrics in Cloud Computing [6]

Parameter	Description
Makespan	Algorithm completion time.
Load balancing	The process of allocating resources and completing the workload of cloud computing.
Execution time	The exact time taken to perform the task.
Completion time	The time in which the execution of task is completed.
Bandwidth	The amount of data that can move from one point to another in a certain period of time.
Response time	The amount of time after which a process gets the resource for the first time after entering the ready queue.
Resource utilization	The amount of resource that is busy in executing time.

In the remainder of this paper, the related work is described, an illustration on the proposed system model is given, experimental results and discussion are presented, and finally, the conclusion is offered.

Related Works

Due to the role that cloud computing plays in our lives, an increase in the number of its users, and thus problems in allocating resources to tasks, have appeared. Therefore, this issue became the focus of a large amount of research. Different methods are being used to process this problem, as follows:

Elmougy, *et al*. [7] suggested a hybrid task scheduling algorithm, which merges Shortest Job First (SJF) and Round Robin (RR) algorithms, utilizing dynamic variable task quantity to balance the waiting time between short and long tasks. In addition, the ready queue is split into two queues: the first one contains short tasks, while the other includes long tasks. This algorithm allocates two tasks from one queue, and one job from the other. The simulations showed that the algorithm gave results on SJF priority, RR, and time slicing priority-based Round Rubin by decreasing waiting and response times, with long tasks being starved partly.

Suri and Rani [8] presented four stages in their scheduling algorithm proposed model, which are minimization, grouping, ranking, and execution. They took into consideration some constraints that affected the algorithm performance, such as waiting time, completion time, and starvation. The tasks were sorted depending on the idea of SJF algorithm. The results of the experiments proved that their suggested algorithm was preferable over the First Come

First Serve (FCFS) and the Largest Processing Time First (LPTF) algorithms in enhancing the parameters of performance.

Mokhtar *et al*. [9] proposed a Hybrid-SJF-LJF (HSLJF) algorithm which combines SJF and LJF algorithms. This algorithm sorts the tasks in an ascending order. Then, it choses one task according to SJF and another rendering to LSF. Finally, it chooses the virtual machine that has the minimum completion time to perform the assigned task. The experimental results showed the notability of HSLJF in minimizing response time and execution time, while rising resource utilization and throughput when compared to existing algorithms.

Alworafi *et al*. [10] suggested a Modified Shortest Job First algorithm (MSJF) to reduce the time taken to complete the final task (Makespan) and the average response time, while maximizing resource utilization. The MSJF provides two functions, one computes the average task length, and the other balances the load amongst virtual machines. Sending the longest jobs to the quickest machine is one of MSJF's key advantages. The results revealed that the suggested MSJF outperforms SJF and FCFS in terms of performance.

Krishna *et al*. [11] employed a mixture of shortest job first and priority scheduling, which was examined in a cloud context. The conclusions exhibited that the average waiting time and turnaround time were considerably lowered, whereas the efficiency of cloud resource management was considerably boosted.

All the above suggested algorithms did not take into consideration the stability of cloud computing, especially if all the requests are arriving at the same time. The meaning of stability is to give big tasks an opportunity to be executed without delay in order to prevent dead lock state.

Proposed System Model

The proposed model is divided into three parts; the first is the tasks layer, the second is the broker, and the third is the service provider, as shown in Figure 2. In a cloud computing environment, a data center handles a large number of hosts. The host is a physical computing machine (server). Each host can be divided into a number of virtual machines (VMs). Each virtual machine has its own resources and configuration, and the cloud provider is responsible for providing services and computing users. The cloud broker acts as a middle layer between users and providers. It coordinates the use of the available resources and distributes them among the tasks, while maintaining cloud stability. Each task has its own set of characteristics and resources, including task length and identity.

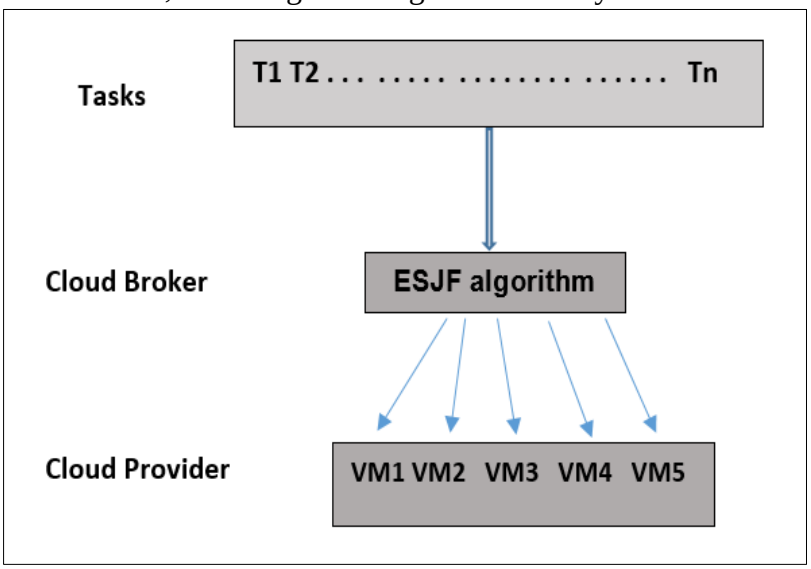

Figure 2- Proposed cloud computing system model

Task scheduling is a critical component that influences system performance. As a result, developing a good algorithm for assigning tasks to virtual machines is crucial. Hence, we present a modified algorithm to achieve these advantages.

This paper aims to develop the shortest job first scheduling algorithm in cloud computing, to achieve better work scheduling in order to minimize waiting and completion time. This algorithm is called Enhancement Shortest Job First (ESJF) with time slice.

In this proposed algorithm, firstly, the tasks are sorted in the first queue based on SJF algorithm. Next, the shortest task length is divided by two to get the time slice (time slice = shortest task length / 2). The estimated completion time would then be measured in the virtual machines to decide which is the quickest.

The non-completed task will be entered into a new circular queue and sorted in a descending order. This means that the task is taken from the beginning of the second waiting queue, and then, in the next time, the task is selected from the end of the queue according to its length. This idea of quantum time, long with sorting the tasks once in an ascending order and another in a descending order, aims to prevent the starvation as much as possible. These tasks are distributed among 5 VMs, which are operated in parallel.

The completion time (CT) can be calculated by the following Eq. 1 [9].

$$CT = BT + load_{VMj} \quad (1)$$

where BT refers to burst time (execution time) of the task, that is calculated by Eq. 2 [12].

$$BT = \frac{L}{MIPSj} \quad (2)$$

where L represents the task length that is measured by the amount of million instructions per second (MIPS) assigned to the VM_i and $load_{VMj}$ showed the previous current tasks performed by the VM_j as assigned in Eq. 3 [12]:

$$Load_{vmj} = \sum_{i=1}^{N} BT_i \quad (3)$$

where N is the number of tasks in a given VM. The proposed algorithm ESJF will chose the longest task, then the lowest CT will be selected. This process will continue for all the tasks, depending on the time slice.

Waiting Time (WT) can be calculated for task(i) as in Eq. 4 [12].

$$WTi = TTi - (TLi - ARTi) \quad (4)$$

where TT refers to turnaround time, TL is task length, and ART denotes arrival time. In this paper, all tasks arrive in the same time, then:

Arrival Time= 0 for all tasks. Hence, the waiting time is calculated as:

$$WTi = TTi - TLi \quad (5)$$

The pseudocode for the proposed ESJF algorithm is as follows:

Input: List of n tasks and list of m VMs.

Output: Mapping list of n tasks on available m VMs with fulfilled user and provider stability.

1. Start
2. Create new scheduling for task
3. Ascending order for task depending on execution time
4. For first task in schedule calculated
5. Slice time=execution time dive 2
6. Set slice as default
7. I=task number: F_task=0
8. Loop while task >=1
9. Execution for task(i) -= slice time
10. Descending order for task
11. Execution for task(i) -= slice time
12. If execution task (for 1-task number) =0
13. F_task ++

14. Task –
15. End if
16. Ascending order for task
17. End loop
18. Save task report for VM
19. End

Experimental Results and Discussion

The proposed ESJF algorithm was implemented and tested using the Python version 5.0 on a laptop whose configurations are: Core i7, 16 GB RAM, and 1TB hard disk. The presented algorithm was compared with the traditional SJF algorithm. Table 2 describes simulation environment parameters.

Table 2- Parameters of cloudsim simulation

Simulation Parameters	Value
Number of datacenters	1
Number of hosts	1
Number of VM per host	5
Host memory	16 GB
Host storage	1 TB
Host bandwidth	10000
VM MIPS	10000

The traditional SJF and the proposed ESJF algorithms were implemented and tested to 40 tasks executed in two environments, once in a normal environment and another in a cloud environment containing 5 vms. all these tasks arrive at the same time, i.e. arrival time = 0. These tasks are distributed among the vms. It may execute one task in more than one vm at the same time in order to provide speed and reduce waiting time, especially for the big task. The SJF algorithm works on the principle of arranging tasks in an ascending order and then executing them incrementally. In this algorithm, the execution of larger tasks is delayed, then the waiting time is large, so the system may enter the dead lock state. While the modified ESJF algorithm takes the first tasks from the queue, then the last tasks, which are the longest, then the second smallest tasks, followed by the second longest, and so on. Each task is executed in a specific time, which is half length of the shortest task. Tasks that are not completed in this time may be implemented in more than one vm.

Tables 3 and 4 show the effects of the proposed ESJF algorithm in contrast to the standard SJF algorithm in terms of waiting time and completion time, all of which are calculated in seconds according to the load processed in five parallel virtual machines. These tables demonstrate that ESJF algorithm achieved the stability of the system and protected it from deadlock situation; as opposed to SJF algorithm which may enter a long waiting state for long tasks. We observe in the SJF algorithm that the highest waiting time reached was 239, while the ESJF algorithm reached the highest waiting time at 238. In addition, the average waiting time in ESJF algorithm was equal to 400.92 which is less than that in SJF algorithm, that is 483.6.

Table 3- Comparison of the waiting time between traditional SJF and ESJF algorithm (in Sec.).

Task No	Exe. time	SJF	SJF waiting time	SJF waiting time in VMs	ESJF waiting time	ESJF waiting time in VMs	VM NO

1	9	4	0	0	0	0	3
2	9	4	4	0.48	13	0.78	2
3	5	4	8	0.96	26	0.78	1
4	8	4	12	1.44	39	2.34	2
5	9	4	16	1.92	52	1.56	1
6	4	4	20	2.4	65	9.75	5
7	9	4	24	2.88	78	2.34	1
8	8	4	28	3.36	91	5.46	2
9	4	4	32	3.84	103	3.09	1
10	4	5	36	4.32	115	3.45	1
11	5	5	41	4.92	128	19.2	5
12	7	5	46	5.52	141	12.69	3
13	4	5	51	6.12	154	4.62	1
14	7	5	56	6.72	166	14.94	3
15	6	5	61	7.32	178	21.36	4
16	4	5	66	7.92	190	5.7	1
17	6	5	71	8.52	201	24.12	4
18	4	6	76	9.12	212	6.36	1
19	5	6	82	9.84	224	26.88	4
20	5	6	88	10.56	236	35.4	5
21	8	6	94	11.28	238	14.52	2
22	9	6	100	12	230	6.9	1
23	4	6	106	12.72	218	6.54	1
24	4	6	112	13.44	206	30.9	5
25	8	6	118	14.16	195	11.7	2
26	8	7	124	14.88	183	10.98	2
27	6	7	131	15.72	171	20.52	4
28	9	7	138	16.56	159	4.77	1
29	6	8	145	17.4	146	13.14	3
30	6	8	153	18.36	133	15.96	4
31	6	8	161	19.32	120	10.8	3
32	5	8	169	20.28	107	12.84	4
33	5	8	177	21.24	95	14.25	5
34	9	9	185	22.2	82	2.46	1
35	6	9	194	23.28	69	6.21	3
36	6	9	203	24.36	56	5.04	3
37	5	9	212	25.44	43	5.16	4
38	4	9	221	26.52	30	4.5	5
39	5	9	230	27.6	17	2.55	5
40	7	9	239	28.68	4	0.36	3

Table 4- Comparison of the completion time between traditional SJF and ESJF algorithm (in Sec.).

Task No	Exe. time	SJF	SJF completion time	SJF completion time in 5 VMs	ESJF completion time	ESJF completion time in 5 VMs	VM NO

1	9	4	4	4	4	4	3
2	9	4	8	4.48	17	4.78	2
3	5	4	12	4.96	30	4.78	1
4	8	4	16	5.44	43	6.34	2
5	9	4	20	5.92	56	5.56	1
6	4	4	24	6.4	69	13.75	5
7	9	4	28	6.88	82	6.34	1
8	8	4	32	7.36	95	9.46	2
9	4	4	36	7.84	107	7.09	1
10	4	5	41	9.32	120	8.45	1
11	5	5	46	9.92	133	24.2	5
12	7	5	51	10.52	146	17.69	3
13	4	5	56	11.12	159	9.62	1
14	7	5	61	11.72	171	19.94	3
15	6	5	66	12.32	183	26.36	4
16	4	5	71	12.92	195	10.7	1
17	6	5	76	13.52	206	29.12	4
18	4	6	82	15.12	218	12.36	1
19	5	6	88	15.84	230	32.88	4
20	5	6	94	16.56	242	41.4	5
21	8	6	100	17.28	244	20.52	2
22	9	6	106	18	236	12.9	1
23	4	6	112	18.72	224	12.54	1
24	4	6	118	19.44	212	36.9	5
25	8	6	124	20.16	201	17.7	2
26	8	7	131	21.88	190	17.98	2
27	6	7	138	22.72	178	27.52	4
28	9	7	145	23.56	166	11.77	1
29	6	8	153	25.4	154	21.14	3
30	6	8	161	26.36	141	23.96	4
31	6	8	169	27.32	128	18.8	3
32	5	8	177	28.28	115	20.84	4
33	5	8	185	29.24	103	22.25	5
34	9	9	194	31.2	91	11.46	1
35	6	9	203	32.28	78	15.21	3
36	6	9	212	33.36	65	14.04	3
37	5	9	221	34.44	52	14.16	4
38	4	9	230	35.52	39	13.5	5
39	5	9	239	36.6	26	11.55	5
40	7	9	248	37.68	13	9.36	3

Figure 3 also shows how much time is spent waiting for tasks to be completed. As compared to

The current algorithms, our proposed ESJF algorithm improved the user system stability by reducing task waiting time. As can be seen in Table 3, as task length increases, the waiting time increases in the SJF, but, during the ESFJ algorithm, we observe that the waiting time for the longest tasks is slightly longer than the waiting time for the smallest tasks. We prove that

the long tasks do not wait long, as all tasks have a specific time slice, so the system is in stability to avoid dead lock.

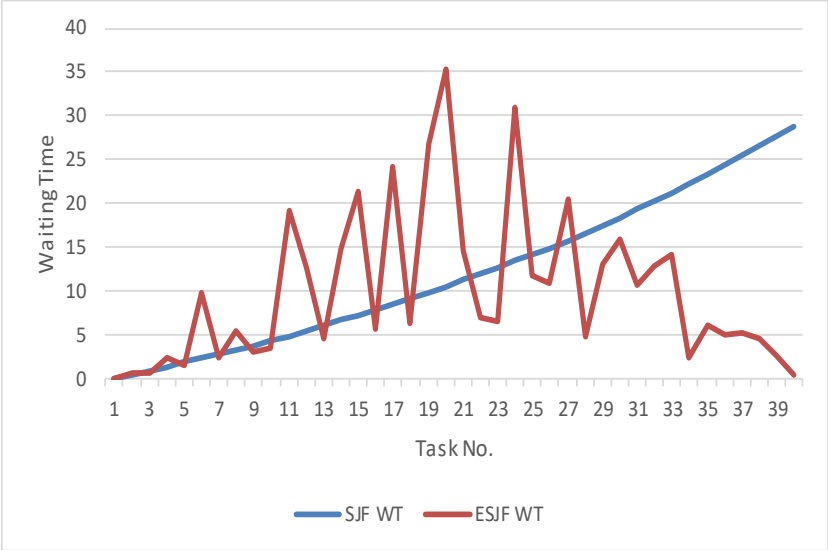

Figure 3- Comparison of waiting time between SJF and ESJF

Figure 4 also indicates the time taken to complete the tasks. Table 4 shows that, as the task duration increases, the SJF completion time increases, but the ESJF completion time performs the tasks in an ascending and descending order with time slice for each task. Hence, every task takes its time, then the delay will be less. These algorithms are implemented for 40 tasks of various lengths, distributed over 5 virtual machines.

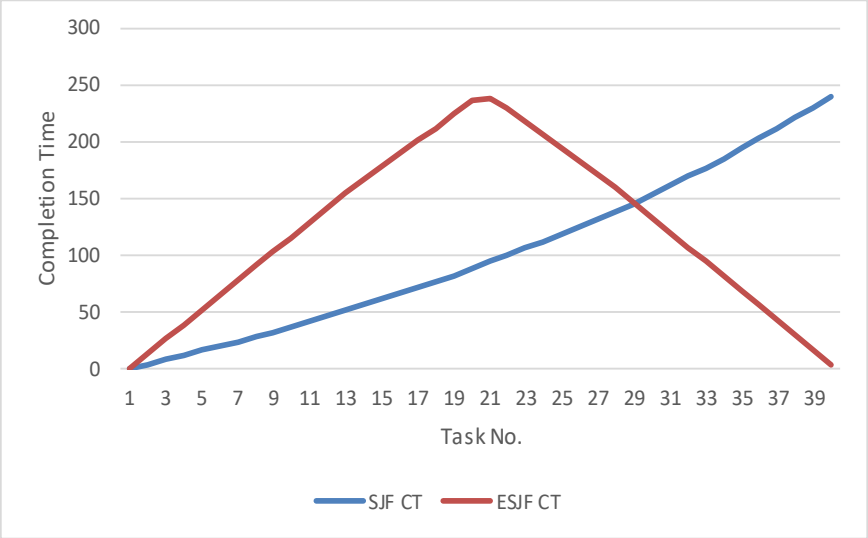

Figure 4- Comparison of completion time between SJF and ESJF

Conclusions

Task scheduling plays an important role in improving cloud computing performance. Especially, cloud computing is a dynamic environment with a large number of tasks that can run concurrently. This paper proposed an ESJF scheduling algorithm in order to minimize waiting and completion times of processing tasks, by the distribution of tasks among resources to improve stability. Using ESJF, tasks are increasingly sorted in the first queue by length and assigned to 5 virtual machines to process. If the task is not completed, and there is no free vm, it is added to a descending queue, which begins with the first task in the sorted queue and works its way down to the last task, depending on the time slice policy. The enhancement algorithm was implemented on 40 tasks distributed over 5 resources according

to vm load. Simulation results illustrated that the suggested algorithm outperformed the existing algorithms in terms of reduced waiting time and completion time; while on the other side, resource utilization was increased by achieving stability. As a future work, we are planning to expand the study by assessing the performance of these algorithms on a real workload, taking into account multiple metrics.

References

[1] Younis, M.I., Younis, M.F., Abed, M.M. and Alserawi, A.A. "Development of an Attendance System Based on Cloud/Fog Computing with Data Recovery Capability". *Iraqi Journal of Science*, pp.1190-1201, 2020. DOI: 10.24996/ijs.2020.61.5.26.

[2] DAR, A.R., Ravindran, D. and Islam, S. "Fog-based Spider Web Algorithm to Overcome Latency in Cloud Computing". *Iraqi Journal of Science*, pp.1781-1790, 2020. DOI: 10.24996/ijs.2020.61.7.27.

[3] Rashid, A. and Chaturvedi, A. "Cloud computing characteristics and services: a brief review". *International Journal of Computer Sciences and Engineering*, vol. 7, no. 2, pp.421-426, 2019. DOI: https://doi.org/10.26438/ijcse/v7i2.421426.

[4] Panwar N. and Rauthan M. Analysis of various task scheduling algorithms in cloud environment. In 2017 7th International Conference on Cloud Computing, Data Science & Engineering-Confluence, pp.255-261, 2017. IEEE. DOI: 10.1109/CONFLUENCE.2017.7943159.

[5] Abed, M.M. and Younis, M.F. "Developing load balancing for IoT-cloud computing based on advanced firefly and weighted round robin algorithms". *Baghdad Science Journal*, vol. 16, no. 1, 2019. DOI: org/10.21123/bsj.2019.16.1.0130.

[6] Alhaidari, F., Balharith, T. and Eyman, A.Y., Comparative Analysis for Task Scheduling Algorithms on Cloud Computing. In *2019 International Conference on Computer and Information Sciences (ICCIS)* (pp. 1-6). IEEE. DOI: 10.1109/ICCISci.2019.8716470.

[7] Elmougy, S., Sarhan, S. and Joundy, M. "A novel hybrid of Shortest job first and round Robin with dynamic variable quantum time task scheduling technique". *Journal of Cloud computing*, vol. 6, no. 1, pp.1-12, 2017. DOI: 10.1186/s13677-017-0085-0.

[8] Suri, P.K. and Rani, S., Design of task scheduling model for cloud applications in multi cloud environment. In *International Conference on Information, Communication and Computing Technology*, pp. 11-24, 2017. Springer, Singapore. DOI: 10.1007/978-981-10-6544-6_2

[9] Alworafi, M.A., Dhari, A., El-Booz, S.A., Nasr, A.A., Arpitha, A. and Mallappa, S. "An enhanced task scheduling in cloud computing based on hybrid approach". In *Data Analytics and Learning*, pp. 11-25, 2019. Springer, Singapore. DOI: 10.1007/978-981-13-2514-4_2.

[10] Alworafi, M.A., Dhari, A., Al-Hashmi, A.A. and Darem, A.B., December. An improved SJF scheduling algorithm in cloud computing environment. In *2016 International Conference on Electrical, Electronics, Communication, Computer and Optimization Techniques (ICEECCOT)*, pp. 208-212, 2016. IEEE. DOI: 10.1109/ICEECCOT.2016.7955216.

[11] Krishna, A.V., Ramasubbareddy, S. and Govinda, K., Task scheduling based on hybrid algorithm for cloud computing. In *International Conference on Intelligent Computing and Smart Communication 2019*, pp. 415-421, 2020. Springer, Singapore.

[12] Arif, K.I. "A Hybrid MinMin & Round Robin Approach for Task Scheduling in Cloud Computing". *International Journal of Control and Automation*, vol. 13, no. 1, pp.334-342, 2020.