IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Mathematical Modelling of Engineering Problems
Vol. 9, No. 5, October, 2022, pp. 1201-1209
Journal homepage: http://iieta.org/journals/mmep

Modeling Algorithms for Task Scheduling in Cloud Computing Using CloudSim

Mohammed Gollapalli[1*], Abrar Alamoudi[2], Arwa Aldossary[2], Arwa Alqarni[2], Sarah Alwarthan[2], Yousof Z. AlMunsour[1], Mamoun M. Abdulqader[2], Rami M. Mohammad[1], Sghaier Chabani[3]

[1] Department of Computer Information Systems, College of Computer Science and Information Technology, Imam Abdulrahman Bin Faisal University, Dammam P.O. Box 1982, Saudi Arabia
[2] Department of Computer Science, College of Computer Science and Information Technology, Imam Abdulrahman Bin Faisal University, Dammam P.O. Box 1982, Saudi Arabia
[3] Network and Communications Department, College of Computer Science and Information Technology, Imam Abdulrahman Bin Faisal University, Dammam P.O. Box 1982, Saudi Arabia

Corresponding Author Email: magollapalli@iau.edu.sa

https://doi.org/10.18280/mmep.090506

Received: 18 April 2022
Accepted: 4 October 2022

Keywords:
cloud computing, FCFS, Max-Min, RR, virtualization, CloudSim

ABSTRACT

As the number of cloud users are spontaneously growing globally, there is an urgent need to constantly provide quality services to consumers. Consequently, task scheduling plays an essential role in improving the performance of the cloud computing environment. Most of the published research in this field share common goals, which can be summarized in maximizing resource utilization, reducing cost, and increasing performance. This research provides the foundation knowledge on the latest works done to enhance and optimize the existing task scheduling algorithm in cloud computing by considering various parameters. Furthermore, in this study, we have applied comparative study to analyze the performance of three task scheduling algorithms namely Max-Min, First Come First Serve (FCFS), and Round Robin (RR) in cloud computing environments based on the performance metric of the Virtual Machines (VM) resources' cost, average time and makespan to find the best performing algorithm in the cloud environment. The experimental evaluations were conducted using CloudSim simulation tool. The results show that Max-Min achieved better performance based on makespan and average waiting time than other algorithms in Space and Time-shared policies.

1. INTRODUCTION

Cloud computing has been the technology trend in recent decades for hosting, storing, and distributing services through the Internet. Cloud computing offers a variety of services at various levels to satisfy the needs of users. These services have been widely used in many different applications and domains. Three types of cloud computing services on different layers are used by private and public organizations to reduce their operational consumption as shown in Figure 1 [1].

(1) Infrastructure as a service (IaaS) offers users physical equipment like storage, services, and virtual machine networks. Amazon EC2 and RackSpace Cloud are examples of IaaS [2].

(2) Platform as a service (PaaS) offers developers the required computing platform to run and develop their applications such as Apprenda and Google Apps engine [3].

(3) Software as Service (SaaS) has allowed the end-user to access the services or applications that cloud providers offer directly. Cisco WebEx and Google Apps are an example of SaaS [3].

The public, private, and hybrid cloud deployment options are the most common. All users could access a public cloud based on a pay-as-you-use. However, the Private cloud is when a particular organization owns the cloud. The Hybrid cloud consists of both Private and Public Clouds [3].

Since there is a massive number of users using cloud computing, it is vital to provide good quality services for them. One of the most prominent roadblocks in cloud computing is task scheduling, which focuses on distributing activities to available resources at the right times to deliver a valuable Quality of Service (QoS) [4]. The efficiency of cloud computing services is influenced significantly by task scheduling algorithms. Many studies have been published with the goal of improving the quality of cloud services by reducing the time it takes to complete a task and the time it takes to do it. Furthermore, one of the key objectives used to improve task scheduling algorithms is to increase resource utilization.

The primary contribution of this study is to summarize the recent studies in this area and to present a comparison between three task scheduling algorithms: FCFS, RR, and Max-Min. CloudSim simulator toolkit will be used to determine the best algorithm according to the performance metric of the VMs resources' cost, average time and makespan. Section 2 introduces the key concepts in task scheduling. Section 3 discusses previous studies on task scheduling that have been published. The proposed methodology is presented in Section 4. Section 5 discusses the final outcomes. Section 6 summarizes the findings and future research.

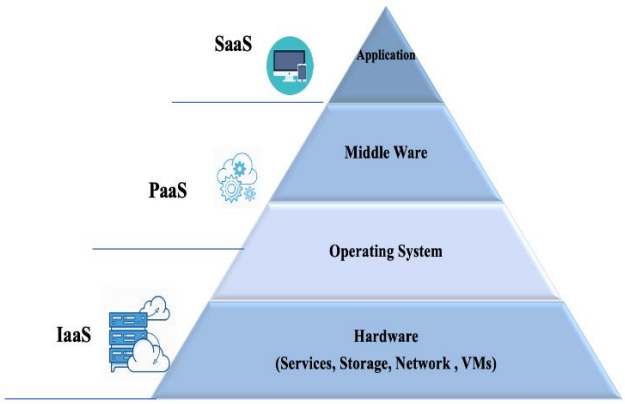

Figure 1. Different models in the cloud computing

2. BACKGROUND

This section introduces and discusses the fundamental concepts of task scheduling in cloud computing environments. The first section goes over the scheduling level in cloud computing, and the second section goes over the scheduling policies. The remaining sections go over the various scheduling algorithms and performance metrics. The final section provides an overview of the simulation tool used in this study to assess the efficiency of task scheduling algorithms.

2.1 Cloud computing scheduling levels

The techniques for assigning resources to particular activities are referred to as scheduling. Utilizing scheduling algorithms is mostly intended to raise the caliber of cloud computing. In order to do this, average waiting times and makespan must be kept to a minimum while utilization is increased. Scheduling the cloud's resources are categorized into two levels: virtual machines (VMs), and host. At the host level, VMs are assigned to physical machines (PMs) by using a VM scheduler, this level is known as VM scheduling. Tasks at the VM level are assigned to VMs for the purpose of execution by using a task scheduler, this level is known as Task Scheduling [5], as shown in Figure 2. This study will concentrate on the VM level, with the goal of scheduling tasks, which is the most difficult issue in cloud computing.

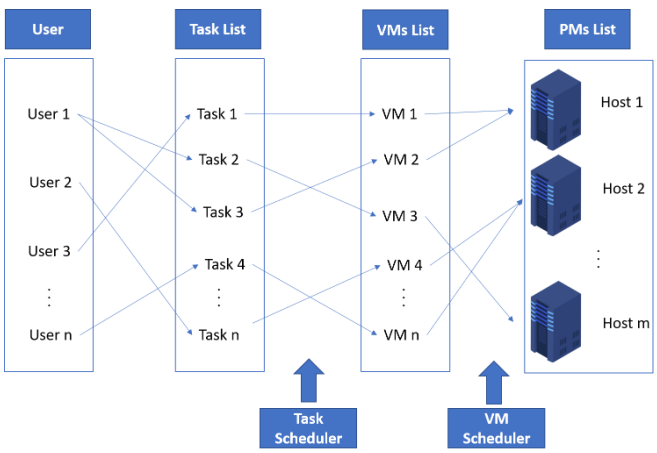

Figure 2. Cloud Resources Scheduling level (adapted with modification from [5])

2.2 Scheduling policies

According to the scheduling policy, the tasks are allocated to the resources. Two types of scheduling policies are provided by cloud computing: Space and Time-shared policies. One task is permitted to be performed at a given VM in Space-Shared policy, which means that the task will own the VM until it completes its execution. In Time-Shared policies, the VM is being shared by multiple tasks, which means it allows several tasks to run in parallel over a VM [6, 7]. This work focuses on both policies to evaluate various task scheduling algorithms.

2.3 Task scheduling classes

Task scheduling is a technique that has been used to assign resources to a particular task to be completed effectively. This section contains a brief description of scheduling algorithm types including Static/Dynamic scheduling, Preemptive/Non-Preemptive scheduling, and Heuristic /Metaheuristic scheduling approach.

2.3.1 Static/Dynamic scheduling

There are two types of task scheduling: static task scheduling and dynamic task scheduling. When the execution period is known and there is a minimum running time, static task scheduling is used. Dynamic task scheduling, on the other hand, has a maximum running time and the execution period is unknown [4].

2.3.2 Preemptive/ Non-Preemptive scheduling

Preemptive scheduling enables a newly arrived task, that has a small job or a higher priority, to be executed instead of the currently running task. As a result, the current task will be halted, and the new task will be allocated to the instant of the resource in order to run. On the other hand, non-preemptive scheduling does not allow to a running task to be interrupted. As a result, if the resource is allocated to a task, this task will not stop even if a higher priority task arrives [8, 9].

2.3.3 Heuristic/Metaheuristic

The problem of task scheduling is that it is a non-polynomial complete problem for the system. There are two approaches to task scheduling algorithms: The heuristic approach and the metaheuristic approach [10]. Heuristic approach offers a specific solution for a particular problem. Whereas the metaheuristic approach provides a consistent solution and a master strategy that can solve a wide range of problems. Genetic Algorithm (GA) is an example of metaheuristic algorithms, whereas First Come First Serve (FCFS), Round Robin (RR), and Min-Max are examples of heuristic algorithms. The metaheuristic algorithms require more time to execute than heuristic algorithms [11, 12].

The primary goal of this research is to evaluate and compare the following algorithms based on the performance metrics of the VMs resources' cost, average time, and makespan [13].

(1) First Come First Serve (FCFS): Is a heuristic non-preemptive scheduling algorithm where the tasks are queued based on the time which the task has arrived, then they will be assigned to the available resource to be executed.

(2) Round Robin (RR): Is known as a heuristic preemptive scheduling algorithm which works in the same way that FCFS does but with considering the time quantum. That means the task will be assigned to the available resource for a certain time

quantum then the task will be preempted. The preempted task will be queued back to get another chance to complete its execution.

(3) Max-Min: Is a heuristic method for scheduling tasks based on their completion time on each VM. It chooses the task with the shortest completion time among all available VMs to be executed on the best free VM with the shortest completion time.

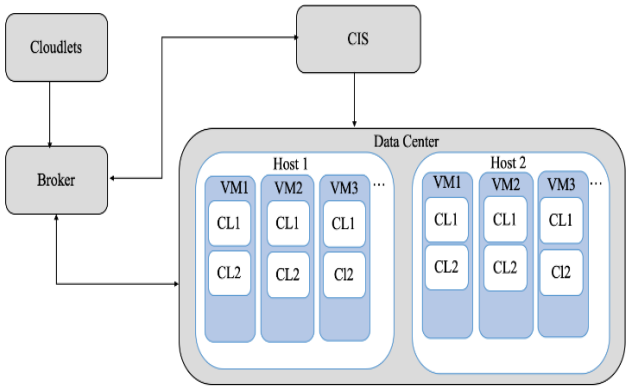

Figure 3. CloudSim entities (adapted with modification from [14])

2.4 Parameter overview

Several task scheduling algorithms were evaluated based on different parameters. This section presents a brief discussion about the most common parameters [15, 16].

(1) Makespan: The maximum amount of time for completing the most recent tasks that have been scheduled. The makespan of scheduling algorithms must be kept to a minimum for the system's performance.

(2) Execution Time: The amount of time required to complete the assigned task. A better scheduling algorithm's optimal goal is to minimize execution time.

(3) Load balancing: It is the process of managing and distributing workloads across multiple servers in order to maximize system performance and avoid system downtime. Many scheduling algorithms strive to maintain load balance in order to improve the performance of the cloud system.

(4) Deadline: It is the maximum time in which a task must be completed. Efficient scheduling algorithms always attempt to implement tasks within the limits of the deadline.

(5) Scalability: It is the capacity of the system to operate well when its size has increased or de-creased to meet the needs of the user.

(6) Throughput: The throughput refers to how many tasks have been executed in a given amount of time. The cloud environment always aims to give high throughput.

(7) Resource Utilization: It demonstrates how efficiently the resources at hand are used. Increasing the likelihood that resources will be completely utilized is a key component of improving resource utilization. One of the goals that cloud environments are aiming to accomplish is to maximize resource consumption.

(8) Average Waiting Time: It refers to the time interval for the task before starting their execution.

(9) Cost: It is an economic parameter that refers to the total amount of payment required of using the resources, which are charged by cloud consumers to the cloud providers.

2.5 CloudSim overview

In order to evaluate scheduling algorithms before their actual implementation in a cloud environment, a cloud computing simulator tool, such as CloudSim, can be used. CloudSim [17] is a simulation tool used to simulate the cloud computing environment's infrastructure and services. Cloud computing simulator tools are used to evaluate the new services and algorithms with-out any concern about performance issues that may occur when they are actually implemented on the real cloud.

CloudSim affords a set of functions that form the basis of the cloud environment including: Cloud system entities' creation (data center, broker, hosts and VMs), simulation clock management, tasks handling and queuing, and connection between cloud components. The Cloud Information Service (CIS) is a component of CloudSim that includes the entire data center information. The data center has one or more physical machines (hosts), which include many of the VMs. VMs are responsible for executing the tasks and applications which are called cloudlets in CloudSim. The broker is responsible to assign the VMs to a particular host, then assign the cloudlets to the VMs. When the cloudlets of a specific VM completed, then the broker will destroy the VM that is finished all its cloudlets [14]. Figure 3 shows the CloudSim entities.

3. LITERATURE REVIEW

This section presents and analyzes previous research in the domain of task scheduling algorithms conducted in the cloud computing environment. The related works are divided into two sections: The first shows the improved task scheduling algorithms, and the second section presents studies that evaluate existing scheduling algorithms.

3.1 Task scheduling enhancing algorithms

Hlaing and Yee [18] proposed a new static task algorithm to improve the efficiency of cloud computing environments. A queue of independent tasks and virtual machines will be considered as an input to the new algorithm. An independent task will be allocated to a proper VM by considering the VM's processing cost and power. Authors used a CloudSim tool to assess the proposed algorithm by comparing between the proposed algorithm and two existing algorithms, which are Shortest Job First (SJF) and FCFS. Authors found that the proposed algorithm outperformed FCFS and SJF by minimizing the cost and execution time.

Mazumder et al. [19] proposed a new strategy for dynamically allocating algorithms based on task types and minimizing the restrictions of some well-known algorithms such as Min-Min and Max-Min. They also presented the trade-off between the makespan and the average waiting time. The results of the experiment revealed that the proposed algorithm outperforms the Min-Min algorithm in terms of makespan, while outperforming the Max-Min algorithm in terms of average waiting time.

Fang et al. [20] enhanced task scheduling algorithms depend on an adaptive genetic algorithm (AGA). The suggested algorithm aims to op-timize the total task time and balance the load of each computing resource. They compared their proposed work with traditional genetic algorithm and adaptive genetic algorithm by using the CloudSim platform.

They found that the suggested algorithm achieved good performance compared to others. Also, they noted that if the jobs' number is minimal, then the proposed algorithm is not as efficient as AGA.

Sarvabhatla et al. [21] suggested a new methodology for reducing the energy usage of virtual machines in data centers. The advantage of this model is that it assigns the task dynamically to VM that has the highest efficient of energy. They compared the model with two algorithms which are Most Efficient Server First (MESF) and Random Scheme Algorithm considering response time and energy cost metrics. The outcomes presented that the proposed approach has a high-performance advantage over MESF and Random Scheme Algorithm.

Sood et al. [22] proposed the Hybridized Firefly Gravitational Search Algorithm (HFGSA), which is based on two algorithms: The Firefly Algorithm (FA) and the Gravitational Search Algorithm (GSA) (GSA). The authors used the CloudSim toolbox to compare the proposed algorithm to the FA, GSA, and ACO algorithms in diverse situations. In certain cloud setups, the suggested approach takes less time to execute than others.

Kaur and Sengupta [6] proposed an enhanced Time-Shared algorithm for the successful assignment of tasks to VMs. The proposed algorithm aims to minimize the free RAM space in the host and increase the task's response time. The researchers compared their proposed work with Time-Shared scheduling police by using the CloudSim tool. They evaluated these algorithms based on response time and total delay metrics. They found that their proposed work obtained better performance in both metrics.

Orthogonal Taguchi-Based-Cat Swarm Optimization is a revolutionary work scheduling technique presented by Gabi et al. [23]. (OTB-CSO). The proposed approach was tested using the CloudSim tool and the makespan, with the goal of reducing dynamic scheduling execution time. Several algorithms were compared to the suggested work. They discovered that the suggested approach outperformed the competition by achieving the shortest possible makespan.

Elmougy et al. [24] introduced a hybrid scheduling system based on SJF and RR with a dynamic variable task quantum termed (SRDQ). Their goal was to solve the scheduling algorithm starving problem in cloud computing. They put SRDQ to the test using the CloudSim simulation tool, comparing the results against the Time Slice Priority Based RR (TSPBRR), RR, and SJF algorithms. The experiment found that the proposed algorithm minimizes reaction time, waiting time, and moderates the task's starving issue when the execution time is considerable.

Gupta and Garg [10] employed a meta-heuristic strategy to improve job scheduling algorithms based on the ACO algorithm, or load balancing ant colony optimization algorithm (LB-ACO). By improving load balancing and reducing makespan time, this technique seeks to enhance resource use and execution performance. They compared the suggested approach to a fast and elitist multiobjective genetic algorithm using the CloudSim tool (NSGA-II). Given the desired metrics, they discovered that the LB-ACO outperformed the NSGA-II.

Kumari and Jain [25] introduced a new approach that attempted to reduce the makespan and maximize resource utilization. The proposed approach, known as Max-Min PSO, combines the Particle Swarm Optimization (PSO) technique with the Max-Min Algorithm. Using the CloudSim tool, the proposed work is assessed and compared to two existing algorithms: Bee Colony Optimization Algorithm and Particle Swarm Optimization Algorithm (PBCOPSO). When compared to the PBCOPSO method, the average makespan and CPU utilization improved by 5.01 percent and 3.63 percent, respectively.

Hicham and Chaker [26] devised a novel CPU allocation mechanism to minimize the average waiting time for the task. Using the CloudSim toolkit, the suggested technique, RR based on the Average Burst Time of the Task (RRABT), was compared to the original RR. Authors found that the growing in the cloudlets numbers, the average waiting time of the RRABT will be lower than the average waiting time for the traditional RR. Moreover, two more algorithms, which are FCFS and SJF, were evaluated with different numbers of cloudlets. They discovered that by increasing the number of tasks, the FCFS' average waiting time will increase when compared to the SJF's average waiting time. Finally, the SJF has the shortest average waiting time, followed by the RRABT, FCFS, and RR.

Another task scheduling technique was developed [27], with the goal of completing jobs with low latency in the shortest time possible. Tasks are classified into five groups in the proposed algorithm, Grouped Tasks Scheduling (GTS), based on the similarity of the task's features. After experimenting with the suggested method, they discovered that the GTS algorithm took less time to execute than the Min-Min algorithm. They also achieved lower latency than the Min-Min and TS algorithms.

Agarwal and Srivastava [28] suggested a Genetic algorithm-based task scheduling method and compared it to the CloudSim tool's FCFS algorithm and Greedy-based approach. They discovered that genetically based task scheduling outperforms others in terms of the execution time parameter.

3.2 Task scheduling evaluating algorithms

Ibrahim et al. [29] conducted a comparative analysis of eight static task heuristic scheduling algorithms, including Min-Min, Load-Balanced-Improved-Min-Min (LBIMM), Minimum Completion Time (MCT), Resource-Aware-Scheduling-Algorithm (RASA), Resource-Aware-Load-Balancing-Algorithm (RALBA), Max-Average (MaxAvg (TASA). The study took into account the following factors: makespan, throughput, and resource consumption. The comparison of algorithms was carried by using CloudSim tools. In comparison to all other algorithms, the TASA attained a high performance in the aforementioned parameters, according to the output.

Alhaidari et al. [14] compared various algorithms called FCFS, SJF, RR, and LTF based on the completion time by using the CloudSim tool. They applied 20 experiments according to two policies of resource allocation, which are Space-Shared and Time-Shared policies with distinct scenarios. The outcomes of the study revealed that the SJF achieved higher performance than the other algorithms in terms of the Space-Shared poli-cy in whole scenarios. Moreover, the performance of the Time-Shared re-source policy performed better than the other policy with completion time metric.

A comparative evaluation of many contemporary work scheduling algorithms was offered by Anushree and Xavier [3]. The comparative study was proposed based on performance

indicators, algorithm benefits, and algorithm drawbacks [30, 31]. As a result of this research, they discovered that no single strategy can attain all of the essential parameters.

Pratap and Zaidi [32] worked on a comparative study between three of the static task scheduling algorithms, which are FCFS, SJF, and RR by using CloudSim tools. They used Time-Shared and Space-Shared policies of execution on the cloudlet while considering the waiting time and turnaround time for each algorithm. They found that the RR achieved higher performance com-pared with others.

Kumar et al. [33] compared various static load balancing algorithms, which are SJF, FCFS, equal distribution of task, and unequal distribution of task. The authors contrasted the algorithms based on the execution time by using CloudSim tools to measure the performance, assuming that the tasks and virtual machines are using Space-Shared policies. They found that the equal distribution approach has better performance as compared to others.

Borrowed-Virtual-Time (BVT), Start-Time-Fair-Queuing (STFQ), and Weighted-Round-Robin are the three task scheduling methods studied by Jambigi et al. [34]. (WRR). The CloudSim simulator was used to simulate all three algorithms. They discovered that the BVT had greater energy consumption efficiency, but had the longest execution time when compared to the others, based on execution time and energy consumed.

The task scheduling approaches assessed in this study [35] were Minimum-Execution-Time (MET), Max-Min, Min-Min, Sufferage, Minimum-Completion-Time (MCT), and FCFS. The comparison was made using the following metrics: the degree of imbalance, cost, throughput, and makespan. The program that was utilized to compare all six algorithms was CloudSim. They discovered that Min-Min was the optimal method for maximizing cost, throughput, and makespan. The rest of the algorithms, on the other hand, performed admirably in terms of work scheduling in IaaS cloud computing.

In conclusion, multiple studies have been conducted to assess work scheduling algorithms using a variety of performance indicators. We discovered that just a few studies looked at scheduling approaches using Time and Space Shared rules together. The CloudSim program was used in the majority of the publications to test the performance of scheduling approaches. As a result of the findings of previous studies, the algorithms employed in this study will be FCFS, RR, and Max-Min, with the makespan, average waiting time, and cost of using VMs parameters taken into account. The policies of Time-Shared and Space-Shared are considered. Finally, the CloudSim tool will be used to assess how well various scheduling strategies compare.

4. METHODOLOGY

The methodology used in this research for comparing the behavior of various task scheduling algorithms in cloud environments is described in this section [36, 37]. The comparison was carried out using CloudSim simulation tool which was used to configure the cloud environment and execute the scheduling algorithms. The study started by determining the characteristics of the main entities in CloudSim, such as Datacenter, Physical Machine/Host, and Virtual Machine. Then we applied the algorithms to schedule the set of the proposed tasks and compared their behavior based on the makespan, average waiting time, and cost parameters. Moreover, we examined these algorithms in both resource scheduling policies which are Space and Time shared policies.

4.1 Simulation workflow on CloudSim (12 steps)

This section discusses the simulation workflow which is applied in this study to compare between the three algorithms: FCFS, RR, and Max-Min. First, we specify the characteristics of the simulation entities as mentioned in steps 1 through 9. Then, the simulation for the task scheduling algorithm is initiated. Once the simulation is successfully completed, the Cloud Information Service (CIS) will ask the entities to shut down. Finally, the simulation output will be printed. The simulation output contains the start time and finish time for executing each cloudlet, which is used in the comparison between the performance of each algorithm. The steps of cloud computing task scheduling simulation are as follows [17]:

(1) Set the users' number (related to brokers' number)
(2) Set common variables (current time, user number, ...)
(3) Create cloud information service (CIS)
(4) Create data center (DC) instance by specifying the Processor Elements (PEs), hosts that house the PEs, and data center characteristics (time-zone, cost, CostperMem)
(5) Processing Element (PE): MIPS is the computing capacity of the PE.
(6) Host:
a. RAM
b. Storage (secondary storage)
c. Bandwidth (BW)
(7) Create data center broker where it gets the information of the data center from the CIS and assigns the cloudlets to the resources available in the data center.
(8) Create VMs and identify their characteristic:
a. Number of Cores (PEs).
b. RAM
c. Bandwidth (BW)
(9) Share the VM information with broker.
(10) Determine the characteristics of each cloudlet:
a. Required MIPS/ length (Millions of Instructions Per Second)
b. Required Bandwidth (BW)
c. Required RAM
d. Required number of Cores (PEs)
(11) Share the cloudlet information with broker
(12) Start simulation
(13) Stop simulation (shut down the entities)
(14) Print the results of simulation

4.2 CloudSim simulation configuration

This section describes the properties of CloudSim entities that are used for simulating the cloud environments to test and measure the efficiency of task scheduling techniques. The configuration of cloud simulation will be as follow:

(1) One data center which has a single host. The characteristics of data center are presented in Table 1.

(2) One host with one core, 1000 MIPS computing capacity, and 2048 MB of RAM. Table 2 presents the host's properties.

(3) Two VMs allocated to a single host. Time-Shared allocation policy is utilized for assigning the VMs to the host resources. The characteristics of the VMs are presented in Table 3.

(4) The experiment tested different sets of cloudlets that have been executed in the proposed cloud environment. Each cloudlet has a different length, the characteristics of the cloudlets are presented in Table 4.

Table 1. Data center characteristics

Characteristic	Value
No. of data center	1
No. of hosts	1
System architecture	x86
Operating system	Linux
Time-zone	10.0
CPU's cost	3.0
Memory's cost	0.05
Bandwidth's cost	0
Storage's cost	0.001

Table 2. Host properties

Characteristic	Value
PEs	1
Computing Capacity of the PEs (MIPS)	1,000 MIPS
RAM for each host	2,048 MB
Bandwidth (BW)	10,000
Storage	1,000,000

Table 3. Virtual machine characteristics

# of VM	Pe	MIPS	RAM	Bandwidth
VM 1	1	300	512	1,000
VM 2	1	500	512	1,000

Table 4. Cloudlet properties

Characteristic	Value
# of Cloudlets	From 10 to 50
Cloudlets Length	Different length from 1000 to 5100
CPU Utilization	Full
RAM Utilization	Full
Bandwidth Utilization	Full

5. RESULTS AND DISCUSSION

The results obtained by evaluating the three task scheduling algorithms in terms of makepan, average waiting time, and the cost of using the VM resources metrics are discussed in this section. The experiment has gone through several phases [38], we have evaluated each algorithm by considering two points: a) the effect of raising the number of cloudlets, and b) the resource allocation policies that have been applied.

Table 5, Table 6, and Table 7 show the performance metrics of the FCFS, RR, and Max-Min algorithms in Space-Shared and Time-Shared policies. The results indicated that the Time-Shared policy outperforms the Space-Shared policy considering the cost of using VMs resources and the average waiting time. The main reason for the superiority of the Time-Shared policy over the Space-Shared policy is that Time-Shared policy distributes the processing capabilities of the VMs among the cloudlets. Moreover, the results show that the two policies are approximately similar based on the makespan metric.

Table 5. Makespan of each algorithm in both policies

	Space-Shared			Time-Shared		
# of cloudlets	FCFS	RR	Max-Min	FCFS	RR	Max-Min
10	50.0993	50.0993	38.136	50.09	50.09	38.1667
20	100.43	100.43	76.5847	100.4633	100.4633	76.6167
30	151.0947	151.0947	114.53	151.12	151.12	114.4833
40	202.094	202.094	151.972	202.1	202.1	151.954
50	253.4253	253.4253	190.786	253.4267	253.4267	190.916

Table 6. Cost of each algorithm in both policies

	Space-Shared			Time-Shared		
# of cloudlets	FCFS	RR	Max-Min	FCFS	RR	Max-Min
10	25.1541	25.1542	25.1686	18.4560	18.4560	18.6888
20	50.4491	50.4491	50.5707	37.0440	37.0440	37.2694
30	75.9538	75.9538	76.0059	55.7896	55.7896	55.9844
40	101.5806	101.5806	101.700	74.706	74.706	74.784
50	127.378	127.378	127.522	93.764	93.764	93.903

Table 7. Average waiting time of algorithm in both policies

	Space-Shared			Time-Shared		
# of cloudlets	FCFS	RR	Max-Min	FCFS	RR	Max-Min
10	10.6893	10.6893	10.155	0	0	0
20	25.4834	25.4834	24.0682	0	0	0
30	40.4216	40.4216	37.9875	0	0	0
40	55.4572	55.4571	52.0553	0	0	0
50	70.6223	70.6223	66.2943	0	0	0

From the results, we found that Max-Min outperformed FCFS and RR in terms of the makespan metric in both policies as shown in Figure 4 and Figure 6. Moreover Max-Min achieved a better average waiting time compared to FCFS and RR in terms of Space-Shared policy as shown in Table 7. On the other hand, RR and FCFS performed similar to Max-Min in terms of the total cost in both policies, as shown in Figure 5 and Figure 7. Overall, the performance measurement for the three algorithms has increased when raising up the cloudlet's number. Finally, the simulation revealed that the Max-Min has the best performance in terms of makespan and average waiting time in both policies.

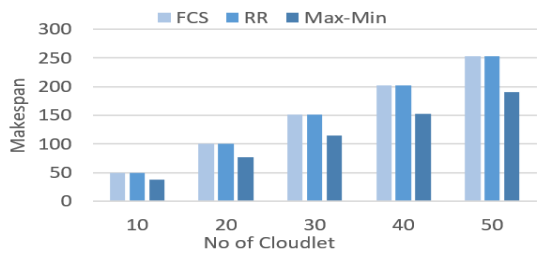

Figure 4. Makespan for each algorithm in Space Shared policy

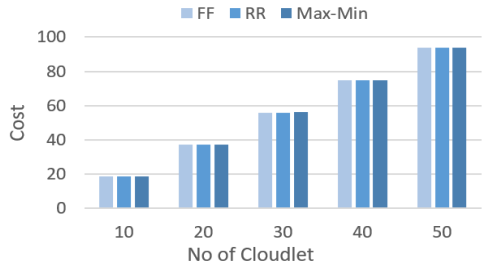

Figure 5. Cost for each algorithm in Space Shared policy

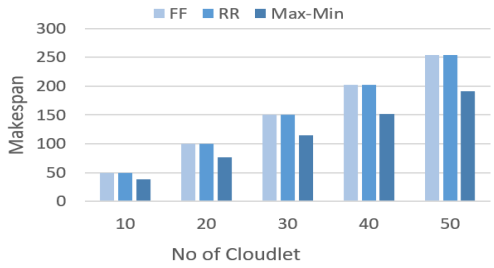

Figure 6. Makespan for each algorithm in Time Shared policy

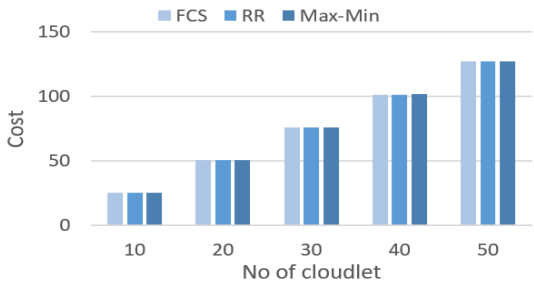

Figure 7. Cost for each algorithm in Time Shared policy

6. CONCLUSIONS

Cloud computing has become a very powerful way to process complicated and large jobs. Task scheduling algorithms in cloud computing plays an important role to reach a high-level of efficiency by allowing full utilization of resources to enhance the QoS. This study aims to compare three task scheduling algorithms called FCFS, RR, and Max-Min. The three algorithms are simulated using CloudSim tool to be analyzed and evaluated according to the performance metric of the VMs re-sources' cost, average time and makespan: While considering both resources allocation policies Time and Space Shared. The outcomes of our experiments show that Max-Min outperformed FCFS and RR in terms of makespan with the two policies. In addition, Max-Min achieved better average waiting time than FCFS and RR in terms of Space-Shared Policy. Furthermore, RR and FCFS recorded approximately similar performance as Max-Min based on the total cost in both policies. In general, the Max-Min achieved the best performance overall in all of the metric measures for both policies which makes Max-Min the most suitable scheduling technique for the proposed cloud computing environment. For future work, it will be very interesting to evaluate the three algorithms with additional metrics such as Throughput and Load Balancing, then show the effect of increasing the virtual machines' number on these algorithms' performance.

REFERENCES

[1] Ali, S.A., Alam, M. (2016). A relative study of task scheduling algorithms in cloud computing environment. In 2016 2nd International Conference on Contemporary Computing and Informatics (IC3I), Greater Noida, India, pp. 105-111. https://doi.org/10.1109/IC3I.2016.7917943

[2] Rani, B.K., Rani, B.P., Babu, A.V. (2015). Cloud computing and inter-clouds–types, topologies and research issues. Procedia Computer Science, 50: 24-29. https://doi.org/10.1016/j.procs.2015.04.006

[3] Anushree, B., Xavier, V.A. (2018). Comparative analysis of latest task scheduling techniques in cloud computing environment. In 2018 Second International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, pp. 608-611. https://doi.org/10.1109/ICCMC.2018.8487908

[4] Bittencourt, L.F., Goldman, A., Madeira, E.R., da Fonseca, N.L., Sakellariou, R. (2018). Scheduling in distributed systems: A cloud computing perspective. Computer Science Review, 30: 31-54. https://doi.org/10.1016/j.cosrev.2018.08.002

[5] Liu, L., Qiu, Z. (2016). A survey on virtual machine scheduling in cloud computing. In 2016 2nd IEEE International Conference on Computer and Communications (ICCC), pp. 2717-2721. https://doi.org/10.1109/CompComm.2016.7925192

[6] Kaur, A., Sengupta, J. (2017). Virtual machine scheduling using improved time shared policy in cloud computing. IJARCET-2017, 6(8): 1274-1277.

[7] Himthani, P.P., Ghanshyam, P., Dubey, P. (2019). Performance analysis of space shared scheduling and time shared scheduling in cloud sim. Int. J. Recent Dev. Eng. Technol., 8(2): 43-49.

[8] Chen, L., Li, X., Ruiz, R. (2018). Idle block based methods for cloud workflow scheduling with preemptive and non-preemptive tasks. Future Generation Computer Systems, 89: 659-669. https://doi.org/10.1016/j.future.2018.07.037

[9] Gollapalli, M., AlMetrik, M.A., AlNajrani, B.S., AlOmari, A.A., AlDawoud, S.H., AlMunsour, Y.Z., Abdulqader, M.M., Aloup, K.M. (2022). Task failure prediction using machine learning techniques in the google cluster trace cloud computing environment. Mathematical Modelling of Engineering Problems, 9(2): 545-553. https://doi.org/10.18280/mmep.090234

[10] Gupta, A., Garg, R. (2017). Load balancing based task scheduling with ACO in cloud computing. In 2017 International Conference on Computer and Applications (ICCA), Doha, Qatar, pp. 174-179. https://doi.org/10.1109/COMAPP.2017.8079781

[11] Jiang, Y., Shao, Z., Guo, Y., Zhang, H., Niu, K. (2015). Drscro: A metaheuristic algorithm for task scheduling on heterogeneous systems. Mathematical Problems in Engineering, 2015: 396582. https://doi.org/10.1155/2015/396582

[12] Soltani, N., Soleimani, B., Barekatain, B. (2017). Heuristic algorithms for task scheduling in cloud computing: A survey. International Journal of Computer Network & Information Security, 9(8): 16-22. https://doi.org/10.5815/ijcnis.2017.08.03

[13] Mathew, T., Sekaran, K.C., Jose, J. (2014). Study and analysis of various task scheduling algorithms in the cloud computing environment. In 2014 International

Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 658-664. https://doi.org/10.1109/ICACCI.2014.6968517

[14] Bahwaireth, K., Tawalbeh, L.A., Benkhelifa, E., Jararweh, Y., Tawalbeh, M.A. (2016). Experimental comparison of simulation tools for efficient cloud and mobile cloud computing applications. EURASIP Journal on Information Security, 2016(1): 1-14. https://doi.org/10.1186/s13635-016-0039-y

[15] Alhaidari, F., Balharith, T., Eyman, A.Y. (2019). Comparative analysis for task scheduling algorithms on cloud computing. In 2019 International Conference on Computer and Information Sciences (ICCIS), pp. 1-6. https://doi.org/10.1109/ICCISci.2019.8716470

[16] Sudheer, M.S., Reddy, K.G., Sree, P.K., Raju, V.P. (2017). An effective analysis on various scheduling algorithms in cloud computing. In 2017 International Conference on Inventive Computing and Informatics (ICICI), Coimbatore, India, pp. 931-936. https://doi.org/10.1109/ICICI.2017.8365274

[17] Calheiros, R.N., Ranjan, R., Beloglazov, A., De Rose, C. A., Buyya, R. (2011). CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and Experience, 41(1): 23-50. https://doi.org/10.1002/spe.995

[18] Hlaing, Y.T.H., Yee, T.T. (2019). Static independent task scheduling on virtualized servers in cloud computing environment. In 2019 International Conference on Advanced Information Technologies (ICAIT), pp. 55-59. https://doi.org/10.1109/AITC.2019.8920865

[19] Mazumder, A.M.R., Uddin, K.A., Arbe, N., Jahan, L., Whaiduzzaman, M. (2019). Dynamic task scheduling algorithms in cloud computing. In 2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA), pp. 1280-1286. https://doi.org/10.1109/ICECA.2019.8822020

[20] Fang, Y., Xiao, X., Ge, J. (2019). Cloud computing task scheduling algorithm based on improved genetic algorithm. In 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), pp. 852-856. https://doi.org/10.1109/ITNEC.2019.8728996

[21] Sarvabhatla, M., Konda, S., Vorugunti, C.S., Babu, M.N. (2017). A dynamic and energy efficient greedy scheduling algorithm for cloud data centers. In 2017 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), pp. 47-52. IEEE. https://doi.org/10.1109/CCEM.2017.9

[22] Sood, K., Jain, A., Verma, A. (2017). A hybrid task scheduling approach using firefly algorithm and gravitational search algorithm. In 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS), pp. 2997-3002. https://doi.org/10.1109/ICECDS.2017.8390005

[23] Gabi, D., Ismail, A.S., Zainal, A., Zakaria, Z. (2017). Solving task scheduling problem in cloud computing environment using orthogonal taguchi-cat algorithm. International Journal of Electrical & Computer Engineering (2088-8708), 7(3): 1489-1497. https://doi.org/10.11591/ijece.v7i3.pp1489-1497

[24] Elmougy, S., Sarhan, S., Joundy, M. (2017). A novel hybrid of Shortest job first and round Robin with dynamic variable quantum time task scheduling technique. Journal of Cloud computing, 6(1): 1-12. https://doi.org/10.1186/s13677-017-0085-0

[25] Kumari, R., Jain, A. (2017). An efficient resource utilization based integrated task scheduling algorithm. In 2017 4th International Conference on Signal Processing and Integrated Networks (SPIN), pp. 519-523. https://doi.org/10.1109/SPIN.2017.8050005

[26] Hicham, G.T., Chaker, E.A. (2016). Cloud computing cpu allocation and scheduling algorithms using cloudsim simulator. International Journal of Electrical & Computer Engineering (2088-8708), 6(4): 1866-1879. https://doi.org/10.11591/ijece.v6i4.10144

[27] Gamal El Din Hassan Ali, H., Saroit, I.A., Kotb, A.M. (2017). Grouped tasks scheduling algorithm based on QoS in cloud computing network. Egyptian Informatics Journal, 18(1): 11-19. https://doi.org/10.1016/j.eij.2016.07.002

[28] Agarwal, M., Srivastava, G.M.S. (2016). A genetic algorithm inspired task scheduling in cloud computing. In 2016 International Conference on Computing, Communication and Automation (ICCCA), pp. 364-367. https://doi.org/10.1109/CCAA.2016.7813746

[29] Ibrahim, M., Nabi, S., Hussain, R., Raza, M.S., Imran, M., Kazmi, S.A., Oracevic, A., Hussain, F. (2020). A comparative analysis of task scheduling approaches in cloud computing. In 2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID, Melbourne, VIC, Australia, pp. 681-684. https://doi.org/10.1109/CCGrid49817.2020.00-23

[30] Gollapalli, M., Alansari, A., Alkhorasani, H., Alsubaii, M., Sakloua, R., Alzahrani, R., Al-Hariri, M., Alfares, M., AlKhafaji, D., Al Argan, R., & Albaker, W. (2022). A novel stacking ensemble for detecting three types of diabetes mellitus using a Saudi Arabian dataset: Pre-diabetes, T1DM, and T2DM. Computers in Biology and Medicine, Vol. 147, 105757. https://doi.org/10.1016/j.compbiomed.2022.105757

[31] Gollapalli, M. (2015). Literature Review of Attribute Level and Structure Level Data Linkage Techniques. International Journal of Data Mining & Knowledge Management Process, 5(5), 01-20. https://doi.org/10.5121/ijdkp.2015.5501

[32] Pratap, R., Zaidi, T. (2018). Comparative study of task scheduling algorithms through cloudsim. In 2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO), pp. 397-400. https://doi.org/10.1109/ICRITO.2018.8748514

[33] Kumar, R.R., Jha, S.K., Garg, D., Vaishnav, S. (2018). Evaluation of load balancing algorithm using cloudsim. In 2018 3rd International Conference on Inventive Computation Technologies (ICICT), pp. 78-81. https://doi.org/10.1109/ICICT43934.2018.9034367

[34] Jambigi, M.V., Desai, V., Athanikar, S. (2018). Comparative analysis of different algorithms for scheduling of tasks in cloud environments. In 2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS), pp. 359-361. https://doi.org/10.1109/CTEMS.2018.8769155

[35] Madni, S.H.H., Abd Latiff, M.S., Abdullahi, M., Abdulhamid, S.I.M., Usman, M.J. (2017). Performance comparison of heuristic algorithms for task scheduling in IaaS cloud computing environment. PloS one, 12(5): e0176321. https://doi.org/10.1371/journal.pone.0176321

[36] Gollapalli, M., Alfaleh, A. (2022). An artificial intelligence approach for data modelling patients inheritance of sickle cell disease (SCD) in the eastern regions of Saudi Arabia. Mathematical Modelling of Engineering Problems, Vol. 9, No. 4, pp. 1079-1088. https://doi.org/10.18280/mmep.090426

[37] Gollapalli, M. (2022). Ensemble Machine Learning Model to Predict the Waterborne Syndrome. Algorithms, Vol. 15, No. 3, pp. 93. https://doi.org/10.3390/a15030093

[38] Gollapalli, M., Li, X., & Wood, I. (2013). Automated discovery of multi-faceted ontologies for accurate query answering and future semantic reasoning. Data & Knowledge Engineering, Vol. 87, pp. 405-424. https://doi.org/10.1016/j.datak.2013.05.005

NOMENCLATURE

AGA	Adaptive Genetic Algorithm
BVT	Borrowed-Virtual-Time
CIS	Cloud Information Service
DC	Data Center
FA	Firefly Algorithm
FCFS	First Come First Serve
GA	Genetic Algorithm
GSA	Gravitational Search Algorithm
GTS	Grouped Tasks Scheduling
LB-ACOA	Load Balancing Ant Colony Optimization Algorithm
LBIMM	Load-Balanced-Improved-Min-Min
MCT	Minimum-Completion-Time
MET	Minimum-Execution-Time
MESF	Most Efficient Server First
OTB-CSO	Orthogonal Taguchi-Based-Cat Swarm Optimization
PM	Physical Machines
QoS	Quality of Service
RASA	Resource-Aware-Scheduling-Algorithm
RALBA	Resource-Aware-Load-Balancing-Algorithm
RR	Round Robin
SJF	Shortest Job First
VM	Virtual Machines