Cost-Efficient Task Scheduling Algorithm for Reducing Energy Consumption and Makespan of Cloud Computing*

Research Article

Najme Mansouri[1] Reyhane Ghafari[2]

Abstract: In cloud computing, task scheduling is one of the most important issues that need to be considered for enhancing system performance and user satisfaction. Although there are many task scheduling strategies, available algorithms mainly focus on reducing the execution time while ignoring the profits of service providers. In order to improve provider profitability as well as meet the user requirements, tasks should be executed with minimal cost and without violating Quality of Service (QoS) restrictions. This study presents a Cost and Energy-aware Task Scheduling Algorithm (CETSA) intending to reduce makespan, energy consumption, and cost. The proposed algorithm considers the trade-off between cost, energy consumption, and makespan while considering the load on each virtual machine to prevent virtual machines from overloading. Experimental results with CloudSim show that the CETSA algorithm has better results in terms of energy consumption, waiting time, success rate, cost, improvement ratio, and degree of imbalance compared with MSDE, CPSO, CJS, and FUGE.

Keywords: Cloud, Task Scheduling, Cost, Energy consumption, Makespan

1. Introduction

Cloud computing is considered the backbone infrastructure IT industry that provides the virtualized pool of resources to end users dynamically [1]. Cloud computing embraced by several big IT companies such as Amazon, Google, IBM, Apple, Microsoft, and Oracle. With the increasing the number of tasks (i.e., requests) as well as the dynamic nature of cloud resources, it is very important to pay attention to objectives such as energy consumption, resource utilization, makespan, system performance, and so on [2]. Task scheduling is one of the main problems in the cloud system. The main idea in task scheduling is to assign tasks to Virtual Machines (VMs) that minimize waste of time and maximize performance [3]. Energy is one of the most important parameters that have a great impact on the performance of the cloud system because it decreases costs and is in accordance with the standard set-in green computing [4, 5]. Cloud data centers energy consumption is mostly contributed by computing resources (42.0%), storage and network equipment (19.2%), and cooling system (15.4%) [6] which is expected to increase by 12% annually [7]. In addition, such energy consumption contributes to 2% of global carbon emissions and is expected to increase in the future [8]. Moreover, about 15% of data center costs are related to electricity costs [9]. Therefore, energy efficiency is a very significant issue. Electricity costs and carbon emissions can be decreased with efficient hardware technologies as well as efficient cloud data center management operations. The inefficient resource management not only can increase the makespan but also incur extra energy consumption. Therefore, it is very important to consider the task scheduling problem to decrease energy consumption and at the same time improve makespan. In addition, profit is vital for business success, so we consider it a high priority parameter [10]. However, most previous works on achieving critical features that are essential for cloud computing, such as simultaneous consideration of makespan, energy consumption, and cost, failed and focused on only one parameter like makespan and did not take into account other parameters [11, 12]. Also, decreasing energy consumption increases makespan and leads to customer dissatisfaction. Therefore, considering these parameters at the same time, which increase both user satisfaction and the profit of the service provider, is very important. This study presents a cost and energy-aware task scheduling algorithm that makes a trade-off among cost, energy consumption, and makespan. In other words, the scheduler tries to execute as many tasks as possible with the main goal of increasing profit. This may conflict with the requirements of users like execution time.

The main contributions of the suggested algorithm can be summarized as follows:

1. The multi-objective scheduling problem is formulated in the form of a mathematical model and the objective functions are introduced;
2. Our proposed algorithm focuses on decreasing cost, energy consumption, and makespan simultaneously, while also preventing resource overload by considering the threshold value;
3. To confirm CETSA performance, we compare it with four well-known scheduling algorithms. Evaluation parameters are makespan, degree of imbalance, cost, success rate, average waiting time, improvement ratio, and total energy consumption.

The remaining of the paper is organized as follows: Section II explains the background of cloud computing and task scheduling. The review of related works on task scheduling algorithms is discussed in Section III. Section IV focuses on the description of the proposed task scheduling. The experiment settings and the results are presented in Section V. Finally, the conclusion is drawn in Section VI.

2. Background

In this section, we explain the background of cloud computing and task scheduling in detail.

* Manuscript received: April 24, 2021, Revised, September 9, 2021; Accepted: January 1, 2022.

[1]. Corresponding author. PhD in Computer Science, Shahid Bahonar University of Kerman, Kerman, Iran.
Email: najme.mansouri@gmail.com.
[2]. BSc. in Computer Science, Shahid Bahonar University of Kerman, Kerman, Iran

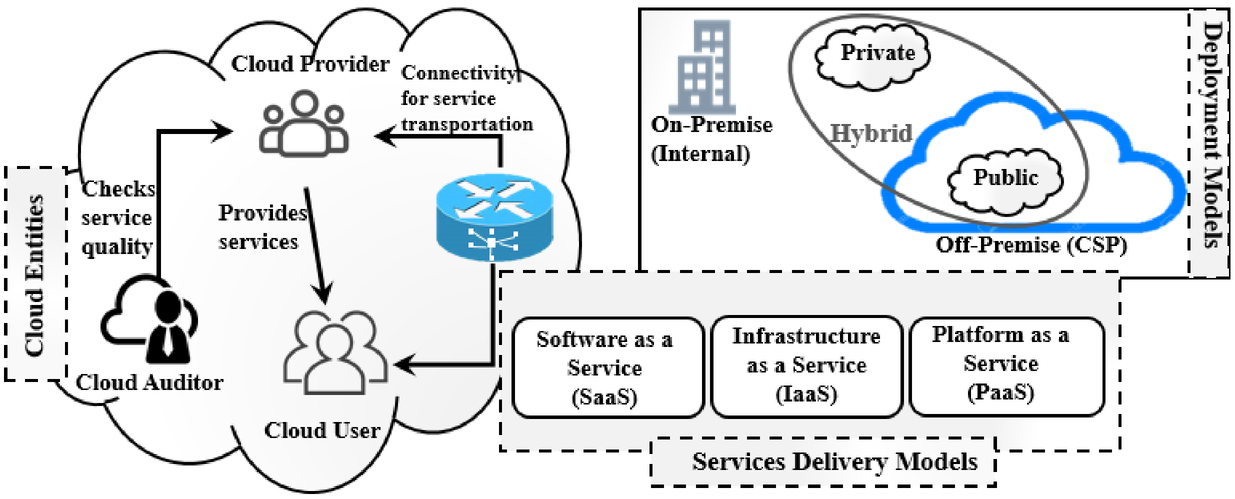


Figure 1. Cloud computing definition [17]

A. Cloud computing

Heterogeneous distributed computing systems are expanding to implement intensive scientific and computational applications. Cloud computing is an example of high-performance distributed computing that provides on-demand access to computing resources over the Internet with minimal management costs or service provider interaction [13, 14]. The cloud can become more popular by maximizing profits and minimizing energy consumption without hurting the service level requirements mentioned by users [15, 16]. Figure 1 summarizes the definition of cloud computing. Cloud computing presents three types of service models: Software as a Service (SaaS), Infrastructure as a Service (IaaS), and Platform as a Service (PaaS)). In addition, there are four cloud deployment models: public cloud, private cloud, community cloud, and hybrid clouds. In addition, cloud computing includes five basic features (broad network access, on-demand self-service, rapid elasticity, resource pooling, and measured service).

When several users request services from the cloud at the same time, we need the appropriate task scheduling algorithms to discover data and computational resources for task execution [18]. An efficient task scheduling algorithm not only affects the cloud performance, but also has a direct effect on the cost factor for cloud consumers, and providers who provide the necessary resources [19].

B. Task scheduling

The problem of task scheduling in the cloud means assigning the tasks to the most appropriate available VMs, according to the constraints [20, 21]. Figure 2 shows the task scheduling model in a cloud environment. Users submit tasks and tasks are transferred to the waiting queue. The data center broker is responsible for identifying and collecting all information about existing resources (VMs). The broker (i.e., scheduler) maps tasks to suitable VMs to meet the requirements of users (e.g., cost, energy consumption, and makespan, etc.). The broker obtains this information with the help of Cloud Information Services (CIS).

The primary kinds of scheduling mechanisms that exist in cloud computing can be divided into five categories: dynamic scheduling, static scheduling, heuristics scheduling, cloud service scheduling, and workflow scheduling [23]. The main purpose of task scheduling is to allocate tasks to resources in a way that optimizes one or more objectives (e.g., cost, energy consumption, makespan, etc.). To ensure the execution of the task at the desired time, it is essential to take into account makespan [24, 25]. For providers, the important factor is profit. While the cloud provider must make and distribute services to tenants, the goal of the cloud provider is to make an economic profit, i.e., monetary profit [26, 27]. In addition, for service providers, energy consumption is a significant parameter due to the environmental and economic impacts [28]. Excessive energy consumption increases the cost of electricity and also harms the environment. It is estimated that the data center releases 62 million tons of CO_2 into the atmosphere [29]. Therefore, it is very important to take some measures to decrease the widespread energy consumption of cloud data centers and improve energy efficiency when task scheduling [30, 31]. For companies that suggested large-scale cloud services (e.g., Google), energy bills related to companies' infrastructure are often an important part of their financial plans [32]. Decreasing energy consumption in large-scale cloud systems will profit cloud service providers, decrease user costs, and decrease CO_2 emissions [33, 34].

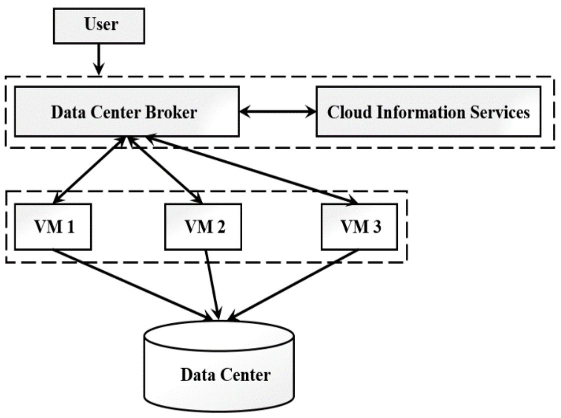


Figure 2. Task allocation in cloud system [22]

3. Review of related works

In recent years, the attention of researchers has been attracted to many problems facing cloud computing such as energy efficiency, resource management, and availability. All these challenges are significantly affected by using an efficient task scheduling strategy [35, 36]. There are many scheduling strategies based on different objectives to satisfy the requirements of users or enhance resource utilization.

Krishnaveni and Prakash [37] presented an Execution Time-based Sufferage Algorithm (ETSA) that not only reduces makespan but also improves resource utilization with a balanced load. The presented algorithm for all tasks calculates the sufferage value of completion time as well as the sufferage value of execution time. Then, it compares these two values for each task and assigns tasks to the available resources. The experimental results showed that the ETSA has a better performance in terms of makespan, resource utilization with a balanced load compared to other algorithms. Nevertheless, the energy efficiency of the proposed strategy is very low.

Chen et al. [38] investigated the use of the Whale Optimization Algorithm (WOA) [39] to solve the problem of optimizing multi-objective task scheduling in the cloud environment. At first, the authors proposed an advanced method named Improved WOA for Cloud task scheduling (IWC) to improve the search capabilities of WOA. Then, they used the IWC to schedule tasks in the cloud. The experimental results showed that the IWC method has better convergence speed and accuracy in search of the optimal task scheduling plans. In addition, IWC has better performance in terms of system load and the cost of system resource utilization for both small-scale and large-scale tasks compared to Particle Swarm Optimization (PSO) [40], Ant Colony Optimization (ACO) [41], and WOA [39]. The main weakness of the presented algorithm is that it needs more agents for searching the best resource.

Elaziz et al. [42] introduced a new hybrid algorithm for task scheduling in the cloud using a Moth Search Algorithm (MSA) [43] and a Differential Evolution (DE) [44] and called it MSDE. MSDE's goal is to decrease the makespan that needs to schedule many tasks on various VMs. The MSA is inspired by the behavior of moths insects and their relationships. Although MSA has good exploration capability, its exploitation capability is not very good. Therefore, the resaerchers combined the attributes of the Genetic Algorithm (GA) [45] and the DE as a local search strategy to increase the MSA exploitation capability. The authors used CloudSim [46] to simulate MSDE. The simulation results indicated that the MSDE could effectively schedule tasks and reduce makespan. Nevertheless, they did not discuss the cost or energy Quality of Service (QoS) parameters during the scheduling process.

Jacob and Pradeep [47] suggested a multi-objective task scheduling algorithm using a combination of Cuckoo Search (CS) [48] and PSO [40] and called it CPSO. The presented hybrid algorithm aimed to decrease the makespan, cost, and deadline violation rate. One of the advantages of CPSO is the rapid convergence and achievement of the optimal solution. The experimental results showed that CPSO performs better in terms of makespan, cost, and deadline violation rate compared to other algorithms. The main weaknesses of the presented algorithm are that it has a high possibility of overloading and the underutilization of resources.

Dubey et al. [49] suggested a modified HEFT (Heterogeneous Earliest Finish Time) [50] algorithm for task scheduling in the cloud environment. The suggested algorithm at first calculates the rank for all tasks based on the average of tasks on all the processors. Then, it assigns tasks to the suitable machine. The suggested modified HEFT algorithm efficiently distributes the load between the processors to decrease makespan. The simulation results showed that the suggested algorithm has better performance in terms of makespan and load balance compared to the HEFT [50] and CPOP [51] algorithms. But the conflicting factors such as time and cost are not discussed.

Mansouri and Javidi [52] offered the Cost-based Job Scheduling algorithm (CJS). The CSJ uses data-intensive and computation-intensive simultaneously. CJS considers various parameters such as processing power, data, and network features in the job allocation process. The main criterion for scheduling in CJS is the job characteristics along with the data and computation needs. The researchers defined a cost function based on computation cost, data transfer cost, network cost. They used CloudSim [46] to evaluate the performance of CJS. The simulation results showed that CJS performed better in terms of makespan, bandwidth consumption, and processor utilization. Nevertheless, energy consumption is not considered.

Shojafar et al. [53] introduced a hybrid algorithm using GA [45] and fuzzy theory [54] and named it FUGE. The presented algorithm uses fuzzy theory to calculate the fitness function. The FUGE's goal is to create the optimal load balance with respect to execution time and execution cost. The FUGE takes into account different parameters such as VM bandwidth, VM processing speed, VM memory, and job length for assigning tasks to suitable resources. The experimental results demonstrated that the FUGE has better performance in terms of execution time, execution cost, and average degree of imbalance compared to other algorithms. However, energy consumption and other important QoS have not been considered.

Zhao et al. [55] suggested a Power-Aware Task scheduling algorithm (PATS) to decrease cloud power consumption. First, the researchers developed a task scheduling model to predict cloud power consumption and formulate the problem of task scheduling. Then, they examined the effect of physical machine kind as well as task execution time on cloud power consumption and suggested a task scheduling algorithm for solving the formulated problem. The researchers used CloudSim [46] to evaluate PATS. The results showed that PATS decrease power consumption compared to other algorithms. But, sometimes VMs are overloaded due to improper mapping of tasks with machines.

Table 1 summarizes the scheduling algorithms discussed. It can be seen that the measures of energy, cost, and budget were not considered in some strategies despite their significant impact on the cloud service. Therefore, we propose a hybrid task scheduling algorithm that considers three conflicting objectives (i.e., cost, makespan, and energy consumption).

Table 1. Summary of reviewed task scheduling algorithms

	Year	Execution time	Makespan	Monetary cost	Resource utilization	Reliability	Energy consumption	Tool	Technique	Disadvantage
Krishnaveni and Prakash [37]	2019	✓	✓	×	✓	×	×	CloudSim	Sufferage value for execution time	- Does not consider cost and energy usage.
Chen et al. [38]	2019	✓	×	✓	✓	×	×	Matlab	Improved WOA	- Convergence speed and accuracy are low; - High scheduling overhead
Elaziz et al. [42]	2019	✓	✓	×	×	×	×	CloudSim	MSA and DE	- High time complexity, - Does not consider the usage of memory, the peak of the demand, and overloads.
Jacob and Pradeep [47]	2019	✓	✓	✓	×	×	×	CloudSim	CS and PSO	- Does not optimize QoS parameters such as energy consumption; - Cannot distribute the load uniformly.
Dubey et al. [49]	2018	✓	✓	×	×	×	×	CloudSim	The rank of tasks	- Does not continuously monitor nodes; - Does not compare with any state-of-art scheduling strategy
Mansouri and Javidi [52]	2019	✓	✓	×	×	×	×	CloudSim	Processing power, data, and network features	- Does not consider important factors such as energy consumption; - Resources may be overloaded or underutilization.
Shojafar et al. [53]	2014	✓	✓	✓	×	×	×	CloudSim	GA and fuzzy theory	- Does not consider VM energy consumption; - High monitoring overhead.
Zhao et al. [55]	2019	✓	×	×	×	×	✓	CloudSim	Physical machine power consumption of idle and busy states	- Does not include objectives such as network optimization and guarantee QoS.

4. Cost and Energy-Aware Task Scheduling Algoritm (CETSA)

This section is divided into three subsections: The basic concepts of the task scheduling problem are discussed in subsection *A*, subsection *B* explains the objective function, which intends to optimize various QoS parameters such as cost, makespan, and energy consumption, and subsection *C* describes our algorithm.

A. Task scheduling model

In the cloud system, tasks are scheduled based on the improvement of different QoS parameters. The task scheduling problem can be defined as how different tasks can be optimally assigned to a certain number of VMs. To model the task scheduling problem, it can be assumed that all submission tasks are independent, it is not possible to migrate tasks between VMs, and VMs are heterogeneous and also have various processing capabilities and power efficiency.

Consider a cloud system consisting of *m* VMs that are defined by set *V*, $V= \{V_1, V_2, ..., V_m\}$, where V_j indicates the *j-th* VM in the cloud system. Also, consider *n* independent tasks submitted by the users that are defined by set *T*, $T= \{T_1, T_2, ..., T_n\}$, where T_i indicates the *i-th* task in the task queue.

B. Objective functions

The purpose of objective function is to consider performance factors such as energy consumption, cost, makespan, and so on.

Because most previous scheduling algorithms have focused on user goals (e.g., execution time), the proposed algorithm considers the goal of a cloud service provider (e.g., energy consumption) as well as user satisfaction during the scheduling process. Thus, we define the objective function in such a way that it decreases cost, makespan, and energy consumption.

Cost: As a business service, task scheduling in a cloud system should not only be an efficient scheduler, but it should also reduce economic costs. One of the primary goals of service providers is to maximize revenue. Scheduling that reduces costs without violating QoS increases both user and service provider satisfaction. To measure the allocation cost, each use of resources (e.g., processing element cost, memory cost, etc.) must be calculated. The Cost (*C*) of processing tasks on V_j is calculated based on Equation 1 [56]:

$$C_{V_j} = \sum_{j=1}^{m} sum(V_j) \times (V_{PEj} \times V_{ramj} \times V_{bwj}) \qquad (1)$$

where V_{PEj}, V_{ramj}, and V_{bwj} indicate the cost of processing element, memory, and bandwidth performance of VMs, respectively.

Makespan: One of the most popular optimization factors of scheduling is minimizing makespan because it improves performance and brings user satisfaction because users tend to execute their applications quickly. Makespan (*MS*) is the maximum amount of execution time of all VMs and mathematically it can be expressed as Equation 2:

$$MS = Max(E_j) \quad 1 \le j \le m \qquad (2)$$

where E_j indicates the execution time of *j-th* VM and it is calculated based on the decision variable U_{ij} [57].

$$U_{ij} = \begin{cases} 1, & if\ T_i\ is\ assigned\ to\ V_j \\ 0 & if\ T_i\ is\ not\ assigned\ to\ V_j \end{cases} \qquad (3)$$

$$E_j = \sum_{i=1}^{n} U_{ij} \times TC_{ij} \qquad (4)$$

where TC_{ij} indicates the *i-th* $(1 \le i \le n)$ task completion time in *j-th* $(1 \le j \le m)$ VM and it is calculated based on Equation 5 [57]:

$$TC_{ij} = \frac{L_i}{PE_j} \qquad (5)$$

where L_i indicates the length of the *i-th* task (i.e., length of the task is defined in terms of the number of instructions (Millions of instructions)) and PE_j indicates the processing capability of *j-th* VM.

Energy consumption: When talking about energy consumption, green computing technology should be included as well, because as energy consumption decreases, pollution also decreases. Benefits of reducing energy consumption include minimizing performance loss, maximizing profits, minimizing CO_2 emissions, minimizing power consumption, maximizing resource utilization. Therefore, more attention should be paid to energy consumption in the cloud system to make cloud services environmentally friendly. Each VMs has two states when created through the host or physical machine: 1) active state, and 2) idle states. Total energy consumption is defined as the sum of energy consumed in active and idle states. The total Energy Consumption (*EC*) is calculated by Equation 6 [57]:

$$EC = \sum_{j=1}^{m} ([E_j \times \alpha_j + (MS - E_j)\beta_j]) \times PE_j \qquad (6)$$

where E_j and *MS* are obtained according to Equation 4 and Equation 2, respectively, and represent execution time and makespan, α_j indicates joules/Millions of instruction consumed by *j-th* VM in the active state and β_j indicates joules/Millions of instruction consumed by *j-th* VM in the idle state, and $(MS - E_j)$ indicates the amount of time will remain idle by *j-th* VM.

C. The CETSA Schema

Load balancing of tasks on the cloud is one of the most important aspects of cloud computing since overloaded resources lead to SLA violation and system failure. VMs should run in parallel and execute the task as quickly as possible. VMs can carry a load of more than one task at the same time. Therefore, load balancing prevents a situation where some nodes are overloaded while others are idle or have a little task to do. To check whether VMs need load balancing or not, the load and capacity of each VM must first be found based on Equation 7 [58] and Equation 8 [58]:

$$V_{load} = (N \times L) / V_{Mips} \qquad (7)$$

where *N* indicates the number of tasks allocated, *L* indicates the length of the task and V_{Mips} is Million Instructions Per Second (MIPS) of the VM.

$$V_{capacity} = PE_{Number} \times PE_{Mips} \times V_{bw} \qquad (8)$$

where PE_{Number} indicates the number of processing elements in the current VM and PE_{Mips} is MIPS capability of a processing element of VM and V_{bw} is bandwidth linked with VM.

Then, the Processing Time (*PT*) of the current VM can be calculated based on Equation 9 [58]:

$$PT\_V_i = V_{load} / V_{capacity} \qquad (9)$$

Finally, the standard deviation (σ) of the load is calculated according to Equation 10:

$$\sigma = \sqrt{1/M \sum_{i=1}^{M} (X_i - X)^2} \qquad (10)$$

where X_i is the processing time of the current VM, *X* is the average processing time of VM, and *M* belongs to the VM in

VM set.

Now by maintaining a threshold value and comparing it with the (σ), it can be understood whether the given VMs need load balancing or not. When users submit the tasks, we perform the following steps to schedule the tasks according to the CETSA algorithm:

1. Select the tasks one by one.
2. According to the selected task, we calculate the cost, makespan, and energy consumption based on Equation 1, Equation 2, and Equation 6, respectively.
3. Energy, cost, and makespan are important parameters in task scheduling. The scheduling algorithm assigns tasks to VMs that consume less energy because reducing energy consumption improves reliability, productivity, and availability. Moreover, reducing energy consumption not only decreases energy costs but also aids to protect the natural environment by reducing carbon emissions. Efficient resource management is the key to balancing cloud performance and costs while maintaining services availabile. In addition, makespan is one of the main parameters in assigning tasks to VMs because makespan must be minimized to measure the performance of each algorithm as well as to satisfy users. Makespan represents the time taken from the moment a user sends the request to the completion of the last task unit. Makespan contains both processing time and waiting time. Cost is also another important parameter that the task scheduling algorithm must take into account and schedule tasks on VMs with a minimum cost to increase the provider's profit in addition to user satisfaction. Given the above issues, because these parameters are less considered simultaneously, we considered all these parameters simultaneously in one formula, so that we normalized the values obtained for cost, energy consumption, and makespan in step 2. Because we wanted to minimize these parameters, we put them in the fraction denominator and express them as one formula. Equation 11 shows the merit of each VM based on cost, makespan, and energy consumption. The higher the merit, the smaller the fraction denominator. In other words, a higher value of merit indicates less makespan, cost, and energy consumption. Therefore, to use the VMs efficiently, the proposed algorithm seeks to find VMs with higher merit values. The calculation of the merit for all VMs according to cost, energy consumption, and makespan is as follows:

$$Merit = \frac{1}{C + MS + EC} \tag{11}$$

4. Create a list and sort the VMs in descending order based on the merit calculated in the previous step and place them in the list.
5. Select the first VM from the list to assign the task.
6. According to Equation 10 and considering the threshold of 90%:
    6.1. If the (σ) is less than 90%, there is no overloaded and assign the task to the selected VM.
    6.2. If the (σ) is more than 90%, i.e., the VM is overloaded and the selected VM is removed from the list and select the next VM of the sorted list and go to step 6.

We do this process until all tasks are assigned to VMs. Figure 3 shows the flowchart of the CETSA algorithm.

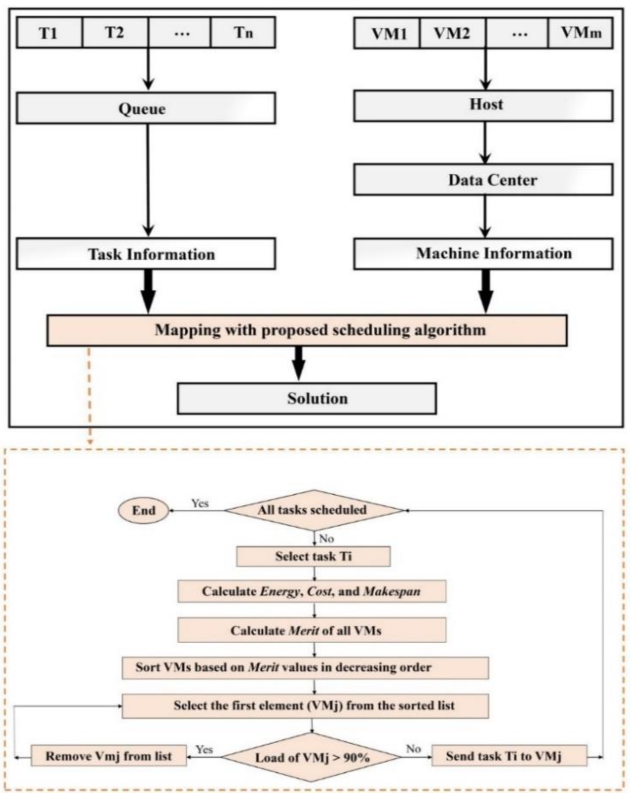

T1
T2
…
Tn
VM1
VM2
…
VMm
Queue
Host
Data Center
Task Information
Machine Information
Mapping with proposed scheduling algorithm
Solution
End
Yes
All tasks scheduled
No
Select task Ti
Calculate Energy, Cost, and Makespan
Calculate Merit of all VMs
Sort VMs based on Merit values in decreasing order
Select the first element (VMj) from the sorted list
Remove Vmj from list
Yes
Load of VMj > 90%
No
Send task Ti to VMj

Figure 3. The proposed task scheduling framework

5. Performance analysis

In this section, we explain the environment setup and then evaluate the performance of the proposed scheduling (CETSA) in comparison with the MSDE [42], CPSO [47], CJS [52], and FUGE [53].

A. The environment setup

The performance of the presented task scheduling algorithm is analyzed based on the simulation results. The cloud computing experiment was performed through the CloudSim 3.0.3 simulator. CloudSim [46] is one of the most popular simulators in the cloud environment. The simulator can compare the performance of various task scheduling techniques. This helps to adjust performance bottlenecks before execution, thus saving on costs. The steps for running CloudSim are as follows [59]:

1. Initialize CloudSim.
2. Create a data center and service broker to coordinate and assign resources and create VMs.
3. Add the VM to the list of VMs and send the list of VMs to the service broker.
4. Create a cloud task set, add the cloud task cloudlet to the cloud task set, and send it to the service broker.
5. Start the simulation and print the results after the simulation.

Figure 4 shows the CloudSim structure. CloudSim consists of three main layers. The first layer is user code that provides configuration factors such as the number of VMs, number of users, etc., the second layer manages the execution of key elements such as cloudlets and resources when simulating, and the third layer, which is the CloudSim core simulation engine, models the queuing and communication between the components. The simulation environment of the experiments has been presented in Table 2.

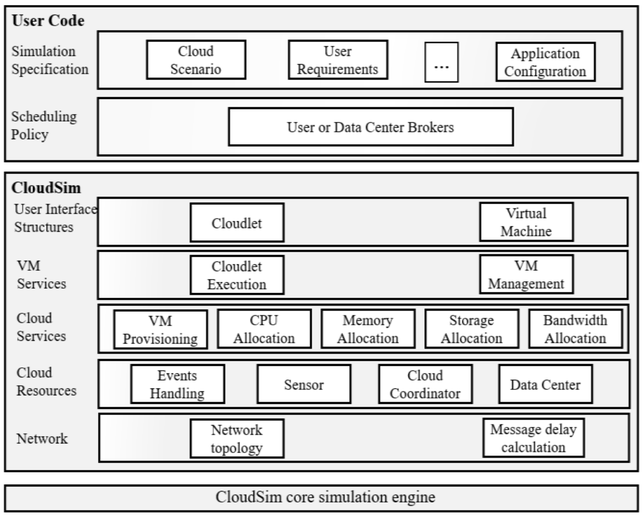


Figure 4. CloudSim structure [60]

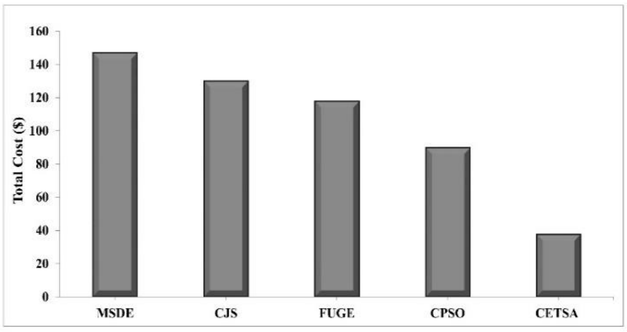
Table 2. Simulation parameters

Parameters	Value
Number of data centers	10-50
Total number of VMs	100
MIPS of processing element	500-4500
Number of processing element per VM	1-4
VM memory (RAM)	15-35 (GB)
Total number of tasks	100-500
Length of task	100-2000 (MI)
Storage cost	\$0.20 to \$0.30/GB
Processing cost	\$1.25 to \$2.25/10^9 MI
Transfer cost	\$0.25/GB
Load threshold	90%

B. Discussion of results

The purpose of the CETSA algorithm is to schedule the tasks in a way that minimizes costs. There is always a conflict between time and cost. The presented algorithm considers the trade-off issue between cost and time. Figure 5 shows the total cost of MSDE, CJS, FUGE, CPSO, and CETSA. It is observed that CETSA achieves 74%, 71%, 68%, and 58% lower cost than MSDE, CJS, FUGE, and CPSO, respectively. Therefore, CETSA obtains higher revenue and profit than other algorithms. This is because CETSA uses energy efficiently and provides better scheduling for various tasks. Consequently, it reduces the cost of cloud providers through energy efficiency.

Cloud data centers consume a lot of electrical energy, which increases the cost and CO_2 emissions day by day. Reducing energy consumption is one of the most challenging issues in cloud computing. Figure 6 shows the comparison of performance in terms of energy consumption. Obviously, with increasing the number of tasks, energy consumption also increases. Compared to all algorithms, the CETSA algorithm consumes the least energy for all cases and performs better in energy saving.



Figre 5. Total cost

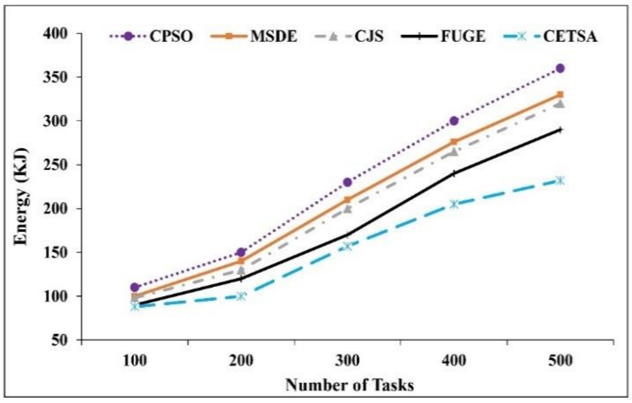


Figure 6. Total energy consumption

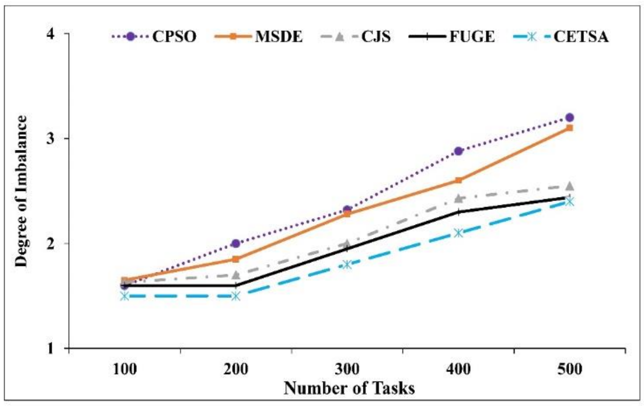
The energy consumption by CETSA is 2-20% less than that of FUGE for 100 through 500 number of tasks, respectively. The energy consumption minimization by CETSA is 10-28% less than that of CJS for 100 through 500 number of tasks, respectively. This is because the FUGE algorithm and the CJS algorithm do not take into account energy consumption and only focus on cost.

In Figure 7 and Figure 8, the performance of the proposed algorithm is tested based on the degree of imbalance and compared with other algorithms. The degree of imbalance measures the imbalance among VMs. It explains the amount of load distribution among the VMs. It can be calculated by Equation 12 [61]:

$$D_i = \frac{L_{Tasks}}{PE_{Number} \times PE_{Mips}} \qquad (12)$$

where L_{Tasks} represents the total length of tasks that are assigned to VM_j, PE_{Number} indicates the number of processing elements, and PE_{Mips} is MIPS of the processing element.

The small value of the degree of imbalance indicates that the load of the system is more balanced and efficient. Figure 7 shows the degree of imbalance of each algorithm with the number of tasks from 100 to 500. It can be seen from Figure 7 that CETSA leads to the improvement of the performance in terms of VMs load balancing. FUGE and CJS perform better than MSDE and CPSO in minimizing the degree of imbalance, since they consider task and machine capability during scheduling decisions. Figure 8 shows the degree of imbalance between CETSA and other algorithms, where the number of tasks is kept constant while the number of data centers varies between 10 and 50 data centers. Figure 8 shows that CETSA has better performance especially when computing resources are limited because CETSA considers the load of the VM for task assignment.



Figre 7. Degree of imbalance for the different number of tasks

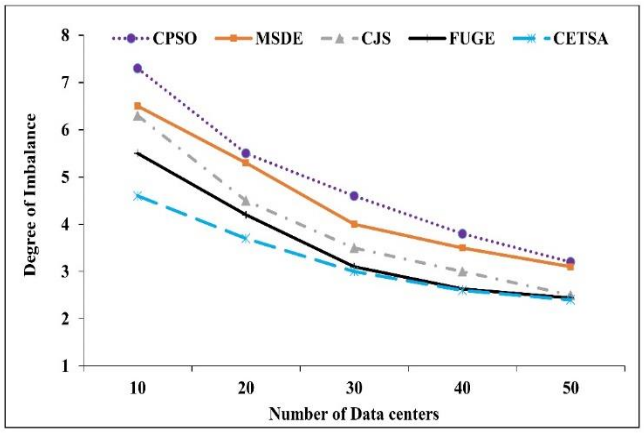


Figure 8. Degree of imbalance for the different number of data centers

Makespan or the completion time is the time when the execution of the last task is finished. Measuring the makespan is significant because it helps to minimize energy consumption and meet the deadline of the task. Figure 9 shows the makespan comparison between the CETSA algorithm and other algorithms. Assume that the number of data centers is fixed and the number of tasks is gradually increased from 100 to 500 tasks. The Y axis shows the effect on makespan while increasing the number of tasks. According to Figure 9, the makespan increases over the increasing number of tasks. CETSA performs better than other algorithms in minimizing the makespan. It was observed that the CETSA algorithm decreases the makespan up to 26% in comparison to CPSO, 23% to MSDE, 14% to CJS, and 9% to FUGE in the case of 500 tasks allocated.

Figure 10 shows the makespan of various task scheduling algorithms as the number of data center changes. The effect of increasing the number of data centers indicates that makespan reduces linearly. From the results, we can see that the CETSA has a lower makespan than all the other scheduling algorithms, where CETSA is notably better than CPSO, MSDE, and CJS, but slightly better than the FUGE algorithm. The reason is that CETSA takes into account the execution time.

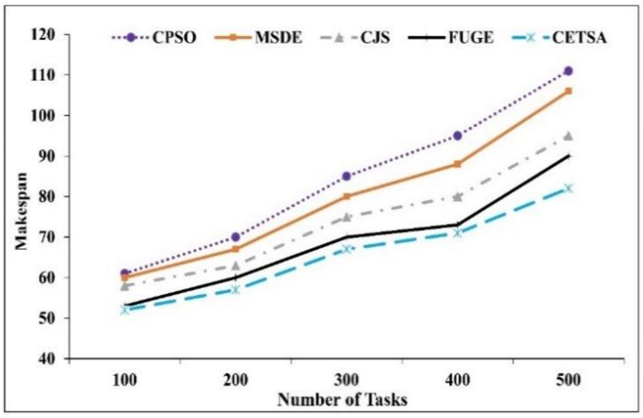


Figure 9. Makespan for the different number of tasks

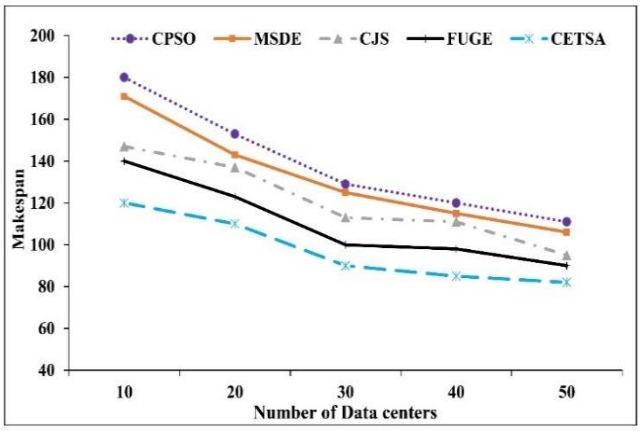


Figure 10. Makespan of the different number of data centers

Figure 11 and Figure 12 show the performance of CETSA in terms of improvement ratio compared to other algorithms. Improvement ratio indicates the algorithm efficiency based on the execution time reduction. It is defined as Equation 13 [62]:

$$IR_j\% = \frac{\sum_{i=1, i\neq j}^{n} E_i - E_j}{\sum_{i=1, i\neq j}^{n} E_i} \times 100 \tag{13}$$

where E_i shows the execution time of the *i-th* algorithm. The proposed algorithm has the best improvement ratio compared to the CPSO, MSDE, CJS, and FUGE. These results are because the improvement ratio is closely related to the execution time and the CETSA algorithm endeavors to achieve the shortest total execution time by considering important parameters such as processing time.

Figure 13 and 14 show the success rate for different task scheduling algorithms. The success rate is defined as the ratio of the number of successfully executed tasks to the total number of tasks submitted to the system. We can observe that the CETSA algorithm gives highier success rate than other scheduling algorithms. This is because the CETSA algorithm endeavors to specify the most suitable VM that can minimize cost, energy consumption, and makespan parameters, simultaneously. The success rate of FUGE is 2-5% smaller than that of CETSA for 100 through 500 number of tasks, respectively. Figure 14 shows clearly that the proposed algorithm has a better success rate in comparison with other scheduling algorithms. Besides, the CPSO algorithm has the lowest success rate values.

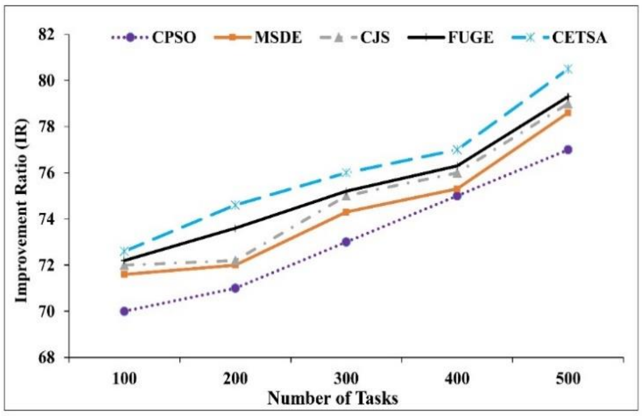


Figure 11. Improvement ratio of the different number of tasks

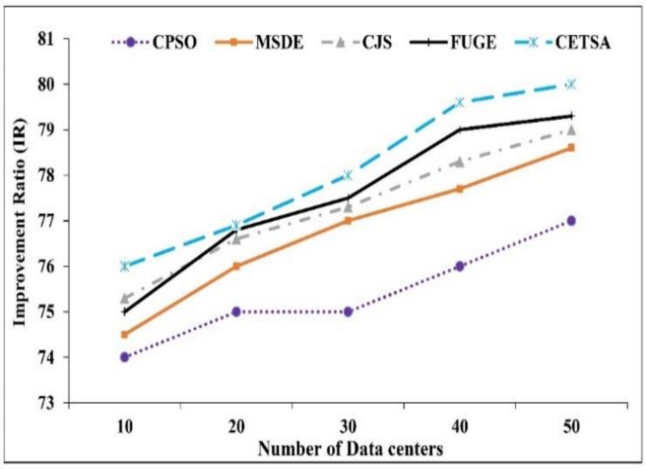


Figure 12. Improvement ratio of the different number of data centers

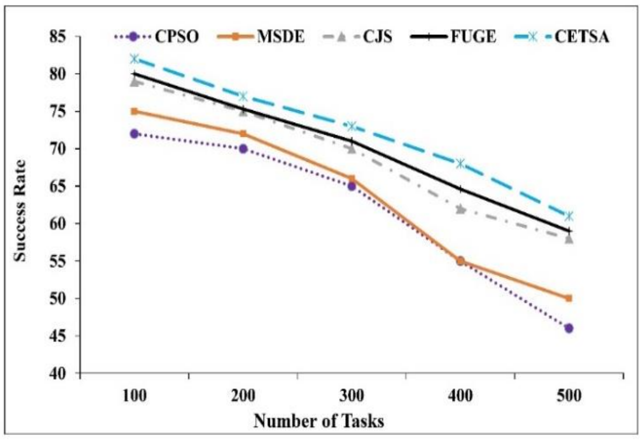


Figure 13. The success rate of the different number of tasks

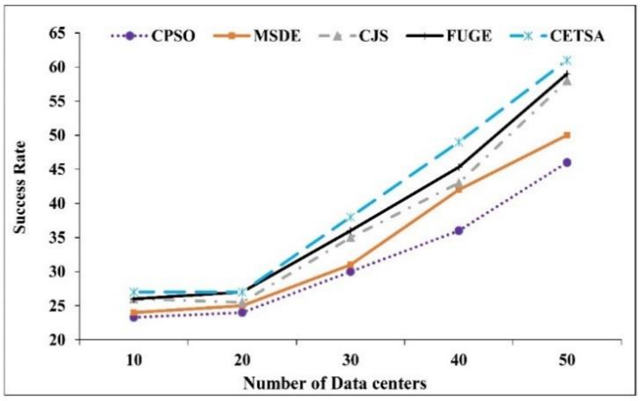


Figure 14. The success rate of the different number of data centers

Figure 15 shows the average waiting time needed for each task, which is one of important factors in analyzing algorithms. As Figure 15 shows, the waiting time for each algorithm increases as the number of tasks increases. Compared to other task scheduling algorithms, CETSA provides the lowest average waiting time. Waiting time is determined by Equation 14 [56]:

$$WT_i = \max_{j=1}^{m} \sum_{i=1}^{sum(v_j)} time_{ij} \tag{14}$$

where $time_{ij}$ indicates the time required for task T_i to execute on the allocated VM V_j and $sum(v_j)$ indicates the total number of tasks allocated to VM V_j.

The waiting time minimization by CETSA is 2-11% less than that of FUGE for 100 through 500 number of tasks, respectively. Figure 16 shows the average waiting time of each algorithm with the number of data centers varying from 10 to 50. It can be seen from Figure 16 that CETSA leads to improvement of the performance in terms of average waiting time. Besides, the calculated waiting time of CETSA algorithm is approximately 33% less than that of CPSO, 31% than that of MSDE, 19% than that of CJS, and 11% than that of FUGE in 50 data centers. This could be because the CETSA considers the processing time, and this can reduce the average waiting time.

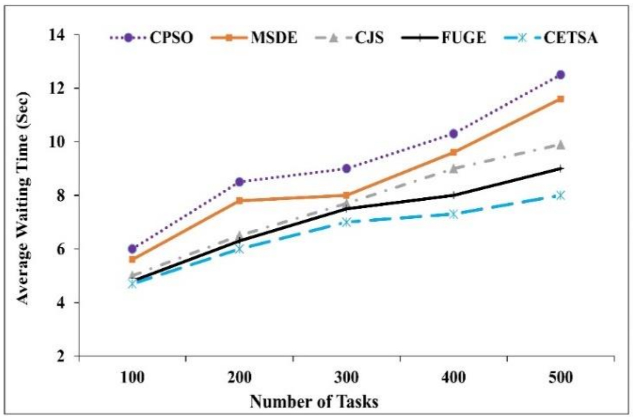


Figure 15. Average waiting time of the different number of tasks

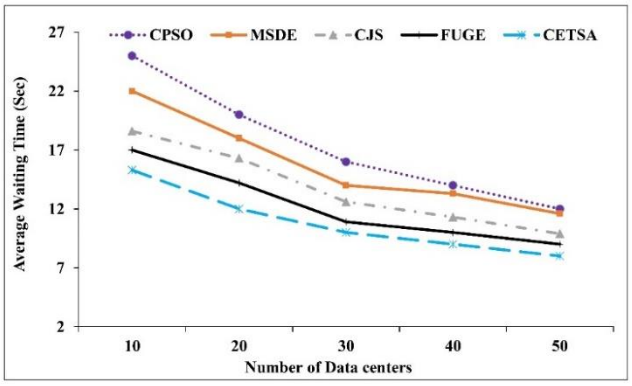


Figure 16. Average waiting time of the different number of data centers

6. Conclusion

Cloud computing is a complex large-scale system, consisted of thousands of cloud resource nodes and communication links. Finding reliable VM resources and scheduling tasks among suitable resources for successful execution is one of the major challenges in the cloud environment. Although there are many task scheduling algorithms in the cloud, most scheduling algorithms focus on user satisfaction and parameters such as execution cost or makespan. A small number of scheduling algorithms consider provider aims and parameters such as cost and energy consumption simultaneously. This study proposed a cost and energy-aware task scheduling algorithm that efficiently schedules tasks in cloud resources and optimizes the parameters such a cost, energy consumption, and makespan. To evaluate the performance of the CETSA algorithm, the CloudSim simulator was used and compared with MSDE, CPSO, CJS, and FUGE algorithms. The experimental results show the efficiency of the proposed algorithm. The results demonstrate that CETSA reduces cost, energy consumption, makespan, degree of imbalance, and average waiting time while improving the success rate and improvement ratio in a more efficient way in comparison with other algorithms. In future work, we will consider other QoS factors such as reliability, availability, and SLA violation for better cloud services. Besides, we will use meta-heuristic algorithms to improve performance and get better results.

7. Reference

[1] N. Mansouri and M. M. Javidi., "A review of data replication based on meta-heuristics approach in cloud computing and data grid", *Soft Computing*. vol. *24, pp. 19*, 2020.

[2] R. Medara, R. S. Singh, and Amit, "Energy-aware workflow task scheduling in clouds with virtual machine consolidation using discrete water wave optimization", *Simulation Modelling Practice and Theory*. vol. *110*, 2021.

[3] E. H. Houssein, A. G. Gad, Y. M. Wazery, and P. N. Suganthan., "Task scheduling in cloud computing based on meta-heuristics: Review, Taxonomy, open challenges, and future trends", *Swarm and Evolutionary Computation*, vol. *62*, pp. 1–41, 2021.

[4] R. Medara and R. S. Singh., "Energy efficient and reliability aware workflow task scheduling in cloud environment", *Wireless Personal Communications*, 2021.

[5] M. Sharma and R. Garg., "An artificial neural network based approach for energy efficient task scheduling in cloud data centers", *Sustainable Computing: Informatics and Systems*. vol. *26*, 2020.

[6] L. A. Barroso, J. Clidaras, and U. Hölzle., "The datacenter as a computer: An introduction to the design of warehouse-scale machines", *Synthesis lectures on computer architecture*. vol. 8(3), pp. 1–154, 2013.

[7] A. Uchechukwu, K. Li, and Y. Shen., "Energy consumption in cloud computing data centers", *International Journal of Cloud Computing and Services Science (IJ-CLOSER)*. vol. 3(3), pp. 31–48, 2014.

[8] B. Whitehead, D. Andrews, A. Shah, and G. Maidment., "Assessing the environmental impact of data centres part 1: Background, energy use and metrics", *Building and Environment*. vol. 82, pp. 151–159, 2014.

[9] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel. (2008). The cost of a cloud: research problems in data center networks. *ACM SIGCOMM Computer Communication Review*. vol. 39(1), pp. 68–73, 2008.

[10] A. Khelifa, T. Hamrouni, R. Mokadem, and F. Ben Charrada. (2020). SLA-aware task scheduling and data replication for enhancing provider profit in clouds. *Procedia Computer Science*. vol. *176*, pp. 3143–3152, 2020.

[11] M. Lavanya, B. Shanthi, and S. Saravanan., "Multi objective task scheduling algorithm based on SLA and processing time suitable for cloud environment", *Computer Communications*, vol. 151, pp. 183–195, 2020.

[12] A. Asghari, M. K. Sohrabi, and F. Yaghmaee., "Online scheduling of dependent tasks of cloud's workflows to enhance resource utilization and reduce the makespan using multiple reinforcement learning-based agents", *Soft Computing*, vol. *24(21)*, 2020.

[13] N. Mansouri, R. Ghafari, and B. M. H. Zade., "Cloud computing simulators: A comprehensive review", *Simulation Modelling Practice and Theory*, vol. *104*, pp. 1–101, 2020.

[14] N. Mansouri, B. M. H. Zade, and M. M. Javidi., "A multi-objective optimized replication using fuzzy based self-defense algorithm for cloud computing", *Journal of Network and Computer Applications*, vol. 171, pp. 1–33, 2020.

[15] N. K. Biswas, S. Banerjee, U. Biswas, and U. Ghosh., "An approach towards development of new linear regression prediction model for reduced energy consumption and SLA violation in the domain of green cloud computing", *Sustainable Energy Technologies and Assessments*, vol. 45, 2020.

[16] Z. Tong, X. Deng, H. Chen, and J. Mei., "DDMTS: A novel dynamic load balancing scheduling scheme under SLA constraints in cloud computing", *Journal of Parallel and Distributed Computing*, vol. 149, pp. 138–148, 2021.

[17] D.A. Shafiq, N.Z. Jhanjhi, A. Abdullah., "Load balancing techniques in cloud computing environment: A review", *Journal of King Saud University*, 2021.

[18] K. Dubey and S. C. Sharma., "A hybrid multi-faceted task scheduling algorithm for cloud computing environment", *International Journal of System*

Assurance Engineering and Management, 2021.

[19] A. Khelifa, T. Hamrouni, R. Mokadem, and F. Ben Charrada., "Combining task scheduling and data replication for SLA compliance and enhancement of provider profit in clouds", *Applied Intelligence*, 2021.

[20] G. Sreenivasulu and I. Paramasivam., "Hybrid optimization algorithm for task scheduling and virtual machine allocation in cloud computing", *Evolutionary Intelligence*, 2020.

[21] B. Wang, C. Wang, W. Huang, Y. Song, and X. Qin., "Security-aware task scheduling with deadline constraints on heterogeneous hybrid clouds", *Journal of Parallel and Distributed Computing*, vol. *153*, pp. 15–28, 2021.

[22] A. Pradhan, S. K. Bisoy, and A. Das. (2020). A survey on PSO based meta-heuristic scheduling mechanism in cloud computing environment. *Journal of King Saud University-Computer and Information Sciences*, 2020.

[23] R. Jia, Y. Yang, J. Grundy, J. Keung, and L. Hao. (2021). A systematic review of scheduling approaches on multi-tenancy cloud platforms. *Information and Software Technology*. vol. *132*, pp. 1–55, 2021.

[24] M. Hosseinzadeh, M. Y. Ghafour, H. K. Hama, B. Vo, and A. Khoshnevis., "Multi-objective task and workflow scheduling approaches in cloud computing: a comprehensive review", *Journal of Grid Computing*, vol. 18(3), pp. 327–356, 2020.

[25] P. Han, C. Du, J. Chen, F. Ling, and X. Du., "Cost and makespan scheduling of workflows in clouds using list multiobjective optimization technique", *Journal of Systems Architecture*, vol. *112*, 2021.

[26] N. Rizvi, R. Dharavath, and D. R. Edla., "Cost and makespan aware workflow scheduling in IaaS clouds using hybrid spider monkey optimization", *Simulation Modelling Practice and Theory*, vol. 110, 2021.

[27] C. K. Swain, B. Gupta, and A. Sahu., "Constraint aware profit maximization scheduling of tasks in heterogeneous datacenters", *Computing*, vol. 102(10), pp. 2229–2255, 2020.

[28] M. Sohaib Ajmal, Z. Iqbal, F. Zeeshan Khan, M. Bilal, and R. Majid Mehmood., "Cost-based energy efficient scheduling technique for dynamic voltage and frequency scaling system in cloud computing", *Sustainable Energy Technologies and Assessments*, 2021.

[29] M. Hussain, L.-F. Wei, A. Lakhan, S. Wali, S. Ali, and A. Hussain., "Energy and performance-efficient task scheduling in heterogeneous virtualized cloud computing", *Sustainable Computing: Informatics and Systems*, vol. 30, pp. 1–12, 2021.

[30] M. Sharma and R. Garg., "HIGA: Harmony-inspired genetic algorithm for rack-aware energy-efficient task scheduling in cloud data centers", *Engineering Science and Technology, an International Journal*, vol. 23(1), pp. 211–224, 2020.

[31] D. Ding, X. Fan, Y. Zhao, K. Kang, Q. Yin, and J. Zeng., "Q-learning based dynamic task scheduling for energy-efficient cloud computing", *Future Generation Computer Systems*, vol. 108, pp. 361–371, 2020.

[32] D. K. Shukla, D. Kumar, and D. S. Kushwaha., "Task scheduling to reduce energy consumption and makespan of cloud computing using NSGA-II", *Materials Today: Proceedings*, 2021.

[33] A. A. Khan and M. Zakarya., "Energy, performance and cost efficient cloud datacentres: A survey", *Computer Science Review*, vol. 40, 2021.

[34] H. Yuan, H. Liu, J. Bi, and M. Zhou., "Revenue and energy cost-optimized biobjective task scheduling for green cloud data centers", *IEEE Transactions on Automation Science and Engineering*, vol. 18(2), pp. 817–830, 2020.

[35] W. Jing, C. Zhao, Q. Miao, H. Song, and G. Chen., "QoS-DPSO: QoS-aware task scheduling for cloud computing system", *Journal of Network and Systems Management*, vol. 29(1), pp.1 –29, 2020.

[36] J. Kumar Samriya and N. Kumar., "An optimal SLA based task scheduling aid of hybrid fuzzy TOPSIS-PSO algorithm in cloud environment", *Materials Today: Proceedings*, 2020.

[37] H. Krishnaveni and V. S. J. Prakash., "Execution time based sufferage algorithm for static task scheduling in cloud", *Presented at Advances in Big Data and Cloud Computing*. pp. 61–70, 2019.

[38] X. Chen, L. Cheng, C. Liu, Q. Liu, J. Liu, Y. Mao, J. Murphy., "A woa-based optimization approach for task scheduling in cloud computing systems", *IEEE Systems Journal*, vol. 14(3), pp. 3117–3128, 2020.

[39] S. Mirjalili and A. Lewis., "The whale optimization algorithm", *Advances in engineering software*, vol. 95, pp. 51–67, 2016.

[40] J. Kennedy and R. Eberhart., "Particle swarm optimization", *Presented at Proceedings of ICNN'95-international conference on neural networks*, vol. 4, pp. 1942–1948, 1995.

[41] M. Dorigo, V. Maniezzo, and A. Colorni., "Ant system: optimization by a colony of cooperating agents", *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 26(1), pp. 29–41, 1996.

[42] M. Abd Elaziz, S. Xiong, K. P. N. Jayasena, and L. Li., "Task scheduling in cloud computing based on hybrid moth search algorithm and differential evolution", *Knowledge-Based Systems*, vol. 169, pp. 39–52, 2019.

[43] G.-G. Wang., "Moth search algorithm: a bio-inspired metaheuristic algorithm for global optimization problems", *Memetic Computing*, vol. 10(2), pp. 151–164, 2018.

[44] R. Storn and K. Price., "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces", *Journal of global optimization*, vol. 11(4), pp. 341–359, 1997.

[45] J. H. Holland, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[46] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya., "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms", *Software: Practice and experience*, vol. 41(1), pp. 23–50, 2011.

[47] T. P. Jacob and K. Pradeep., "A multi-objective optimal task scheduling in cloud environment using cuckoo particle swarm optimization", *Wireless Personal Communications*, vol. 109(1), pp. 315–331, 2019.

[48] X.-S. Yang and S. Deb., "Cuckoo search via Lévy

flights", *Presented at 2009 World congress on nature & biologically inspired computing (NaBIC)*, pp. 210–214, 2009.

[49] K. Dubey, M. Kumar, and S. C. Sharma., "Modified HEFT algorithm for task scheduling in cloud environment", *Procedia Computer Science*, vol. 125, pp. 725–732, 2018.

[50] H. Topcuoglu, S. Hariri, and M.-Y. Wu., "Performance-effective and low-complexity task scheduling for heterogeneous computing", *IEEE transactions on parallel and distributed systems*, vol. 13(3), pp. 260–274, 2002.

[51] B. L. Pan, Y. P. Wang, H. X. Li, and J. Qian., "Task scheduling and resource allocation of cloud computing based on QoS", *Advanced Materials Research*, vol. 915, pp. 1382–1385, 2014.

[52] N. Mansouri and M. M. Javidi., "Cost-based job scheduling strategy in cloud computing environments", *Distributed and Parallel Databases*, pp. 1–36, 2019.

[53] M. Shojafar, S. Javanmardi, S. Abolfazli, and N. Cordeschi., "FUGE: A joint meta-heuristic approach to cloud job scheduling algorithm using fuzzy theory and a genetic method", *Cluster Computing*, vol. 18(2), pp. 829–844, 2015.

[54] P. Vas, *Artificial-intelligence-based electrical machines and drives: application of fuzzy, neural, fuzzy-neural, and genetic-algorithm-based techniques*. 45. Oxford university press, 1999.

[55] H. Zhao, G. Qi, Q. Wang, J. Wang, P. Yang, and L. Qiao., "Energy-efficient task scheduling for heterogeneous cloud computing systems", *Presented at 2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*. pp. 952–959, 2019.

[56] X. Wei., "Task scheduling optimization strategy using improved ant colony optimization algorithm in cloud computing", *Journal of Ambient Intelligence and Humanized Computing*, 2020.

[57] U. K. Jena, P. K. Das, and M. R. Kabat., "Hybridization of meta-heuristic algorithm for load balancing in cloud computing environment", *Journal of King Saud University-Computer and Information Sciences*, 2020.

[58] A. Gupta, H. S. Bhadauria, and A. Singh., "Load balancing based hyper heuristic algorithm for cloud task scheduling", *Journal of Ambient Intelligence and Humanized Computing*. pp. 1–8, 2020.

[59] I. Bambrik., "A survey on cloud computing simulation and modeling", *SN Computer Science*, vol. 1(5), pp. –34, 2020.

[60] S. R. Jena, R. Shanmugam, K. Saini, and S. Kumar., "Cloud computing tools: inside views and analysis", *Procedia Computer Science*, vol. 173, pp. 382–391, 2020.

[61] M. Tawfeek, A. El-Sisi, A. Keshk, F. Torkey., "Cloud task scheduling based on ant colony optimization", *The International Arab Journal of Information Technology*. vol. 12, pp. 129-137, 2015.

[62] D. Gabi, A. S. Ismail, A. Zainal, Z. Zakaria, and A. Al-Khasawneh., "Cloud scalable multi-objective task scheduling algorithm for cloud computing using cat swarm optimization and simulated annealing", *Presented at 2017 8th International Conference on Information Technology (ICIT)*. pp. 599–604, 2017.