Journal of Algorithms and Computation

journal homepage: http://jac.ut.ac.ir

Load-balanced Task Scheduling Algorithm using Walrus Optimizer for Cloud Computing

Z. Jalali Khalil Abadi*[1], N. Mansouri†[1], M. Javidi‡[1] and B. Mohammad Hasani Zade§[1]

[1]Department of Computer Science, Shahid Bahonar University of Kerman, Box No. 76135-133, Kerman, Iran.

ABSTRACT

Cloud computing allows users to access software and hardware resources over the network. Task scheduling plays a major role in achieving cost-effective execution. It is a process of allocating resources to certain tasks in order to optimize one or more criteria. Metaheuristics provide promising solutions for task scheduling by leveraging bio-inspired techniques. In order to solve the task scheduling problem, we present an algorithm that uses a new meta-heuristic algorithm namely, WO (Walrus Optimizer). The proposed method is known as WO-based Task Scheduling Algorithm. Its main objective is to decrease execution cost, load balancing, resource utilization, and makespan. WOTSA is compared against several popular meta-heuristic algorithms. According to the experimental results, WOTSA improves performance in terms of different benchmarks.

ARTICLE INFO

Article history:
Research paper
Received 19, December 2023
Accepted 28, February 2024
Available online 01, August 2024

Keyword: Cloud Computing, Task Scheduling, Meta-heuristic, Walrus Optimizer, Load balancing.

AMS subject Classification: 68W40.

*zj190179@gmail.com
†Corresponding author: N. Mansouri. Email: najme.mansouri@gmail.com
‡javidi@uk.ac.ir
§behnamhasani707@gmail.com

1 Introduction

Large-scale applications have emerged as mobile and networking technologies have evolved rapidly, requiring significant processing power, storage capacity, and/or networking resources to operate efficiently. Computing resources and services can be accessed in the cloud as and when required. It provides IT (Information Technology) infrastructure, innovative services, and applications via the internet to a large number of end users. Modern IT paradigms can be utilized to process and execute large scale applications efficiently [9, 2].
As a result of cloud computing, or distributed computing worldwide, clients pay for every service they use without understanding how the service is hosted and distributed [42, 13]. Performing tasks over processing units is a crucial function of cloud computing [14]. The overall efficiency of cloud computing depends on task scheduling [6]. In cloud computing, task scheduling is an NP-hard problem. Cloud computing services become more efficient and faster when scheduling methods are efficient. Cloud scheduling issues are usually solved using optimization algorithms [31]. The NP-problem can be solved using heuristics and meta-heuristics [30, 19].
The major contributions of this paper are as follows:

- A novel task scheduling algorithm is proposed that simultaneously considers execution cost, load balancing, resource utilization, and makespan.
- The problem of task scheduling is formulated, and objective functions are presented to map tasks to virtual machines in the most efficient way.
- The ability of WO to make a suitable trade-off between exploration and exploitation makes it useful for scheduling.
- The presented algorithm is compared with GEO, FOX, STOA, and ZOA based on extensive experimental testing.

This study is organized as follows. The main concepts and preliminary information are described in Section 2. Section 3 discusses the related paper. Section 4 describes the WOTSA algorithm. In section 5, the proposed algorithm is evaluated for its performance. Future work is discussed in section 6. The full names of all the abbreviations used in the article are listed in Table 1.

2 Background

2.1 Cloud computing

Cloud computing represents a new revolution in internet computing. In comparison with distributed computing, it offers more advantages. In addition to being highly efficient, cloud computing uses a large amount of computing power [40]. Pay-as-you-go models can be used by Cloud Service Providers (CSPs) in regards to their virtual machines (VMs)

Table 1. List of Acronyms

Abbreviation	Full Name
WO	Walrus Optimizer
WOTSA	WO-based Task Scheduling Algorithm
FOX	Fox Optimizer
GEO	Golden Eagle Optimizer
ZOA	Zebra Optimization Algorithm
STOA	Sooty Tern Optimization Algorithm
IT	Information Technology
CSPs	Cloud Service Providers
VMs	Virtual Machines
IaaS	Infrastructure as a Service
PaaS	Platform as a Service
SaaS	Software as a Service
PSO	Particle Swarm Optimization
HWOA	Hybrid Whale Optimization Algorithm
MBA	Mutation-based Bees Algorithm
WOA	Whale Optimization Algorithm
RDWOA	Random Double adaptive Whale Optimization Algorithm
CS	Cuckoo Search
TSP	Task Scheduling Problem
GA	Genetic Algorithm
GSA	Gravitational Search Algorithm
ACO	Ant Colony Optimization
LBACO	Load Balancing Ant Colony
ABFSOS	Adaptive Benefit Factor based Symbiotic Organisms Search
CMABFSOS	Mult-objective Constrained ABFSOS
MVO	Multi-Verse Optimization
IMOMVO	Improved Multi-objective Multi-Verse Optimizer
NSGAII	Non-dominated Sorting Genetic Algorithm II
QOS	Quality of Service

[23]. Further, cloud computing can be used to access a variety of software, and DCs can be used to store the cloud customer's critical data as well as security features [16]. Service providers provide security and compliance to drive the maturing market and adoption of cloud computing. There are several reasons for this growth, including the increased flexibility and cost savings that cloud services will provide [26, 39]. The components of a basic cloud computing architecture diagram are as follows (see Fig. 1):

- Cloud Infrastructure: The components of physical and virtual infrastructure, including servers, storage, and networking. Cloud infrastructure security guidelines, rules, and practices are usually defined by information security professionals [34].

- Virtualization Layer: It illustrates the way virtualization technology abstracts and distributes resources across clouds. Virtualization is a process that separates the physical environment from the services and resources that are deployed. Many benefits can be derived from virtualized environments. The complexity of virtualization and the number of entry points present new challenges for attackers [28, 3].
- Service Models: Showcases various service models, including Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). As a result, it is flexible enough to handle rapidly changing customer needs and gives customers a reliable solution.
- Deployment Models: There are several types of cloud deployments, including public, private, hybrid and community.
- Users and Devices: Describes end users and client devices accessing cloud services.

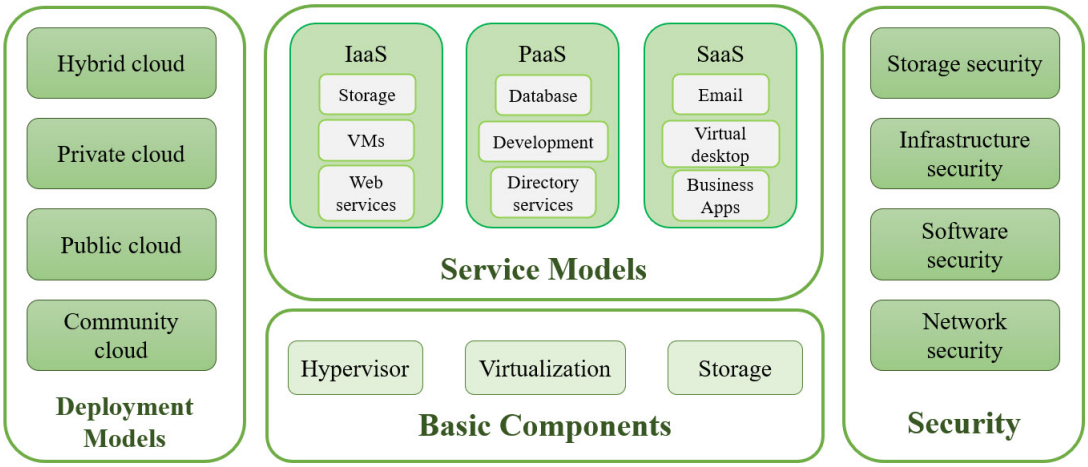


Figure 1. The cloud computing architecture.

Firewalls, encryption, and access controls are all part of cloud security. The provision of services includes security features such as authentication and encryption. Data security, availability, and safety must be ensured by the vendors. Cloud infrastructure depends on scalability, instant elasticity, and metering resources. In the public cloud, unstructured data is stored in a multi-tenant environment [38]. Private cloud services provide a dedicated storage environment protected behind a firewall owned by the customer or organization. In hybrid clouds, private and public cloud services are mixed together to provide more data deployment options and business flexibility.

2.2 Task scheduling

Scheduling involves allocating jobs to available resources. Nodes and resources are capable of running tasks as the smallest numerical units. There are many parameters that can

be attached to a task. An operation requires a resource. As an example, a processor that processes data, a data storage device, or a network link that transports data. Cloud providers and consumers face scheduling problems.

Resource scheduling: Physical machines and servers can be scheduled on demand for random tasks. It is possible to run multiple processes on a single VM, but this degrades performance. Long-term holding of a VM can lead to under provisioning. Service costs increase as over-provisioning consumes resources and time excessively [41].

Workflow Scheduling: This tool helps map processes and manage the interdependencies between work within a process. Resources are also allocated to ensure that various workflows are completed in accordance with objectives [11].

Task Scheduling: Resources are assigned tasks based on their homogeneity, heterogeneity, or centralization. One scheduler is used by all systems to schedule different tasks in centralized task scheduling. Furthermore, centralized scheduling lacks scalability and fault tolerance. Each task sent to the cloud system is handled by a series of schedulers [18, 7].

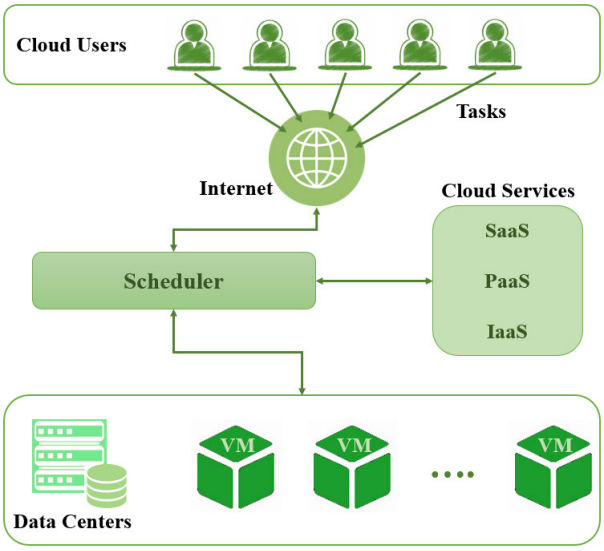


Figure 2. Task scheduling model.

It is important for cloud consumers to execute their jobs in order to solve problems of varying sizes and complexity. Cloud consumers will benefit from wisely selecting and aggregating resources, while cloud providers can maximize their resource utilization. In NP-hard task scheduling problems, a set of tasks is assigned to a set of resources in a way that achieves a specific objective, such as minimizing the overall completion time or resource utilization. This problem belongs to the class of problems that have no known efficient algorithm that can solve them in polynomial time. In order to find near-optimal solutions in a reasonable amount of time, researchers often use heuristics and approximation algorithms. It is possible to get good practical solutions from these algorithms by trading optimality for efficiency. The optimal solution to these problems cannot be derived from any algorithm in polynomial time [1]. The process of generating schedules has

a high overhead. Due to the competition for resources and time in the cloud, metaheuristic algorithms like PSO are used [37]. Figure 2 illustrates how cloud computing tasks are scheduled. In general, task scheduling models entail a set of dependent tasks (performed by the user) coupled with an interconnected processor. The purpose of this modeling is to determine an optimal distribution strategy for cloud nodes. It is important to understand the resources required to perform a particular task (CPU, memory, storage), the execution time, and any dependencies (if any). Each task is assigned a resource by the scheduler. Task dependencies are taken into account in dependency-aware scheduling, to ensure tasks are executed in the correct order to avoid deadlocks. Distributed schedulers are often used in cloud computing environments to schedule tasks across multiple servers and data centers. Resource allocation and task placement are optimized using heuristic optimization techniques, queueing theory, and machine learning algorithms. Performance and SLA requirements of the applications and users must be met if high throughput, low latency, and efficient resource utilization are to be achieved.
In addition to their ease of use, self-organization, scalability, and robustness, metaheuristic algorithms play a crucial role in task scheduling and load balancing.

2.3 Walrus Optimizer (WO)

The metaheuristic algorithm optimizes the search process by exploitation and exploration. As real-life optimization problems have become more complex, more metaheuristic algorithms have been developed. Walruses use key signals to migrate, breed, roost, feed, gather, and escape (danger signals and safety signals). WO [15] enhances optimization calculation efficiency by advancing artificial intelligence applications and promoting continuous development.
To solve searching optimization problems, WO emulates the social and foraging behaviors of walruses while migrating, breeding, roosting, feeding, gathering, and escaping. By balancing exploration and exploitation, the algorithm is able to find high-quality solutions. When the danger signal is received, the WO determines whether to explore or exploit. The new domain is selected when the danger signal exceeds one during the exploration phase of the algorithm. When an algorithm reaches its late stage, the walrus herd reproduces (exploitation). As a result of the security signal, individual walrus chooses whether to roost or forage during the exploitation phase. In addition, danger signals also control foraging behavior. During the exploitation phase, the safety signal influences whether the walrus chooses to roost or forage. Figure 3 shows the main steps of this algorithm.
The main steps of the WO algorithm are shown in the above figure (Figure 3):

1. **Set the population size:** Define a population of walrus agents with random positions in the search space.
2. **Prey Selection:** Walrus agents select prey (best solution) based on fitness and weighting.
3. **Migration:** Walrus agents move towards their selected prey, with a step size determined by their distance and speed from the prey.

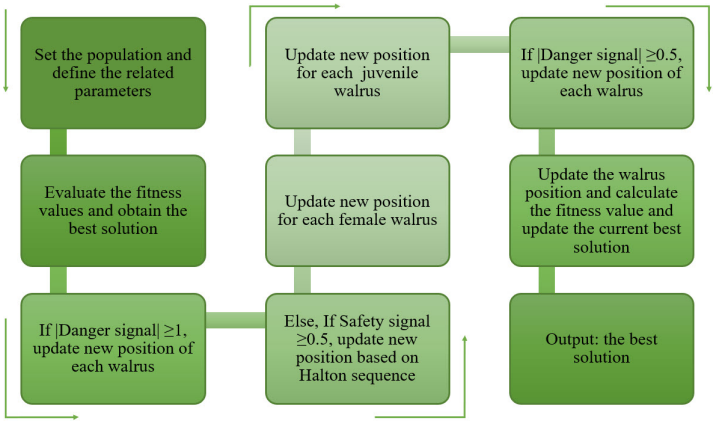


Figure 3. The main steps of WO.

4. **Breeding:** A combination of their positions and random perturbations generates offspring (new solutions).

5. **Roosting and Feeding, Gathering, and Escaping:** Local search (local roosting) and global search (global feeding) are conducted by walrus agents randomly. Their positions are updated according to the best solutions found during roosting and feeding. Convergence is enhanced by Walrus agents gathering around the best solution found so far.

6. **Weighting Mechanism:** Exploration and exploitation are dynamically balanced by the weighting mechanism. Exploration initially covers a larger search space and the algorithm refines the search around promising areas. The best solution is the one that minimizes the objective function by mapping tasks to virtual machines.

WO parameters and assignments are as follows [15]:

- A typical population contains between 30 and 100 walrus agents. Populations with larger sizes allow the algorithm to explore a larger search space, but they also increase computation costs.

- Maximum number of iterations: It is based on the complexity of the problem and the quality of the solution. More iterations can lead to better solutions, but they also cost more.

- Weighting factor ω: The value is initially set high (e.g., 0.9) to facilitate exploration. The weighting factor should be gradually reduced to favor exploitation.

- Migration rate: It is usually set between 0.1 and 0.5. The higher the migration rate, the faster the algorithm will converge, but it may also prematurely converge.

- Breeding probability: In most cases, it is set between 0.5 and 0.9. It increases the likelihood of breeding a new solution, but slows convergence speed.

- Roosting probability: It is usually set between 0.1 and 0.3. The higher the roosting probability, the harder it will be for the algorithm to discover new areas to explore.
- Feeding probability: This value is usually set between 0.1 and 0.3. It is useful for exploring new search space, but it hampers refinement of the best answers.
- Gathering probability: It is usually set between 0.1 and 0.2. Higher gathering probabilities mean faster convergence.
- Escaping probability: It is usually set between 0.01 and 0.1. The higher the escaping probability, the less likely the algorithm is to get stuck in local optima.

In search space, danger signals indicate areas associated with poor solutions. Walrus agents avoid moving toward high danger areas. There are good solutions in a search space with a safety signal. Agents tend to move towards areas with high safety signals. While migrating, each walrus agent calculates the danger and safety signals. Fitness of the solutions at those positions determines safety and danger signals. The highest safety signals and the lowest danger signals are favored by walruses. As a result, the walrus agents are guided towards areas with promising solutions and away from areas with inadequate ones. Gaussian functions are typically used in the WO algorithm to represent danger and safety signals. As the distance from the prey increases, the Gaussian function decreases in value.

3 Related Work

System performance is greatly affected by scheduling tasks in cloud computing. It is necessary to improve and optimize the algorithm serving the task scheduling process. This section discusses some of the latest task scheduling techniques. In order to improve the performance and customer satisfaction level of cloud systems, task scheduling needs to be addressed. In addition, the task scheduling scheme has a direct impact on the execution time and execution cost. Manikandan et al. [22] proposed an HWOA-based MBA for solving multi-objective task scheduling problems. Multi-objective behavior minimizes resource utilization in the hybrid WOA-based MBA algorithm. The Bees algorithm uses mutation operators to enhance RDWOA's output. Resource utilization, makepan, and execution time are used to assess performance.
A Cat Swarm Optimization algorithm was used by Mangalampalli et al. [20] to schedule tasks based on the parameters of makespan, migration time, energy consumption, and total power cost. Prioritization at the task level determines which VMs should receive tasks, and priorities at the VM level determine which VMs should receive tasks. It uses CloudSim simulator to simulate power costs and generates input randomly from CloudSim. Comparing the proposed algorithm with existing algorithms such as PSO (Particle Swarm Optimization) and CS (Cuckoo Search). The metrics evaluated included makespan, migration time, energy consumption, and total power cost, and migration time was reduced by 34% versus PSO and 29% versus CS.

Pirozmand et al. [29] proposed a hybrid algorithm for solving the Task Scheduling Problem (TSP) in cloud computing called GSAGA. In spite of its high search ability, the Genetic Algorithm (GA) is inefficient at local search and stability. Consequently, a stable algorithm can be developed by combining the GA and the Gravitational Search Algorithm (GSA). VM analysis is used to determine energy consumption, gain cost, and makespan in GSAGA.

Manikandan et al. [21] proposed a method for efficient resource allocation using Black widow optimization in combination with fish swarm optimization to reduce cost, energy, and resource utilization. As compared to three other algorithms, the proposed method performed well as far as time, energy, and cost were concerned.

The evolution of cloud computing has reduced costs and improved results. Users use applications, information, data, and applications, as well as resources. In addition, it is important to maintain the system's stability and to be flexible when making changes. Using a neural network to schedule many activities, Sharma et al. [33] presented an updated ACO approach for optimizing global search. Multi-objective techniques can be combined to optimize the organization of tasks and resources. Optimization of the objective function was more effective and faster using LBACO. The ACO approach reduces mean access times and ensures consistent job assignments.

Abdallahi et al. [4] proposed an adaptive benefit factor based symbiotic organisms search (ABFSOS) for faster convergence speed by adapting SOS control parameters. Fine-tuning the penalty function values may prevent infeasible solutions and premature convergence. It investigates the performance of a multi-objective constrained ABFSOS framework (CMABFSOS) on a standard simulator (CloudSim). In all workload instances, the proposed CMABFSOS algorithm produced non-dominated solutions that outperformed the compared algorithms (EMS-C, ECMSMOO). The proposed CMABFSOS algorithm improved significantly over the compared algorithms in terms of hypervolume (convergence and diversity). When compared to EMS-C, CMABFSOS performs 17.02 to 47.73% better, whereas ECMSMOO performs 19.98 to 52.18% better.

The multiverse optimization algorithm (MVO) is one of the most widely used algorithms today. Because MVO restricts search space to the best solution, the search domain is poor, therefore, the search time is long. To solve task scheduling problems, Otair et al. [27] introduced an improved multi-objective multi-verse optimizer (IMOMVO). IMOMVO is introduced as a result of the disadvantages of the original version of MVO as well as its enhanced version. The best and second-best solutions are used for solving the problem of average positioning. It was evaluated using a variety of datasets containing tasks and virtual machines. There are three metrics included in the validation process: task execution time, throughput, and VM processing power. The proposed method performed better than other state-of-the-art methods. In order to evaluate the proposed work, three objectives were used: Execution Time, Throughput, and Makespan.

Imene et al. [17] proposed a multi-objective scheduling method for cloud VMs that used Non-dominated Sorting Genetic Algorithm (NSGA-III). Multi-objective adaptation is used to minimize cost, runtime, and power consumption. A performance comparison showed that NSGAIII was more efficient than its predecessor, Non-dominated Sorting

Genetic Algorithm II (NSGAII).
An analysis of the scheduling algorithms is provided in Table 2. All three parameters have a significant impact on cloud system efficiency, as outlined in Table 2. Several parameters, including makespan, resource utilization, load balancing, and execution costs, are taken into account in the algorithm presented in this paper.

Table 2. Comparison of the scheduling algorithms in cloud environment

Reference	Year	Performance metrics	Simulation tool	Algorithm type	Technique used for improving QoS	Advantages & Disadvantages
Manikandan et al. [22]	2022	Task completion time and execution time	Not reported	HWOA based MBA	Accelerate computations	It is important to consider overloads, memory usage, and peak demands.
Mangalampalli et al. [20]	2022	Makespan, energy consumption, cost, and migration time	CloudSim	The cat swarm optimization algorithm is used for multi-objective task scheduling in cloud computing	A good performance is achieved in makespan, migration time, energy consumption, and total power consumption.	The algorithm didn't evaluate in real time environment.
Pirozmand et al. [29]	2022	Energy consumption, gain cost, and makespan	CloudSim 3.3 and the NetBeans	GSAGA	By evaluating runtime, energy consumption, cost, and makespan, the proposed GSAGA hybrid algorithm enhances the VM analysis system.	SDN-based networks and container-based cloud applications improve quality of service (QoS).

Manikandan et al. [21]	2022	Cost, energy, and resource utilization	CloudSim	BWFSO	In order to reduce costs, energy, and resource usage, it allocates resources efficiently.	Using virtual machine (VM) instances for scheduling has a number of drawbacks, including lengthy startup times and resource consumption. Inefficiencies like this could limit the effectiveness of traditional VM-based scheduling methods.
Sharma et al. [33]	2022	Cost and execution time	CloudSim	LBACO	LBACO optimizes the objective function best. Moreover, it is the fastest. Besides providing higher statistics and mean access times, it also provides more consistent assignment patterns.	ACO-based models should be tailored to high-dimensional datasets. A multi-mode resource-constrained project scheduling challenge can also be examined. It's important to look at the dynamic limitations or tasks on the first fundamental solutions in the cloud-based project tasks.

Abdullahi et al. [4]	2021	Convergence and diversity	CloudSim	ABFSOS	The proposed algorithm which results in better global convergence, thereby producing better task schedule solutions.	More efficient methods of handling the control parameters can be investigated through empirical analysis.
Otair et al. [27]	2022	Execution time, throughput	CloudSim	IMOMVO	Throughput results for the IMOMVO task scheduling algorithm were 0.79, allowing more tasks to be completed. In this ratio, more tasks are executed in a single slot.	The important metrics like energy, cost, load balancing, and security didn't consider.
Imene et al. [17]	2022	Runtime, cost, and energy consumption	CloudSim	NSGAIII	NSGAIII has shown better results for the objectives than NSGAII on one hand and for the good functioning of the system itself on the other.	The application of NSGAIII can cite with more than three QoS values.

The proposed method	2024	Cost, resource utilization, load balancing, and makespan	MATLAB	WOTSA	The proposed algorithm can also avoid falling into the local optimal trap and has better convergence.	We intend to consider other important parameters such as security, scalability, and availability.

4 The WOTSA (Walrus Optimization Task Scheduling Algorithm)

During task scheduling, submitted tasks are assigned to available resources in order to maximize resources' utilization and QoS. Therefore, task assignments are determined by restrictions imposed by users and cloud providers [12]. Using WOTSA, the proposed algorithm generates a set of solutions and divides them into groups to solve the task scheduling problem. In Subsection 4.1, the task scheduling problem concepts are introduced and in Subsection 4.2, the objective function of WOTSA is discussed.

4.1 Task scheduling modeling

VMs are assigned tasks based on user needs and service quality by the data center broker and cloud information service. Various quality of service metrics are improved by task scheduling in cloud computing. Suppose a cloud data center consists of n number of the task such as: $Task = \{Task_1, Task_2, \ldots, Task_n\}$, where $Task_i$ indicated the $i-th$ task, and m number of VMs such as: $VM = \{VM_1, VM_2, \ldots, VM_m\}$, where VM_j indicated the $j-th$ VM. The condition for executing such tasks is that $n > m$.
It considers four important criteria simultaneously, namely makespan, cost, resource utilization, and load balancing. The makespan is a metric that describes the time it takes to finish the last task and exit the cloud system. Minimizing costs increases the user's profit. According to the scheduling algorithm, the lowest-cost VM is selected to execute the task to minimize execution costs. Providing high system performance and optimizing resource utilization is possible through load balancing by distributing computation workloads across available resources (like virtual machines or servers). In load balancing, tasks are distributed evenly across the available resources, preventing one resource from being overworked while others remain underutilized. As a result of this optimization, idle time is reduced and computational resources are utilized more efficiently.

4.2 Objective function

As a result of WOTSA, all tasks are scheduled so that their makespan, resource utilization, and execution cost are minimized. As a result, the user is satisfied and the efficiency increases. The algorithm outputs a matrix with m columns and n rows indicating which VM should execute each task. The objective function is calculated as follows:

$$X = \begin{pmatrix} x_{11} & \cdots & x_{1m} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nm} \end{pmatrix} \tag{1}$$

where x_{ij} is a decision variable and calculated by:

$$x_{ij} = \begin{cases} 1 & \text{if } \mathrm{T}_i \text{ is assigned to } \mathrm{VM}_j \\ 0 & \text{if } \mathrm{T}_i \text{ is not assigned to } \mathrm{VM}_j \end{cases} \tag{2}$$

With this condition:

$$\sum_{j=1}^{m} x_{ij} = 1 \qquad \text{for } 1 \leq \mathrm{i} \leq \mathrm{n} \tag{3}$$

The agent adapts to the task based on the information it receives. Afterward, task positions are randomly assigned based on the task. After that, each number is rounded up. It indicates that the task has been assigned to that VM. The second virtual machine is assigned to the fourth task when its number is rounded to 2.

Makespan: Makespan refers to the time it takes for all tasks to be completed by resources. The effectiveness of resource utilization determines how well VMs are utilized in the cloud. Reducing the makespan rate will satisfy the user and speed up the executions. Makespan is defined as follows [32]:

$$Makespan = \max\left(Ext_j\right),\ 1 \leq j \leq m \tag{4}$$

Where Ext_j is the VM_j execution time which calculated based on Eq. 5.

$$Ext_j = \sum_{i=1}^{n} x_{ij} \times CT_{ij} \tag{5}$$

Where X_{ij} is the decision variable and CT_{ij} is the completion time of executing task i on VM_j which calculated by Eq. 6.

$$CT_{ij} = \frac{lengthoftheTask_i}{processingtimeoftheVM_j} \tag{6}$$

Resource Utilization (RU): Efficiency refers to the number of resources a system utilizes efficiently. It maximizes cloud service provider profits by minimizing idle time

and keeping resources busy executing customer tasks. Based on this definition, the RU is [36]:

$$RU = \frac{TEXEC}{Makespan \times M} \tag{7}$$

Where M shows the number of VMs and $TEXEC$ is the total execution time which calculated by Eq. 8.

$$TEXEC_{ij} = \sum_{i=1}^{m} VMET_j,\ 1 \leq j \leq M \tag{8}$$

Where $VMET_j$ is the execution time of $j-th$ VM which calculated based on Eq. 9.

$$VMET_j = \sum_{i=0}^{M} x_{ij} \times EXEC_{ij},\ 1 \leq j \leq M \tag{9}$$

Which N shows the number of Tasks, and $EXEC_{ij}$ is the execution time of Task i on virtual machine VM_j that obtained based on Eq. 10.

$$EXEC_{ij} = \frac{L_{Task_i}}{Cap_{VM_j}} \tag{10}$$

Where L_{Task_i} is the computing time for executing Task i and Cap_{VM_j} is the capacity of virtual machine j.

Execution Cost: Execution cost depends on how much a VM costs per unit of time and how long it takes to execute a task. In this way, a suitable task scheduling algorithm can allocate tasks to VMs in an optimal way so that execution costs can be reduced. The execution cost of Task i can be calculated as follows [5]:

$$EC_{ij} = Price_j \times \frac{CT_{ij}}{3600} \tag{11}$$

Where $Price_j$ is the price of VM_j and CT_{ij} is the completion time of executing Task i on VM_j.

Load Balancing (LB): During this phase, each virtual machine's load is calculated over time. Therefore, the load of a VM can be calculated as the total length of tasks at time (t) on service queue of VM_i divided by the service rate of VM_i at time t, as given in Eq. 12 [8].

$$LB[VM_i, time(t)] = \frac{Totallengthof Task(T)}{Processingtimeof VM} \tag{12}$$

As a final step, the optimization objective function is calculated as follows:

$$F_{optimal} = Makespan + (\frac{RU}{M}) + (\frac{LB}{M}) + (\frac{TEXEC}{N}) \tag{13}$$

On the basis of all the above, Algorithm 1 illustrates the pseudocode for scheduling tasks based on the WO algorithm. Algorithm 2 shows the fobj (fitness objective function) used in Algorithm 1. Figure 4 shows the flowchart of WOTSA for task scheduling.

Algorithm 1 Pseudocode for WOTSA method

Input: Tasks set, VMs set, the parameters of WOTS algorithm, N (population size) **Output:** Return best search

Begin

1. Initialize set of tasks, $Task = \{Task_1, Task_2, \ldots, Task_n\}$.
2. Initialize set VMs, $VM = \{VM_1, VM_2, \ldots, VM_m\}$.
3. Set the population size and maximum iteration.
4. Set the parameters of WO.
5. $t = 1$
6. **While** ($t \leq$ maximum iteration)
7. **For** $i = 1$ to N
8. $[O_i]$= fobj (walrus$_i$); // Evaluate the fitness values of each walrus using Algorithm 2.
9. Find the best search agent;
10. **End** for
11. $t++$;
12. **End** while
13. **Return** best solution // The allocation matrix which minimizes the objective function.

End

Algorithm 2 Fitness Objective Function (fobj) Pseudocode

Input: Task, VM, Joblen (task information: task length and task ID), St (waiting time).
Output: Optimal solution (O), Makespan, Resource utilization (RU), Execution cost (EC), Load balancing (LB).

Begin

1. **For** each Task
2. Execution time = (Joblen/VM size);
3. **End** for
4. **For** each VM
5. EC = ((Execution time/3600) * VM price);
6. Task processing time = (Joblen/(VM size * CPU utilization)) ;
7. **End** for
8. EC_mean = sum (sum (EC)); // Sum all of VMs excution cost.
9. Makespan = min (sum(execution time + St));
10. **For** $m = 1$ to size(VM)
11. Utilization (m) = (Joblen / Makespan);
12. **End** for
13. **For** each VM
14. LB [VM] = (total length of task/ Task processing time);
15. **End** for
16. RU_mean = (sum (Utilization) / size (VM));
17. O = Makespan + RU_mean + average_LB + EC_mean;
18. **Return**(O) //The best solution for mapping tasks to VMs is the one that minimizes objective function.

End

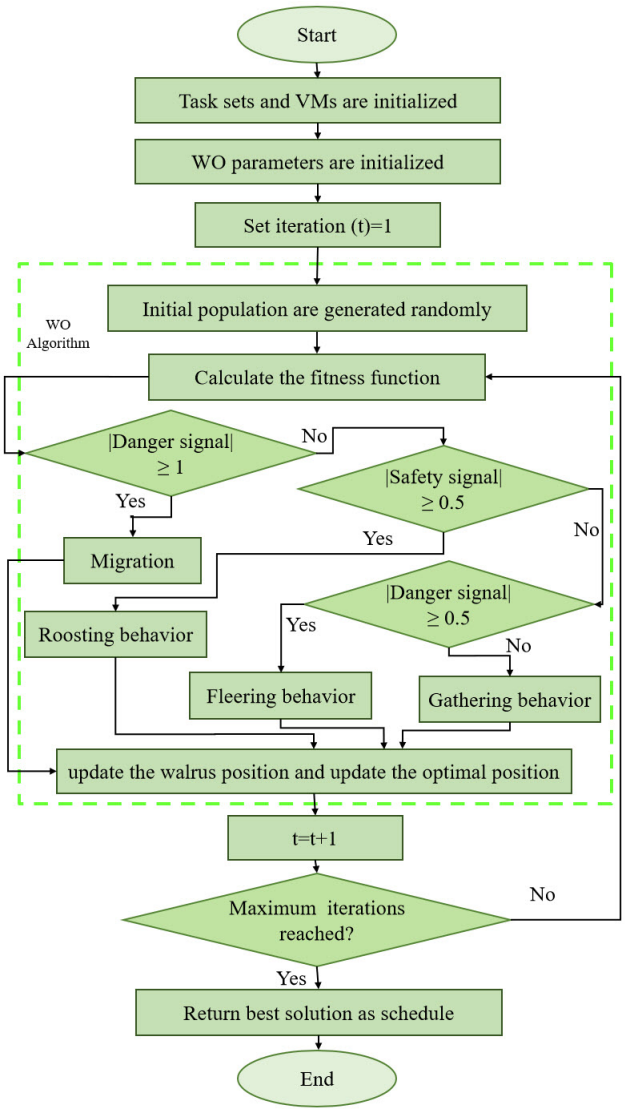
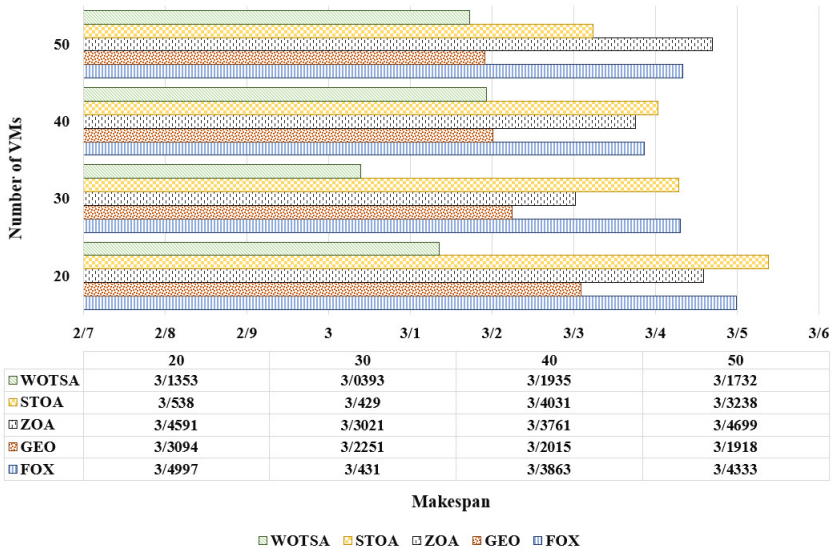


Figure 4. Flow chart of WOTSA [15].

5 Results Discussion

The performance of the WOTSA evaluated in MATLAB R2018a software on a PC with Intel(R) Core (TM) i5-8250U CPU with 1.80 GHz, and RAM of 12 GB. The WOTSA is compared with four well-known meta-heuristic algorithms, FOX (Fox optimizer) [25], GEO (Golden Eagle Optimizer) [24], ZOA (Zebra Optimization Algorithm) [35], and STOA (Sooty Tern Optimization Algorithm) [10]. All proposed meta-heuristic algorithms are evaluated under the same conditions and with the same objective function (the objective function proposed in this paper).

Scenario 1: Tasks are fixed, but VMs range from 20 to 50. In Table 3, details of the scenario 1 environmental simulation are presented.

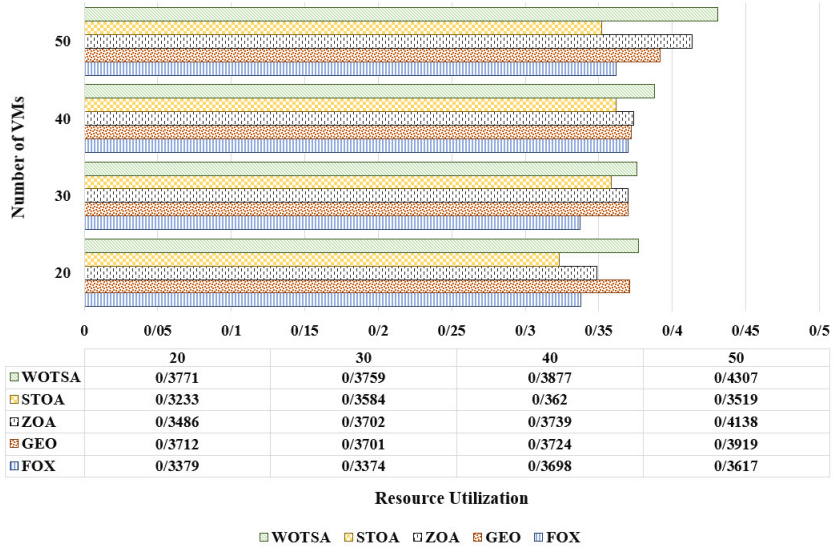


	20	30	40	50
WOTSA	3/1353	3/0393	3/1935	3/1732
STOA	3/538	3/429	3/4031	3/3238
ZOA	3/4591	3/3021	3/3761	3/4699
GEO	3/3094	3/2251	3/2015	3/1918
FOX	3/4997	3/431	3/3863	3/4333

Figure 5. Makespan comparison (various numbers of VMs).

Table 3. Experiment setup details for scenario 1.

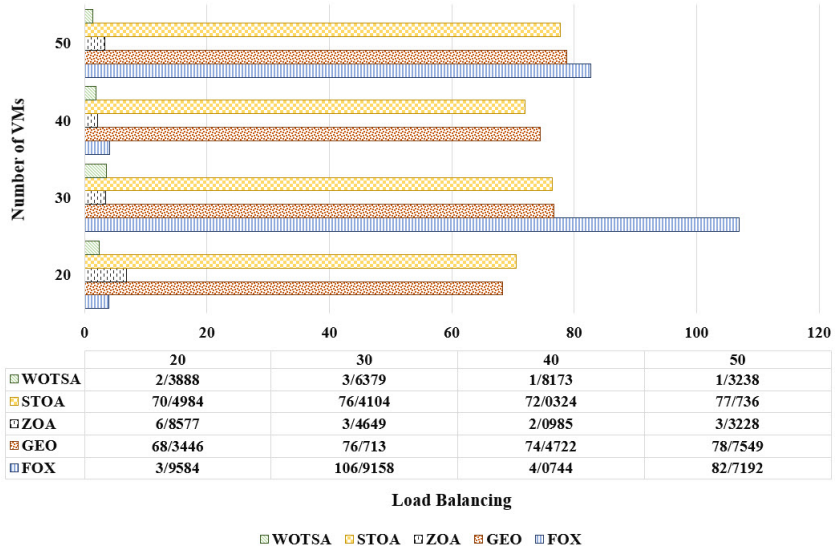
Parameters	Value
Number of tasks	100
Population size	30
Number of VMs	20-50
Maximum iteration	100



	20	30	40	50
WOTSA	0/3771	0/3759	0/3877	0/4307
STOA	0/3233	0/3584	0/362	0/3519
ZOA	0/3486	0/3702	0/3739	0/4138
GEO	0/3712	0/3701	0/3724	0/3919
FOX	0/3379	0/3374	0/3698	0/3617

Figure 6. Resource utilization comparison (various numbers of VMs).

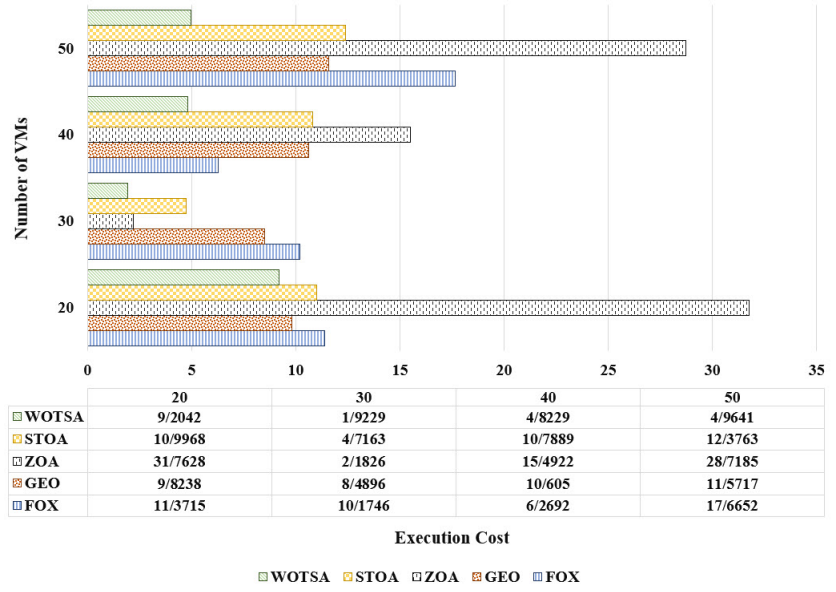
Based on different numbers of VMs and fixed tasks, Figure 5 shows that the WOTSA has the lowest makespan (about 49%) when the number of VMs is increased. Figure



	20	30	40	50
WOTSA	2/3888	3/6379	1/8173	1/3238
STOA	70/4984	76/4104	72/0324	77/736
ZOA	6/8577	3/4649	2/0985	3/3228
GEO	68/3446	76/713	74/4722	78/7549
FOX	3/9584	106/9158	4/0744	82/7192

Figure 7. Load balancing comparison (various numbers of VMs).

6 illustrates how the algorithms utilize resources. In comparison to FOX, GEO, ZOA, and STOA, the WOTSA is more efficient. In Figure 7, we compare the degree of load balancing between different algorithms. According to Figure 8, tasks are allocated to resources reasonable value and VMs are distributed efficiently.



	20	30	40	50
WOTSA	9/2042	1/9229	4/8229	4/9641
STOA	10/9968	4/7163	10/7889	12/3763
ZOA	31/7628	2/1826	15/4922	28/7185
GEO	9/8238	8/4896	10/605	11/5717
FOX	11/3715	10/1746	6/2692	17/6652

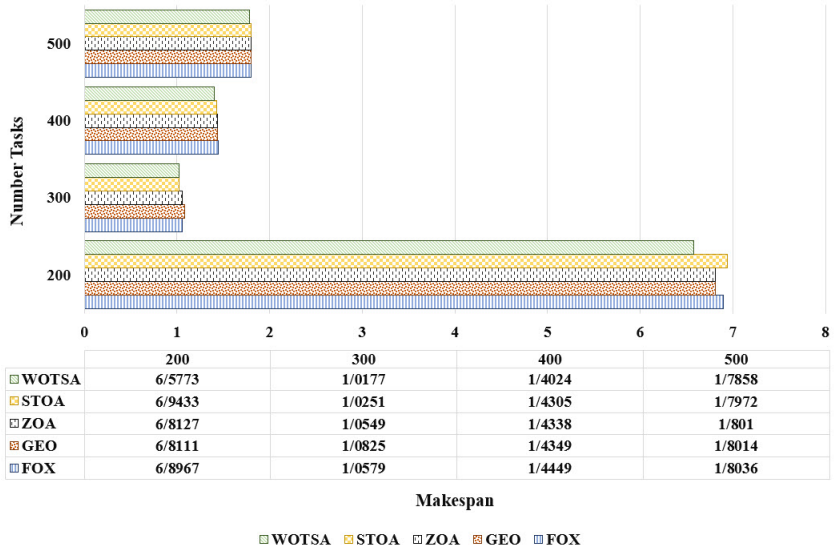
Figure 8. Execution cost comparison (various numbers of VMs).

Scenario 2: This scenario involves a fixed number of VMs with a variable number of tasks. There are 200 to 500 tasks. Table 4 shows the parameters for scenario 2.
In Figure 9, we compare the makespan metric between FOX, GEO, ZOA, and STOA

Table 4. Experiment setup details for scenario 2.

Parameters	Value
Number of tasks	200-500
Population size	30
Number of VMs	50
Maximum iteration	100

based on various task counts. There is a better makespan with WOTSA than with other methods. According to Figure 10, the WO task scheduling algorithm is more efficient than other meta-heuristic algorithms based on resource utilization. When there are more tasks, the resource utilization decreases. A WO-based task scheduling algorithm is illustrated in Figure 11 as having better load balancing. Figure 12 shows that execution costs increase with increasing task numbers.



	200	300	400	500
WOTSA	6/5773	1/0177	1/4024	1/7858
STOA	6/9433	1/0251	1/4305	1/7972
ZOA	6/8127	1/0549	1/4338	1/801
GEO	6/8111	1/0825	1/4349	1/8014
FOX	6/8967	1/0579	1/4449	1/8036

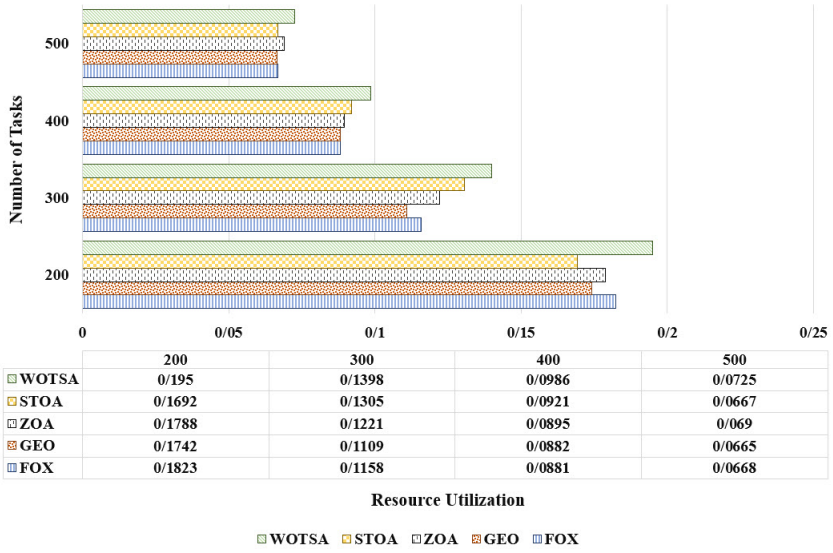
Figure 9. Makespan comparison (various numbers of tasks).

Scenario 3: In this scenario, the experiment is run based on the number of iterations. Table 5 illustrates the parameters of this scenario. Figure 13 shows the convergence

Table 5. Experiment setup details for scenario 2.

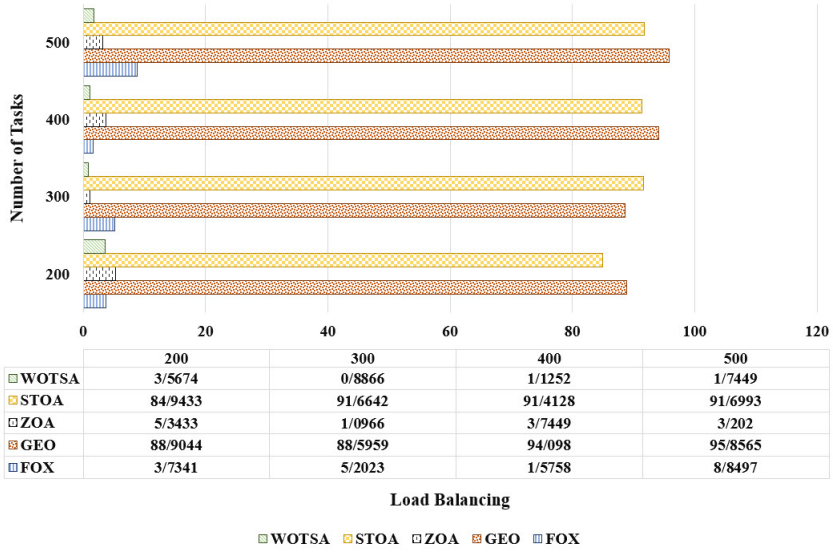
Parameters	Value
Number of tasks	300
Population size	50
Number of VMs	50
Maximum iteration	1-300

speed of the proposed algorithm and the others. According to the results, the proposed algorithm outperforms others. Additionally, because the output data were not close to



	200	300	400	500
WOTSA	0/195	0/1398	0/0986	0/0725
STOA	0/1692	0/1305	0/0921	0/0667
ZOA	0/1788	0/1221	0/0895	0/069
GEO	0/1742	0/1109	0/0882	0/0665
FOX	0/1823	0/1158	0/0881	0/0668

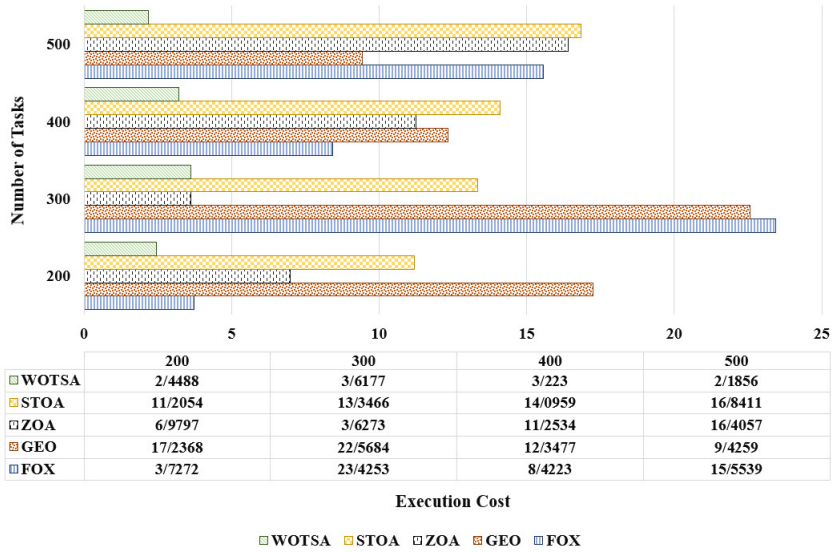
Figure 10. Resource utilization comparison (various numbers of tasks).



	200	300	400	500
WOTSA	3/5674	0/8866	1/1252	1/7449
STOA	84/9433	91/6642	91/4128	91/6993
ZOA	5/3433	1/0966	3/7449	3/202
GEO	88/9044	88/5959	94/098	95/8565
FOX	3/7341	5/2023	1/5758	8/8497

Figure 11. Load balancing comparison (various numbers of tasks).

one another, the 'z_score' function was used to normalize the data by centered means and scaling standard deviations. A disadvantage of STOA is its poor exploitation ability, as can be seen from the figure. ZOA, FOX, and GEO explore the search space in the first iterations, but fall into the trap of local optimization in later iterations and cannot achieve global optimization. With increasing iterations, the fitness value of the WOTSA method diminishes. Due to its trade-off between searching, catching, and handling, WO is a good choice. The first iteration ensures that WOTSA searches the entire search space and avoids the local optimal trap, and the last iteration ensures that WOTSA converges to the best global optimal solution.



	200	300	400	500
WOTSA	2/4488	3/6177	3/223	2/1856
STOA	11/2054	13/3466	14/0959	16/8411
ZOA	6/9797	3/6273	11/2534	16/4057
GEO	17/2368	22/5684	12/3477	9/4259
FOX	3/7272	23/4253	8/4223	15/5539

Figure 12. Execution cost comparison (various numbers of tasks).

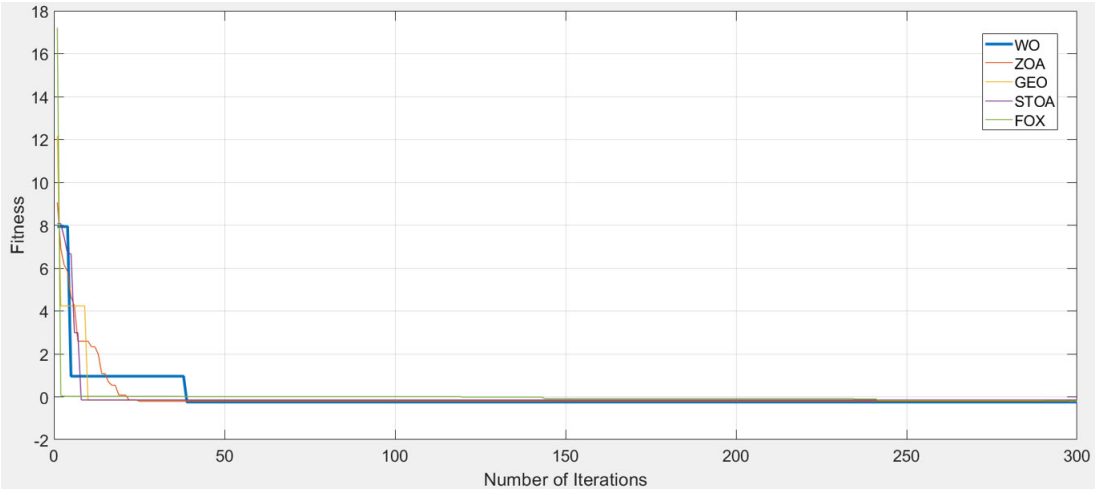


Figure 13. The algorithm's convergency.

6 Conclusion

In cloud computing, virtual resources are made available to companies and end users via a pay-per-use model. The adoption of a task scheduler that maps tasks to VMs optimally is crucial in cloud computing. Thus, this paper presents a task scheduling algorithm based on the popular WO algorithm. Four important objectives are simultaneously optimized with the proposed algorithm, including minimizing makespan, minimizing execution costs, minimizing load balancing, and maximizing resource utilization. As compared to FOX, ZOA, STOA, and GEO, the WOTSA algorithm improves makespan, load balancing, cost, total execution time, resource utilization, and resource load balance. Also, the proposed algorithm has better convergence compared to other algorithms and avoids falling into

the local optimal trap. In the future, we intend to consider security, scalability, and availability as well. In addition, we plan to improve the WO algorithm's efficiency.

References

[1] Zahra Jalali Khalil Abadi, Najme Mansouri, and Mahshid Khalouie. Task scheduling in fog environment—challenges, tools & methodologies: A review. *Computer Science Review*, 48:100550, 2023.

[2] Mohamed Abd Elaziz and Ibrahim Attiya. An improved henry gas solubility optimization algorithm for task scheduling in cloud computing. *Artificial Intelligence Review*, 54(5):3599–3637, 2021.

[3] Mohamed Abdelrazek, John Grundy, and I Mueller. An analysis of the cloud computing security problem. 2010.

[4] Mohammed Abdullahi, Md Asri Ngadi, Salihu Idi Dishing, and Shafi'i Muhammad Abdulhamid. An adaptive symbiotic organisms search for constrained task scheduling in cloud computing. *Journal of ambient intelligence and humanized computing*, 14(7):8839–8850, 2023.

[5] Dabiah Alboaneen, Hugo Tianfield, Yan Zhang, and Bernardi Pranggono. A metaheuristic method for joint task scheduling and virtual machine placement in cloud data centers. *Future Generation Computer Systems*, 115:201–212, 2021.

[6] AI Awad, NA El-Hefnawy, and HM Abdel_kader. Enhanced particle swarm optimization for task scheduling in cloud computing environments. *Procedia Computer Science*, 65:920–929, 2015.

[7] Cebrail Barut, Gungor Yildirim, and Yetkin Tatar. An intelligent and interpretable rule-based metaheuristic approach to task scheduling in cloud systems. *Knowledge-Based Systems*, 284:111241, 2024.

[8] Tushar Bhardwaj and Subhash Chander Sharma. Fuzzy logic-based elasticity controller for autonomic resource provisioning in parallel scientific applications: a cloud computing perspective. *Computers & Electrical Engineering*, 70:1049–1073, 2018.

[9] Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal, James Broberg, and Ivona Brandic. Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation computer systems*, 25(6):599–616, 2009.

[10] Gaurav Dhiman and Amandeep Kaur. Stoa: a bio-inspired based optimization algorithm for industrial engineering problems. *Engineering Applications of Artificial Intelligence*, 82:148–174, 2019.

[11] Tingting Dong, Fei Xue, Chuangbai Xiao, and Jiangjiang Zhang. Workflow scheduling based on deep reinforcement learning in the cloud environment. *Journal of Ambient Intelligence and Humanized Computing*, 12(12):10823–10835, 2021.

[12] Reyhane Ghafari and Najme Mansouri. Cost-aware and energy-efficient task scheduling based on grey wolf optimizer. *Journal of Mahani Mathematical Research*, 12(1):257–288, 2023.

[13] Sachi Gupta, Sailesh Iyer, Gaurav Agarwal, Poongodi Manoharan, Abeer D Algarni, Ghadah Aldehim, and Kaamran Raahemifar. Efficient prioritization and processor selection schemes for heft algorithm: A makespan optimizer for task scheduling in cloud environment. *Electronics*, 11(16):2557, 2022.

[14] Tao Hai, Jincheng Zhou, Dayang Jawawi, Dan Wang, Uzoma Oduah, Cresantus Biamba, and Sanjiv Kumar Jain. Task scheduling in cloud environment: optimization, security prioritization and processor selection schemes. *Journal of Cloud Computing*, 12(1):15, 2023.

[15] Muxuan Han, Zunfeng Du, Kum Fai Yuen, Haitao Zhu, Yancang Li, and Qiuyu Yuan. Walrus optimizer: A novel nature-inspired metaheuristic algorithm. *Expert Systems with Applications*, 239:122413, 2024.

[16] Mehdi Hosseinzadeh, Marwan Yassin Ghafour, Hawkar Kamaran Hama, Bay Vo, and Afsane Khoshnevis. Multi-objective task and workflow scheduling approaches in cloud computing: a comprehensive review. *Journal of Grid Computing*, 18(3):327–356, 2020.

[17] Latreche Imene, Slatnia Sihem, Kazar Okba, and Batouche Mohamed. A third generation genetic algorithm nsgaiii for task scheduling in cloud computing. *Journal of King Saud university-computer and information sciences*, 34(9):7515–7529, 2022.

[18] Zahra Jalali Khalil Abadi and Najme Mansouri. A comprehensive survey on scheduling algorithms using fuzzy systems in distributed environments. *Artificial Intelligence Review*, 57(1):4, 2024.

[19] Pradeep Krishnadoss and Prem Jacob. Ocsa: Task scheduling algorithm in cloud computing environment. *International Journal of Intelligent Engineering & Systems*, 11(3), 2018.

[20] Sudheer Mangalampalli, Sangram Keshari Swain, and Vamsi Krishna Mangalampalli. Multi objective task scheduling in cloud computing using cat swarm optimization algorithm. *Arabian journal for science and engineering*, 47(2):1821–1830, 2022.

[21] N Manikandan, P Divya, and S Janani. Bwfso: hybrid black-widow and fish swarm optimization algorithm for resource allocation and task scheduling in cloud computing. *Materials Today: Proceedings*, 62:4903–4908, 2022.

[22] N Manikandan, N Gobalakrishnan, and K Pradeep. Bee optimization based random double adaptive whale optimization model for task scheduling in cloud computing environment. *Computer Communications*, 187:35–44, 2022.

[23] Mohammad Masdari, Sayyid Shahab Nabavi, and Vafa Ahmadi. An overview of virtual machine placement schemes in cloud computing. *Journal of Network and Computer Applications*, 66:106–127, 2016.

[24] Abdolkarim Mohammadi-Balani, Mahmoud Dehghan Nayeri, Adel Azar, and Mohammadreza Taghizadeh-Yazdi. Golden eagle optimizer: A nature-inspired metaheuristic algorithm. *Computers & Industrial Engineering*, 152:107050, 2021.

[25] Hardi Mohammed and Tarik Rashid. Fox: a fox-inspired optimization algorithm. *Applied Intelligence*, 53(1):1030–1050, 2023.

[26] Muhammad Faheem Mushtaq, Urooj Akram, Irfan Khan, Sundas Naqeeb Khan, Asim Shahzad, and Arif Ullah. Cloud computing environment and security challenges: A review. *International Journal of Advanced Computer Science and Applications*, 8(10), 2017.

[27] Mohammed Otair, Areej Alhmoud, Heming Jia, Maryam Altalhi, Ahmad MohdAziz Hussein, and Laith Abualigah. Optimized task scheduling in cloud computing using improved multi-verse optimizer. *Cluster Computing*, 25(6):4221–4232, 2022.

[28] Dustin Owens. Securing elasticity in the cloud. *Communications of the ACM*, 53(6):46–51, 2010.

[29] Poria Pirozmand, Amir Javadpour, Hamideh Nazarian, Pedro Pinto, Seyedsaeid Mirkamali, and Forough Ja'fari. Gsaga: A hybrid algorithm for task scheduling in cloud infrastructure. *The Journal of Supercomputing*, 78(15):17423–17449, 2022.

[30] K Pradeep and T Prem Jacob. A hybrid approach for task scheduling using the cuckoo and harmony search in cloud computing environment. *Wireless Personal Communications*, 101:2287–2311, 2018.

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

[32] Mohan Sharma and Ritu Garg. An artificial neural network based approach for energy efficient task scheduling in cloud data centers. *Sustainable Computing: Informatics and Systems*, 26:100373, 2020.

[33] Neetu Sharma, Puneet Garg, et al. Ant colony based optimization model for qos-based task scheduling in cloud computing environment. *Measurement: Sensors*, 24:100531, 2022.

[34] Dawei Sun, Guiran Chang, Lina Sun, and Xingwei Wang. Surveying and analyzing security, privacy and trust issues in cloud computing environments. *Procedia Engineering*, 15:2852–2856, 2011.

[35] Eva Trojovská, Mohammad Dehghani, and Pavel Trojovskỳ. Zebra optimization algorithm: A new bio-inspired optimization algorithm for solving optimization algorithm. *Ieee Access*, 10:49445–49473, 2022.

[36] P Udayasankaran and S John Justin Thangaraj. Energy efficient resource utilization and load balancing in virtual machines using prediction algorithms. *International Journal of Cognitive Computing in Engineering*, 4:127–134, 2023.

[37] Gopika Venu and KS Vijayanand. Task scheduling in cloud computing: a survey. *International Journal for Research in Applied Science & Engineering Technology (IJRASET)*, 8(5):2258–2266, 2020.

[38] Lifei Wei, Haojin Zhu, Zhenfu Cao, Xiaolei Dong, Weiwei Jia, Yunlu Chen, and Athanasios V Vasilakos. Security and privacy for storage and computation in cloud computing. *Information sciences*, 258:371–386, 2014.

[39] Piers Wilson. Positive perspectives on cloud security. *Information Security Technical Report*, 16(3-4):97–101, 2011.

[40] Haitao Yuan, Jing Bi, and MengChu Zhou. Spatial task scheduling for cost minimization in distributed green cloud data centers. *IEEE Transactions on Automation Science and Engineering*, 16(2):729–740, 2018.

[41] Minghai Yuan, Yadong Li, Lizhi Zhang, and Fengque Pei. Research on intelligent workshop resource scheduling method based on improved nsga-ii algorithm. *Robotics and Computer-Integrated Manufacturing*, 71:102141, 2021.

[42] Albert Y Zomaya et al. Parallel and distributed computing handbook. 1996.