INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

ORIGINAL ARTICLE

OPEN ACCESS

Received: 17/05/2025

Accepted: 30/05/2025

Published: 16/06/2025

Citation: Mary JM, George Amalarethinam DI (2025) A Hybrid Model of Gray Wolf Optimization (GWO) and Modified Sunflower Optimization Algorithm (MSOA) for Efficient Task Scheduling in Cloud Computing. Indian Journal of Science and Technology 18(23): 1825-1837. https://doi.org/10.17485/IJST/v18i23.927

* **Corresponding author**.

magegracy@gmail.com

Funding: None

Competing Interests: None

Published By Indian Society for Education and Environment (iSee)

ISSN
Print: 0974-6846
Electronic: 0974-5645

A Hybrid Model of Gray Wolf Optimization (GWO) and Modified Sunflower Optimization Algorithm (MSOA) for Efficient Task Scheduling in Cloud Computing

J Magelin Mary[1]*, D I George Amalarethinam[2]

1 Research Scholar, PG and Research Department of Computer Science, Jamal Mohamed College (Autonomous), Affiliated to Bharathidasan University, Tiruchirappalli, 620020, Tamil Nadu, India
2 Principal and Head, Associate Professor, PG and Research Department of Computer Science, Jamal Mohamed College (Autonomous), Affiliated to Bharathidasan University, Tiruchirappalli, 620020, Tamil Nadu, India

Abstract

Objective: A hybrid GWO-MSOA model is developed for the task scheduling problem. It efficiently approximates solutions for the multi-objective task scheduling problem in a cloud environment. The model optimizes resource utilization, reduces execution time and cost, and improves overall system performance. **Method:** This study combines Gray Wolf Optimization (GWO) and Modified Sunflower Optimization Algorithm (MSOA) for efficient task scheduling in cloud. The GWO-MSOA methodology enhances convergence speed and identifies near-optimal solution. The proposed hybrid model efficiently schedules the task, which reduces makespan and cost, and improves the resource utilization. CloudSim is employed for an extensive performance assessment, evaluating GWO-MSOA with other algorithms for makespan, cost, and resource utilization. The randomly produced dataset, comprising 500 tasks and 50 virtual machines, is utilized to evaluate the performance of the GWO-MSOA. **Findings:** The hybrid GWO-MSOA experimental findings show outstanding makespan, cost, and resource usage performance. The proposed method decreases makespan by 15.05% relative to GWO and by 21.11% relative to MSOA. Furthermore, costs are diminished by 3.46% and 5.86% in comparison to GWO and MSOA, respectively, while resource utilization escalates by 5.69% and 0.46% relative to these methodologies. The results demonstrate that the hybrid GWO-MSOA significantly enhances task scheduling performance regarding makespan, cost efficiency, and resource consumption, establishing it as a viable strategy for cloud computing environments. **Novelty:** GWO is integrated with MSOA, which mitigates entrapment in local minima and early convergence, hence enhancing task scheduling performance. The methodology employs an elite archive that preserves the top-k solutions of the algorithm.

Keywords: Meta-heuristic; Gray wolf optimization; Sunflower optimization; Task scheduling; Cloud

1 Introduction

Cloud computing has fundamentally transformed company operations by facilitating the efficient distribution of computing resources. Cloud users may allocate and deploy Cloud Computing resources according to their needs on a subscription basis using the public interface offered by the Cloud Service Provider. Recent advancements in cloud computing enable a multitude of dispersed and interlinked cloud data centers to provide on-demand services to customers on an as-needed basis more efficiently [1]. The new concept of cloud computing has offered numerous advantages, including decreased infrastructure costs, reduced execution time, and lower maintenance requirements, among others. The heightened workload of cloud computing for processing many applications has consequently led to diminished resource utilization, resulting in a reduced return on investment [2]. A significant cause of diminished cloud computing resource utilization is the improper scheduling of tasks across Virtual Machines (VM), resulting in decreased computational performance. Consequently, work scheduling is crucial in cloud computing for optimizing resource use and delivering satisfactory performance.

Task scheduling allocates user-submitted tasks to appropriate virtual machines within the cloud data center. It significantly influences cloud computing performance, as efficient task scheduling enhances performance and minimizes latency. The task scheduling problem is classified as NP-hard, being a non-deterministic polynomial time problem [3]. The growing complexity of cloud computing has further complicated the resolution of the task scheduling issue, making the development of an effective algorithm increasingly challenging in this environment.

NP-hard problems, such as the task scheduling problem in cloud computing, can be addressed through heuristic and metaheuristic methodologies [4]. Numerous heuristic algorithms have been employed to identify optimal task scheduling within the cloud computing framework. Heuristic approaches yield optimal solutions under specified restrictions. However, the solutions derived from heuristic methods are heavily dependent on established rules and the problem's scale. This approach is excessively costly and extravagant [5].

Metaheuristic approaches offer several solutions rather than a one candidate answer, as is characteristic of heuristic approaches. The performance of metaheuristic-based methods is comparatively superior to that of heuristic methods. Examples of metaheuristics include Genetic Algorithm (GA) [6], Particle Swarm Optimization (PSO) [7], Whale Optimization [8], Gray Wolf Optimization (GWO) [9], and Sunflower Optimization [10]. The Grey Wolf Optimization algorithm surpasses other prominent meta-heuristic algorithms regarding cost and execution time.

Recently many hybrid optimization algorithms [11] are suggested for task scheduling problem. Table 1 delineates significant works in hybrid task scheduling, emphasizing the used methodologies, contributions, and discovered research limitations and gaps that persist. To fill this literature gap, there is a growing interest in incorporating meta-heuristic methods to enhance the efficacy of task scheduling. This study integrates Gray Wolf Optimization (GWO) and the Modified Sunflower Optimization Algorithm (MSOA) for effective task scheduling in cloud computing. The GWO-MSOA methodology improves convergence rate and identifies near-optimal solutions. The suggested hybrid approach effectively schedules tasks, hence minimizing makespan and cost while enhancing resource usage.

The study's substantial contributions are outlined as follows

- A unique hybrid optimization approach has been devised for task schedulers in the cloud computing environment.
- Develop a hybrid scheduling technique to decrease costs, reduce makespan, and enhance resource usage.
- Develop a GWO-MSOA including several operations to tackle the multi-objective task scheduling challenge.
- Evaluate the effectiveness of the suggested work using simulation, then examine and debate the simulation outcomes.

2 Methodology

This section delineates the proposed methodology for multi-objective hybrid task scheduling in cloud computing. The suggested methodology integrates gray wolf optimization with a modified sunflower optimization algorithm for effective work scheduling. This research introduces a novel way for optimizing task resource mapping through the GWO-MSOA technique in a cloud computing environment. To enhance task resource allocation, we seek to optimize the multi objectives of makespan, cost, and resource utilization through the suggested GWO-MSOA methodology. The overall structure of the suggested methodology is illustrated in [Figure 1].

2.1 Multi-Objective Task scheduling

Multi-objective (MO) task scheduling aims to allocate task to cloud resources as efficiently as possible. The task scheduling problem can be articulated as follows:

Table 1. Summary of existing literature

Ref.	Methodology	Contributions	Limitations/Research Gap
[12]	Hybrid grey wolf and whale optimization	It amalgamates the GWO and WOA to improve cloud task scheduling, which minimizes expenses, energy consumption, and time.	It just compares the result with standalone algorithms.
[13]	Hybrid Job scheduling optimization	It integrates Grey Wolf Optimization and Cuckoo Search Optimization to improve resource allocation and work scheduling efficacy.	Further investigation is necessary to determine if the strategy is applicable to diverse massive and practical optimization challenges.
[14]	Hybrid genetic and grey wolf optimization	Implementing mutation and crossover to maintain wolf variety and enhance local search inside the algorithm. An innovative fitness function is introduced that simultaneously considers numerous objectives.	Scalability challenges, such the scheduling of extensive tasks and workload distribution over various clouds, are inadequately addressed.
[15]	Hybrid grey wolf and improved particle swarm optimization	It integrates GWO with PSO to equilibrate the trade-off between the two processes in task distribution.It incorporates chaos and adaptive inertial weight to prevent premature convergence while maximizing convergence speed and global variety.	It fails to equilibrate the traits of local and global search processes to the requisite degree. Absence of fault tolerance feature.
[16]	Gaussian Cloud Whale optimization	It employs opposition-based learning, adaptive adjustment factor, and Gaussian Cloud Whale optimization for task scheduling.	It faces challenges with operational expenses in cloud tasks.
[17]	Hybrid capuchin search and inverted ant colony optimization	It intends to use the advantages of migration and scheduling through hybrid capuchin and inverted ACO algorithms.	A manually established threshold is utilized, rather than an automatic one. Cost, threshold value, and consumption of bandwidth are addressed with limitations.
[18]	Hybrid Moth Swarm and Chameleon Swarm Algorithm	A new scheduling model is created to improve job scheduling efficiency while maintaining data security.	High space and time complexity
[19]	Hybrid grey wolf and cuckoo search optimization	It amalgamates the advantages of both CSO and GWO. It alleviates the constraints of each particular approach (GWO, CSO), hence augmenting the efficiency and efficacy of the resource allocation process.	Additional inquiry is required to ascertain whether the proposed methodology can address many intricate, real-world optimization challenges.
[20]	Hybrid Henry Gas-Harris Hawks Comprehensive Opposition	It addresses multi-objective optimization problems, emphasizing the minimization of makespan and the maximization of resource consumption.	It remains ensnared in local optima. The quantity of work increases as data settings get more complex.

Let consider set of virtual machines V = $\{v_1, v_2, v_3, ..., v_m\}$ and m signifies the entire quantity of VMs in the cloud. Each machine has unique resources and corresponding costs.

Let consider set of Tasks T = $\{t_1, t_2, t_3, ..., t_n\}$, and n signifies the number of tasks conducted on the VM (m). Every assignment comprises a specified quantity of instructions. The number of tasks surpasses the number of VM. The sets T and VM serve as inputs for a scheduling algorithm, and an effective distribution of all tasks among a group of VMs produces a final solution designed to minimize makespan and cost while maximizing resource utilization.

A MO optimization problem often has several fitness (objective) functions represented as $F(x) = [f_1(x), f_2(x), ..., f_k(x)]$, where k denotes the number of objectives, and $f_i(x)$ signifies the i^{th} objective function. This problem lacks a singular solution, and a collection of non-dominated options, referred to as Pareto optimum solutions, can be identified [21]. This work encompasses fundamental objectives of resource utilization, makespan, and cost. These objectives are in conflict, as they aim for the maximization of resource use while simultaneously seeking the minimum of makespan and expense.

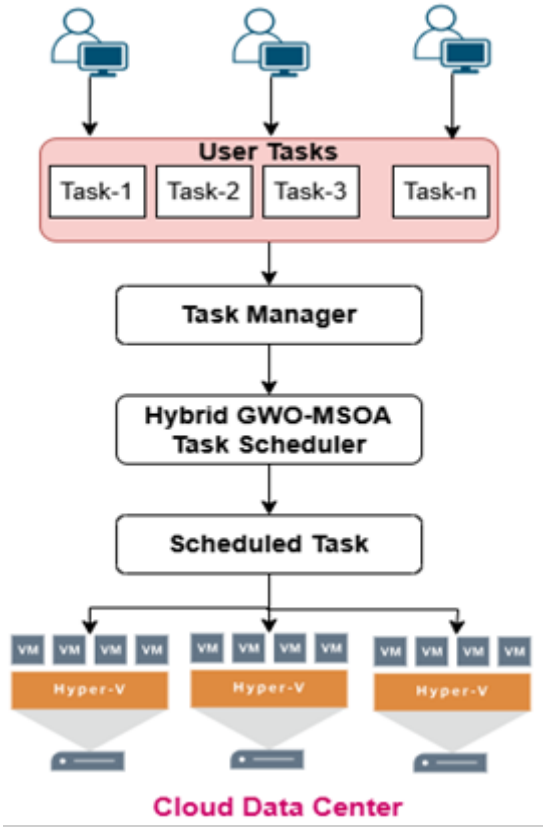

Fig 1. General architecture of proposed approach

2.1.1. Objective-1: Minimize Makespan

The first objective of this study is to reduce the makespan. A crucial element in task scheduling is makespan, which is defined as the aggregate processing time for every task across a designated set of virtual machines. It is calculated as follows:

$$O1\,(Makespan) = Max\,(ET\,(v_i)), \qquad 1 \leq i \leq m \tag{1}$$

where ET indicates the execution time of i^{th} VM. The execution time of VM is computed as,

$$ET\,(v_i) = \sum_{j=1}^{n} \frac{Length(t_j)}{MIPS(v_i)} \tag{2}$$

Where Length(t_j) is the duration of the j^{th} task, measured by the number of instructions (in millions). MIPS(v_i) denotes the processing efficiency of the i^{th} VM, quantified in millions of instructions per second.

2.1.2. Objective-2: Minimize Cost

The second objective is articulated in terms of cost, namely the expense incurred for processing the task, which can be computed using:

$$O2\,(TotalCost) = \sum_{i=1}^{m} cost_i \times ET(v_i)) \tag{3}$$

Where $cost_i$ indicates the i^{th} vm unit cost.

2.1.3. Objective-3: Maximize Resource Utilization

The third objective of this work is to maximize the resource utilization. It is a statistic for quantifying resource utilization. Increased utilization indicates that the cloud provider attains maximum revenue. It can be calculated using,

$$O3\ (RU) = \frac{\sum_{i=1}^{m} ET\,(v_i)}{Makespan \times m} \tag{4}$$

The fitness function for the specified multi-objective task scheduling issue is derived from the weighted sum method applied to each of the objective function. The proposed fitness function (F) can be calculated as follows:

$$F = (\tau 1 \times O1) + (\tau 2 \times O2) + (\tau 3 \times (1 - O3)) \tag{5}$$

Where $\tau 1$, $\tau 2$ and $\tau 3 (\tau 1 + \tau 2 + \tau 3 = 1)$ represents the balance coefficients among makespan, total cost, and RU. The smallest value of the fitness function produces an optimal solution.

2.2 Grey Wolf Optimization

The primary advantage of employing the grey wolf optimizer is its applicability in addressing both limited and unconstrained issues across various domains. This optimizer commences with solutions generated using arbitrary values, without calculating any derivatives of the search space to identify the best solution[22]. This advantage allows the grey wolf optimizer to address real-world issues characterized by unknown or costly derivative information. This optimizer is also beneficial in addressing issues related to the unidentified search space of practical issues.

The Grey Wolf is a member of the canine family. They inhabit communities characterized by a distinct division of labor and cooperative efforts for survival, dependent on hierarchical structures. The dominant grey wolf is referred to as the alpha (α) wolf, the subsequent rank is the beta (β) wolf, the third tier is designated as the delta (δ) wolf, and the lowest rank is the omega (ω) wolf. Alpha serves as the leader, engaged for decision-making about hunting and sleeping schedules, which justifies the designation of alpha as a dominant wolf, as the pack must adhere to its directives. Beta signifies the subordinate wolves that aid the alpha in decision-making and maintaining discipline within the pack. Delta should yield to all alphas and betas while concurrently dominating the omega. Consequently, ω has a subordinate position in the hierarchy and its role is subordinate to all the other dominating wolves.

The primary phases of grey wolf hunting are as follows: Monitoring the prey, surrounding the target until it ceases movement and hunting the prey. This initial phase (tracking the prey) involves the wolves diverging randomly in their pursuit of prey, which may be statistically modeled by calculating the distance among the prey and the dominant wolf.

$$D_v = \left| C_v \times P_p(t) - P_w(t) \right| z \tag{6}$$

D_v denotes the computational distance among the prey and the grey wolf, C_v refers to the coefficient vector, $P_w(t)$ signifies the position of the grey wolf at a specific time interval t, and P_p represents the position of the prey at the same time interval t. The value of C_v can be determined via,

$$C_v = 2 \times rand(1,0) \tag{7}$$

The encircling stage involves prey identification by alpha, beta, and delta wolves, while the remaining wolves adjust their locations at time intervals of t+1 according to the specified equations.

$$P_w(t+1) = P_p(t) - D_v \times A \tag{8}$$

'A' represents as a parameter to modulate the capacities for exploration and exploitation in the optimization process.

$$A = 2 \times a \times rand(0,1) - a \tag{9}$$

Where 'a' indicates convergence parameter and it can be computed as,

$$a = 2 \times \left(1 - \frac{t}{maxIter}\right)^2 \tag{10}$$

Where t indicates current iteration and maxIter represents maximum iteration. During the hunting phase, wolves possess superior awareness of the prey's whereabouts. Consequently, the wolves adjust their positions based on the prey's location

for attacking, designating the top three dominant wolves as alpha, beta, and delta within the hierarchy as the optimal strategy. Consequently, the alpha is capable of attacking the prey. The prey's position is revised according to the locations of the three dominant wolves. Omega randomly updates their coordinates in proximity to the prey.

$$D_\alpha = |c_1 \times P_\alpha(t) \times P(t)| \tag{11}$$

$$D_\beta = |c_2 \times P_\beta(t) \times P(t)| \tag{1}$$

$$D_\delta = |c_3 \times P_\delta(t) \times P(t)| \tag{13}$$

$$P_1 = P_\alpha - a_1 \times D_\alpha \tag{14}$$

$$P_2 = P_\beta - a_2 \times D_\beta \tag{15}$$

$$P_3 = P_\delta - a_3 \times D_\delta \tag{16}$$

$$P = \frac{P_1 + P_2 + P_3}{3} \tag{17}$$

The Greywolf optimizer functions by employing a collection of arbitrary solutions known as grey wolves. A collection of arbitrary solutions is evaluated based on several objective functions[23]. The values of several objective functions signify the quality of each solution. The foremost three quality solutions are identified as alpha, beta, and delta wolves. The Grey Wolf Optimizer consistently adjusts the positions of wolves. If a solution surpasses the alpha, beta, and delta wolves, the relevant solutions are substituted with the identified solution. The Grey Wolf Optimizer ceases iteration of solutions once specific termination conditions are met.

2.3 Proposed GWO-MSOA Task Scheduling

This section introduces a novel methodology for optimizing task resource allocation through the hybrid GWO-MSOA in the cloud computing environment. We aim to enhance task resource mapping by optimizing the many objectives of makespan, cost, and resource consumption with the suggested GWO-MSOA technique. This study integrates GWO with MSOA for efficient task scheduling. The detailed description and algorithm of MSOA is explained in [24]. Algorithm-1 explains the proposed GWO-MSOA.

This algorithm accepts a set of tasks and virtual machines as input. The primary objective of this algorithm is the efficient allocation of incoming tasks to suitable virtual machines while decreasing makespan and cost and maximizing resource utilization. The process begins with the configuration of all parameters, followed by the random generation of the population. Subsequent to the random production of the population, the fitness function was assessed utilizing **Eq. (5)**. The populations are arranged according to fitness value. Identify the alpha, beta, and delta groups of wolves, with alpha representing the lowest fitness rating, beta exceeding alpha but being inferior to other wolves, and delta surpassing beta while still being less fit than the remaining wolves. The others will be classified as omega wolves. The even populations are updated based on GWO using **Eq. (11)** to **Eq. (17)** and compute fitness value for newly updated population. The alpha, beta and delta populations are updated and stored into elite board. The elite board maintains top three populations which has lowest fitness value. The odd populations are updated based on MSOA [24] which includes crossover, mutation and levy flight operations. It requires multiple iterations, after which the optimal solutions will be identified based on the parameters specified in the fitness function. Figure 2 shows the workflow of proposed GWO-MSOA.

Fig 2. Workflow of proposed approach

Algorithm-1 GWO-MSOA Task Scheduling

Input: m virtual machines ($V = \{v_1, v_2, v_3, ..., v_m\}$), n tasks ($T = \{t_1, t_2, t_3, ..., t_n\}$)
Output: Optimal Allocation of Tasks to VMs, Makespan, Cost, Resource Utilization
Step01: Initialize parameters : population size, maximum number of iterations (maxIter), EB = []
Step02: Randomly generate the initial population
Step03: Compute fitness for initial population using **Eq. (5)**
Step04: Sort population based on fitness value (Low to High)
Step05: Select the top three grey wolves as P_α, P_β, and P_δ
Step06: Set it = 0
Step07: While (iter<maxIter) do
Step08: For i = 1 to no_of_population
Step09: If (i % 2 = = 0)
Step10: Update the position of population using **Eq. (11)** to **Eq. (17)**
Step11: Compute fitness for newly updated population using **Eq. (5)**
Step12: Update P_α, P_β, and P_δ
Step13: Update EB and Store P_α, P_β, and P_δ into EB
Step14: Else
Step15: Update the population using MSOA
Step16: Apply Crossover, mutation and Levy Flight operation
Step17: Compute fitness for newly updated population using **Eq. (5)**
Step18: Update P_α, P_β, and P_δ
Step19: Update EB and Store P_α, P_β, and P_δ into EB
Step20: EndIf
Step21: End For
Step22: Update the bestSolution
Step23: iter = iter +1
Step24: End While
Step25: Return the bestSolution

3 Results and Discussion

This section analyzes the experimental outcomes of allocating various tasks over a varied number of virtual machines. The results of the suggested GWO-MSOA are juxtaposed with those obtained from traditional algorithms and state-of-arts approaches in terms of makespan, cost, and resource consumption. The simulation is executed using the cloud computing program CloudSim. In the simulation, the number of tasks and virtual machines varies from 100 to 500 and 10 to 50, respectively. The task length ranges from 1000 to 5000 MI, while for VM, 500 to 4000 MIPS. The population size, and maximum iterations are 10, and 150, respectively.

The metrics of makespan, cost, and resource utilization are employed to evaluate the efficiency of the proposed technique.

Makespan refers to the total time needed to complete a sequence of tasks. Minimizing makespan is a common goal in scheduling algorithms, as it significantly affects the system's effectiveness and performance. The makespan is calculated using **equation (1)**.

Cost refers to the economic considerations associated with the allocation and use of resources in a cloud. Cost reduction is crucial for both service providers and consumers, since it directly impacts the economic efficiency and accessibility of cloud services. The cost is calculated using **equation (3)**.

Resource utilization denotes the optimal and proficient deployment of all accessible resources. A high use value signifies that the cloud provider attains maximum profitability. This can be calculated using **equation (4)**.

The suggested methodology assesses two scenarios: The methodology examines the effects of modifying task sizes from 100 to 500, while maintaining a consistent virtual machine count of 20. We analyze the effects of varying VM sizes between 10 and 50, while maintaining the task count at 400. We evaluate the makespan, cost, and resource utilization performance metrics for both cases. Changing the number of tasks while keeping the virtual machines at 20 yields the following results:

Table 2 illustrates a comparison of makespan for different numbers of tasks. As the number of tasks increases, the makespan similarly rises. The task scheduling utilizing GWO and MSOA demonstrates higher makespan values than GWO-MSOA. The duration of each activity varies based on the virtual machine's capacity, leading to a makespan that differs according to the various approaches employed. The new method exhibits superior results and achieves the minimal makespan relative to the two approaches.

Table 2. Makespan Comparison (Fixed VM and Varying Task)

No of Tasks	Makespan (Sec.)		
	GWO	MSOA	GWO-MSOA
100	6.15	5.73	5.19
200	18.31	19.73	17.29
300	34.38	33.27	30.93
400	47.65	50.13	39.76
500	64.67	75.46	52.23

Figure 3 illustrates the graphical comparison of performance metrics fixed VMs over varying tasks.

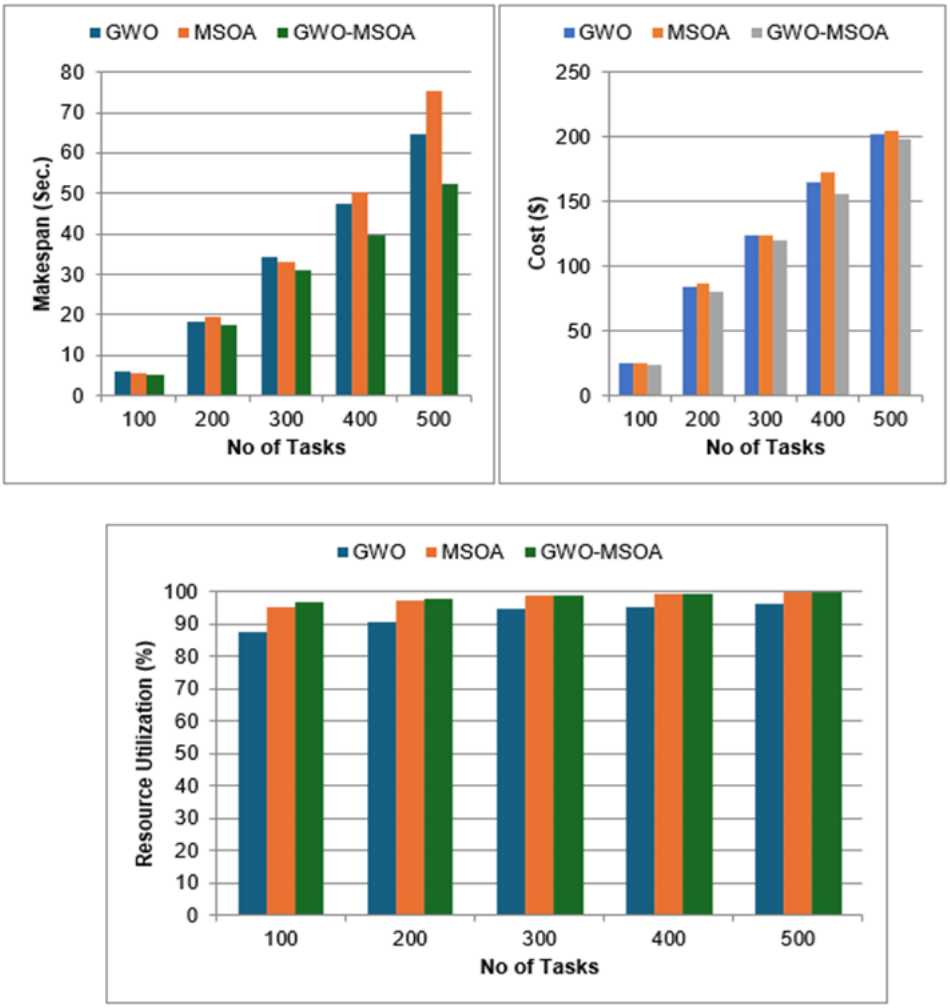

Fig 3. Performance metric comparison for fixed VM and varying No. of tasks

Table 3 demonstrates the effect of a heightened quantity of tasks on processing costs. The duration of each action varies according to the virtual machine's performance, leading to fluctuating expenses dependent on the methods employed. An increase in task quantity correlates with elevated processing costs. The comparative analysis reveals that the proposed strategy yielded the most economical optimal scheduling option.

Table 4 illustrates a comparison of resource utilization. The suggested GWO-MSOA optimizes resource use compared to alternative techniques.

Table 3. Cost Comparison (Fixed VM and Varying Task)

No of Tasks	Cost ($)		
	GWO	MSOA	GWO-MSOA
100	25.45	25.17	23.64
200	83.75	87.37	80.31
300	123.59	124.24	120.06
400	164.83	172.53	155.88
500	201.54	205.11	198.52

Table 4. Resource Utilization Comparison (Fixed VM and Varying Task)

No of Tasks	Resource Utilization (%)		
	GWO	MSOA	GWO-MSOA
100	87.48	95.10	96.68
200	90.66	97.23	97.89
300	94.72	99.01	98.75
400	95.32	99.43	99.55
500	96.49	99.69	99.86

The following findings are provided after modifying the number of virtual machines and setting the task count to 400.

Table 5 demonstrates the effect of the increasing number of VMs on makespan. The makespan for 50 VMs using GWO, MSOA, and GWOMSOA was 14.22, 15.45, and 11.24 seconds, respectively.

Table 5. Makespan (Sec.) Comparison (Fixed Task and Varying VM)

No of VMs	Makespan (Sec.)		
	GWO	MSOA	GWO-MSOA
10	65.52	66.96	62.54
20	47.65	50.13	39.76
30	40.47	43.85	35.61
40	27.13	28.30	23.47
50	14.22	15.45	11.24

The makespan lowers with a spike in the quantity of accessible VMs for execution of tasks. The proposed method demonstrates the minimal makespan relative to the existing methodology. Figure 4 illustrates the graphical comparison of performance metrics for fixed tasks over varying quantities of virtual machines (VMs).

Table 6 illustrates the effect of altering the number of VMs on processing costs. The expenses for 50 VMs for GWO, MSOA, and GWO-MSOA were $245.41, $246.43, and $241.90, respectively.

Table 6. Cost ($) Comparison (Fixed Task and Varying VM)

No of VMs	Cost ($)		
	GWO	MSOA	GWO-MSOA
10	162.89	163.42	134.96
20	170.83	172.53	155.88
30	190.61	188.60	181.53
40	196.15	196.86	191.61
50	245.41	246.43	241.90

Table 7 illustrates a comparison of resource utilization. Augmenting the quantity of virtual machines leads to diminished resource utilization. The resource usage of 50 VMs for GWO, MSOA, and GWOMSOA was 90.53%, 97.01%, and 97.98%, respectively.

Table 7. Resource Utilization (%) Comparison (Fixed Task and Varying VM)

No of VMs	Resource Utilization (%)		
	GWO	MSOA	GWO-MSOA
10	97.16	99.78	99.89
20	95.32	99.43	99.55
30	94.98	98.81	98.90
40	92.19	97.53	98.0
50	90.53	97.01	97.98

Figure 4 illustrates the graphical comparison of performance metrics for fixed tasks over varying quantities of virtual machines (VMs).

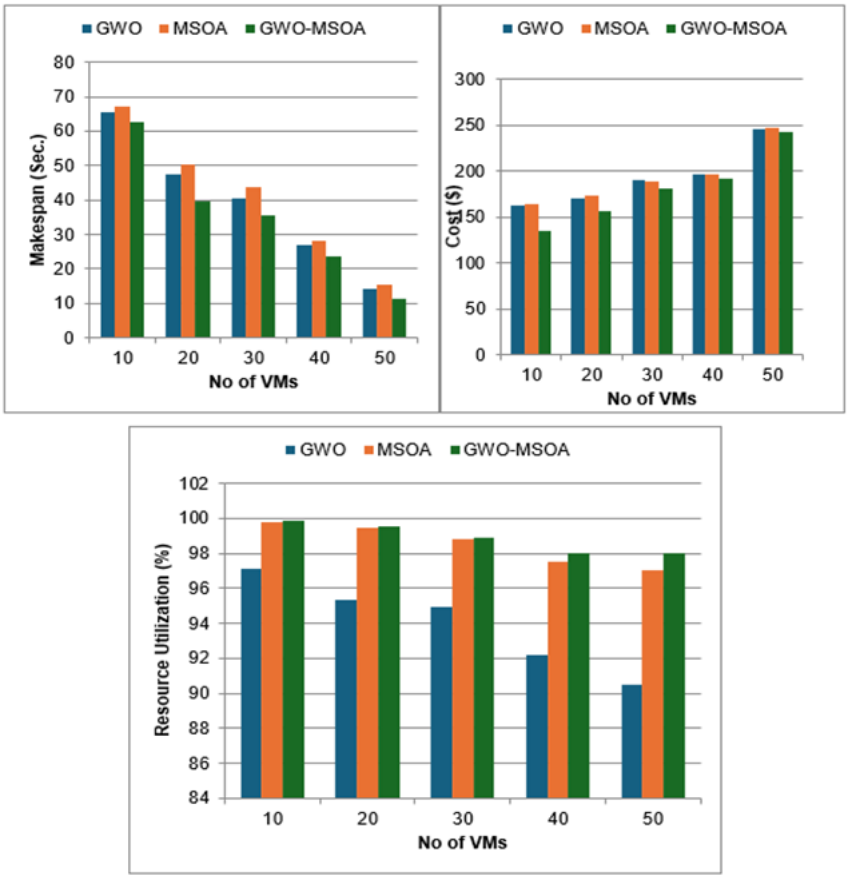

Fig 4. Performance metric comparison for fixed task and varying No. of VMs

The results of the study demonstrate that the proposed scheduling strategy achieved a makespan reduction of up to 15.05%, a cost reduction of 3.46%, and a 5.69% boost in resource utilization compared to GWO. The proposed method exhibits a 21.11% reduction in makespan, a 5.86% drop in cost, and a 0.46% improvement in resource utilization relative to MSOA.

Table 8 shows the comparison of performance metrics for different approaches. The results reveal that the proposed methodology outperforms previous approaches for makespan, cost, and resource use, while also exhibiting enhanced stability and scalability.

Table 8. Performance metrics comparison

Approach	Metrics		
	Makespan (Sec.)	Cost ($)	Resource Utilization (%)
HGWW [12]	80	-	90
MOABCQ [21]	10	76.50	80.1
HES-ACO [24]	60	78.23	85
Proposed GWO-MSOA	5.19	23.64	96.68

4 Conclusion

Task scheduling remains a significant difficulty in cloud computing due to the variability of incoming tasks in terms of size, processing capacity, and the allocation of these tasks to suitable virtual resources. This research introduces a hybrid GWO-MSOA for efficient task scheduling. GWO is incorporated with MSOA, which alleviates entrapment in local minima and premature convergence, hence improving task scheduling performance. The methodology utilizes an elite storage that retains the top-k solutions of the algorithm. The proposed methodology was assessed on the CloudSim platform, and the performance metrics were analyzed. The objective of this work is to enhance resource usage while minimizing makespan and costs. The study's results indicate that the proposed scheduling technique attained a makespan reduction of up to 15.05%, a cost reduction of 3.46%, and a 5.69% increase in resource utilization relative to GWO. The proposed method demonstrates a 21.11% decrease in makespan, a 5.86% reduction in cost, and a 0.46% enhancement in resource usage compared to MSOA. Thus, it can be concluded that the suggested method improves task scheduling performance in terms of makespan, cost, and resource consumption. Future research may expand to develop a deep learning model for efficient task scheduling, integrating additional objectives such as energy consumption, throughput, and degree of imbalance.

References

1) Chiang ML, Hsieh HC, Cheng YH, Lin WL, Zeng BH. Improvement of tasks scheduling algorithm based on load balancing candidate method under cloud computing environment. *Expert Systems with Applications*. 2023;212:118714. https://doi.org/10.1016/j.eswa.2022.118714.
2) Houssein EH, Gad AG, Wazery YM, Suganthan PN. Task scheduling in cloud computing based on meta-heuristics: review, taxonomy, open challenges, and future trends. *Swarm and Evolutionary Computation*. 2021;62:100841. https://doi.org/10.1016/j.swevo.2021.100841.
3) Abualigah L, Diabat A. A novel hybrid antlion optimization algorithm for multi-objective task scheduling problems in cloud computing environments. *Cluster Computing*. 2021;24(1):205–223. https://doi.org/10.1007/s10586-020-03075-5.
4) Al-Arasi R, Saif A. Task scheduling in cloud computing based on metaheuristic techniques: A review paper. *EAI Endorsed Transactions on Cloud Systems*. 2020;6(17):162829. https://doi.org/10.4108/eai.13-7-2018.162829.
5) Motlagh AA, Movaghar A, Rahmani AM. Task scheduling mechanisms in cloud computing: A systematic review. *International Journal of Communication Systems*. 2019;33(6):e4302. https://doi.org/10.1002/dac.4302.
6) Zhou Z, Li F, Zhu H, Xie H, Abawajy JH, Chowdhury MU. An improved genetic algorithm using greedy strategy toward task scheduling optimization in cloud environments. *Neural Computing and Applications*. 2019;32(6):1531–1541. https://doi.org/10.1007/s00521-019-04119-7.
7) Pirozmand P, Jalalinejad H, Asghar A, Mirkamali S, Li Y. An improved particle swarm optimization algorithm for task scheduling in cloud computing. *Journal of Ambient Intelligence and Humanized Computing*. 2023;14(4):4313–4327. https://doi.org/10.1007/s12652-023-04541-9.
8) Zhang Y, Wang J. Enhanced Whale Optimization Algorithm for task scheduling in cloud computing environments. *Journal of Engineering and Applied Science*. 2024;71(1). https://doi.org/10.1186/s44147-024-00445-3.
9) Natesan G, Chokkalingam A. An Improved Grey Wolf Optimization Algorithm Based Task Scheduling in Cloud Computing Environment. *The International Arab Journal of Information Technology*. 2019;p. 73–81. https://doi.org/10.34028/iajit/17/1/9.
10) Emami H. Cloud task scheduling using enhanced sunflower optimization algorithm. *ICT Express*. 2022;8(1):97–100. https://doi.org/10.1016/j.icte.2021.08.001.
11) Malti AN, Hakem M, Benmammar B. A new hybrid multi-objective optimization algorithm for task scheduling in cloud systems. 2023. https://doi.org/10.1007/s10586-023-04099-3.
12) Ababneh J. A Hybrid Approach Based on Grey Wolf and Whale Optimization Algorithms for Solving Cloud Task Scheduling Problem. *Mathematical Problems in Engineering*. 2021;2021:1–14. https://doi.org/10.1155/2021/3517145.
13) Paulraj D, Sethukarasi T, Neelakandan S, Prakash M, Baburaj E. An Efficient Hybrid Job Scheduling Optimization (EHJSO) approach to enhance resource search using Cuckoo and Grey Wolf Job Optimization for cloud environment. *PLOS ONE*. 2023;18(3):e0282600. https://doi.org/10.1371/journal.pone.0282600.
14) Behera I, Sobhanayak S. Task scheduling optimization in heterogeneous cloud computing environments: A hybrid GA-GWO approach. *Journal of Parallel and Distributed Computing*. 2024;183:104766. https://doi.org/10.1016/j.jpdc.2023.104766.
15) Janakiraman S, Priya MD. Hybrid grey wolf and improved particle swarm optimization with adaptive inertial weight-based multi-dimensional learning strategy for load balancing in cloud environments. *Sustainable Computing*. 2023;38:100875. https://doi.org/10.1016/j.suscom.2023.100875.
16) Ni L, Sun X, Li X, Zhang J. GCWOAS2: Multiobjective Task Scheduling Strategy Based on Gaussian Cloud-Whale Optimization in Cloud Computing. *Computational Intelligence and Neuroscience*. 2021;2021:1–17. https://doi.org/10.1155/2021/5546758.
17) Rostami S, Broumandnia A, Khademzadeh A. An energy-efficient task scheduling method for heterogeneous cloud computing systems using capuchin search and inverted ant colony optimization algorithm. *The Journal of Supercomputing*. 2023. https://doi.org/10.1007/s11227-023-05725-y.

18) Alsubai S, Garg H, Alqahtani A. A Novel Hybrid MSA-CSA Algorithm for Cloud Computing Task Scheduling Problems. *Symmetry*. 2023;15(10):1931. https://doi.org/10.3390/sym15101931.
19) Rallabandi VSSSN, Gottumukkala P, Singh N, Shah SK. Optimized efficient job scheduling resource (OEJSR) approach using cuckoo and grey wolf job optimization to enhance resource search in cloud environment. *Cogent Engineering*. 2024;11(1). https://doi.org/10.1080/23311916.2024.2335363.
20) Alkaam NO, Sultan M, Hussin MB, Sharif KY. Hybrid Henry Gas-Harris Hawks Comprehensive-Opposition Algorithm for Task Scheduling in Cloud Computing. *IEEE Access*. 2025;p. 1–1. https://doi.org/10.1109/ACCESS.2025.3530860.
21) Kruekaew B, Kimpan W. Multi-Objective Task Scheduling Optimization for Load Balancing in Cloud Computing Environment Using Hybrid Artificial Bee Colony Algorithm With Reinforcement Learning. *IEEE Access*. 2022;10:17803–17818. https://doi.org/10.1109/ACCESS.2022.3149955.
22) Sanaj MS, Prathap PMJ. An efficient approach to the map-reduce framework and genetic algorithm based whale optimization algorithm for task scheduling in cloud computing environment. *Materials Today: Proceedings*. 2020. https://doi.org/10.1016/j.matpr.2020.09.064.
23) Alsadie D. TSMGWO: Optimizing Task Schedule Using Multi-Objectives Grey Wolf Optimizer for Cloud Data Centers. *IEEE Access*. 2021;9:37707–37725. https://doi.org/10.1109/ACCESS.2021.3063723.
24) Mary JM, Amalarethinam DIG. Modified Sunflower Optimization Algorithm for Task Scheduling in Cloud Computing. *Indian Journal of Science and Technology*. 2025;18(17):1365–1376. https://doi.org/10.17485/IJST/v18i17.537.