Performance Analysis of Multitask Scheduling in Cloud Computing System using Whale Optimization-based Algorithms: A Survey

S. Kavitha
Ph.D. (PT) Research Scholar,
Department of Computer Science, KG College of Arts and Science,
Coimbatore, Tamil Nadu- 641035, India.

G. Paramasivam, PhD
Associate Professor,
Department of Computer Science, KG College of Arts and Science, Coimbatore, Tamil Nadu-641035, India.

ABSTRACT
The rapid growth of cloud computing and task scheduling has become a critical aspect that significantly impacts the overall performance and productivity of cloud-based applications. For service providers and users, inadequate resource management and inefficient task scheduling can result in high cost and resource wastage. In recent years, metaheuristic algorithms have gained prominence for task scheduling in cloud environments. Among them, Whale Optimization Algorithm (WOA) can efficiently explore solution space and optimize complex objective functions. This review paper provides an extensive overview along with the application of WOA-based task scheduling methods in cloud computing. Originally, the WOA principles and operation, which highlight its salient features and optimization capabilities, were reviewed. The existing literature on WOA-based task scheduling methods is reviewed systematically and also the performance of different WOA variants is analyzed. This survey consolidates the current state-of-the-art WOA-based task scheduling methods and also offers insights into their applicability, future directions and performance. It serves as a valuable resource for researchers, practitioners, and educators seeking to understand, evaluate and advance the state-of-the-art cloud task scheduling optimization.

General Terms
Cloud Computing, Task Scheduling

Keywords
Cloud Computing, Multitask Scheduling, Virtual Machines, Metaheuristics, Whale Optimization Algorithm, Quality-of-Service

1. INTRODUCTION
With increasing demand for cloud computing services which has urged a parallel growth in the complexity and the scale of task scheduling, within the challenges in the cloud environment. The overall performance of the cloud system depends upon the optimization of resource utilization, decreasing the response time and effective task scheduling plays a pivotal role. Recently researchers have increasingly used natural-inspired optimization techniques to tackle intricate task scheduling problems in cloud environments. These optimization algorithms are used to increase the utilization and decrease the execution time along with maintaining cost in task scheduling settings [1]. The traditional algorithms used for scheduling lack to cope with intricate and dynamic workloads in the cloud environment. For a single machine, these processes can be challenging and often require computational power, large amounts of memory, and parallel processing capabilities. Due to the intensive computational demands and the need for effective scheduling algorithms become more significant [2]. Among the natural-inspired optimization algorithms, the Whale Optimization Algorithm (WOA) has emerged as a promising approach owing to its ability to mimic the social behaviour of humpback whales during hunting [3]. For addressing the multi-objective and dynamic nature of task scheduling in cloud environments, the WOA exhibits strong exploration and exploitation capabilities, making it well-suited.

Considering various constraints such as task dependencies, resource capacities, and QoS requirements, the WOA algorithm iteratively searches for the optimal mapping of tasks to available cloud resources. Based on the demand cloud resources can be elastically provisioned or de-provisioned and the tasks may be completed at different times or queued for processing. In various works, the WOA has shown the best results and also in different applications based on task scheduling. The nature-inspired approach, integrated with its ability to handle complex limitations and dynamic environments, makes it a valuable tool for optimizing resource utilization and improving the efficiency of workloads. This study aims to evaluate WOA-based task scheduling methods in cloud computing systems and this will help the researchers community to understand the process of WOA and its efficiency and constraints for performing optimum task scheduling in the cloud.

2. WOA
For continuous optimization problems, the WOA algorithm is utilized which is inspired by the social behaviour of humpback whales [3]. The hunting behaviour of the whale is characterized by operational techniques which include encircling prey in the exploitation phase by employing a bubble-net attack method and in the exploration phase searching for prey. Whales with Humpbacks possess the ability to locate prey and coordinate to traverse them. In the WOA, the optimum value in the search space is unknown prior. The hunting behaviour of the whales has collaborated to find the best position. The WOA algorithm operates on the principle that the present best candidate solution is the target prey or is in proximity to the optimum solution. Each search agent in the algorithm will adjust to the most effective search agent from their position. The movement of the search agents towards the best solution is mathematically expressed as:

$$\vec{S} = \left|\vec{C}.\overrightarrow{X^*}(t) - \overrightarrow{X(t)}\right| \quad (1)$$

Here, t refers to the current iteration, $\vec{C}$ refers to the co-efficient vector, $\vec{X}$ refers to the position vector and X^* refers to the position vector of the best solution obtained. The co-efficient vector $\vec{C}$ is calculated using a random vector $\vec{r}$ that ranges from (0, 1):

$$\vec{C} = 2.\vec{r} \quad (2)$$

The movement of a search agent's position (X, Y) can be adjusted based on the present best record's position(X^*, Y^*). By modifying the vector values $A\ and\ C$, various positions near the optimal best agent can be explored to its current position. The attacking method in WOA for the exploitation phase is mentioned as the bubble-net attacking method. This attack method includes two approaches: the shrinking encircling method and the spiral updating position. Humpback whales employ the bubble-net strategy not only for trapping prey but also for attacking. For the creation of a bubble net to trap the prey, this method introduces perturbations around randomly selected solutions within a certain radius. These perturbations encourage the exploration of diverse regions in the solution space, preventing the algorithm from prematurely converging to suboptimal solutions.

The shrinking encircling mechanism refers to the behaviour in Equation (4), where the parameter (a) decreases; the range of fluctuation for a parameter $(\vec{A})$ also decreases. This behaviour signifies a reduction in the intensity of the encircling movement of the whales during the optimization process. The position update using the shrinking encircling method is formulated as follows:

$$\overrightarrow{X}\ (t+1) = \overrightarrow{X^*}(t) - \vec{A}\,.\vec{S} \quad (3)$$

Here, t refers to the current iteration, $\vec{A}$ refers to the co-efficient vector which is calculated as:

$$\vec{A} = 2\vec{a}.\vec{r} - \vec{a} \quad (4)$$

Here, the value of $\vec{a}$ is linearly decreased from 2 to 0 over the iterations on both the exploration and exploitation phases and $\vec{r}$ is the random vector ranges from (0, 1).

The method of updating the position in a spiral manner includes measuring the distance between a whale positioned at coordinates (X, Y) and the prey situated at coordinates(X^*, Y^*).

A spiral equation is formulated to replicate the helical movement observed in humpback whales. This equation guides the gradual adjustment of the whale's position towards the prey, mimicking the behaviour of whales as they navigate through their environment. The helix movement of the whale is given by:

$$\overrightarrow{X}\ (t+1) = \overrightarrow{S'}\,.e^{bl}.\cos(2\pi l) + \overrightarrow{X^*}(t) \quad (5)$$

Here,$\overrightarrow{S'} = \left|\overrightarrow{X^*}(t) - \vec{X}(t)\right|$ and the distance of i^{th} whale to the prey (the best solution obtained for far), b refers to the constant value for defining the shape, $\cos(2\pi l)$ refers to the spiral movement of the whale towards the prey, and l is the random value ranges from [-1, 1]. The probability factor that determines whether to employ the shrinking encircling technique when updating the positions of whales during optimization allows us to emulate the dynamic hunting strategy of whales more realistically within the optimization process. The model is formulated as:

$$\overrightarrow{X}\ (t+1) = \begin{cases} \overrightarrow{X^*}(t) - \vec{A}.\vec{S}, & if\ p < 0.5 \\ \overrightarrow{S'}\,.e^{bl}.\cos(2\pi l) + \overrightarrow{X^*}(t), & if\ p \geq 0.5 \end{cases} \quad (6)$$

Here, p is a random number ranging from [0, 1]. During the exploration phase, the optimization process is similar to the strategy based on adjusting the vector A can be employed. The humpback whales explore randomly based on the positions of each other. To move the search agents far from the reference whale, the A vector is set with the random values between [-1, 1]. The location of a search agent is adjusted based on the location of another search agent selected randomly. This technique along with $|A| \geq 1$, accentuates exploration and enables the WOA algorithm to conduct a thorough global search. The model can be expressed as:

$$\vec{S} = \left|\overrightarrow{C}.\overrightarrow{X_{rand}} - \vec{X}\right| \quad (7)$$

$$\overrightarrow{X}\ (t+1) = \overrightarrow{X_{rand}} - \vec{A}.\vec{S} \quad (8)$$

Here, $\overrightarrow{X_{rand}}$ refers to the random position vector taken from the current population. Algorithm 1 summarizes the WOA.

Algorithm.1. WOA

Population is initiated X_i $(1, 2, 3, \ldots, n)$

Initially, search agent value X^* is selected randomly

The fitness value for each agent is calculated.

Consider X^* as the best search agent.

While $(t < number\ of\ iterations\ (max))$

For search agents, each

Update $a, A, C, l\ and\ p$

If $(p < 0.5)$

If $(|A| < 1)$

Update the position by using eq. (3)

Else if $(|A| \geq 1)$

Random search agent(X_{rand}) is selected

Update current search agent by eq. (8)

End if

Else if $(p \geq 0.5)$

Update current search agent by eq. (5)

End if

End for

Check for search agents beyond the search space and modify

Calculate the fitness value for each search agent

X^* value is updated if there is a better solution

$t = t + 1$

End while

Return X^*.

The WOA initiates a collection of random solutions. The search agents reposition themselves relative to either a randomly selected search agent or the optimal solution found until the iteration. The parameter a is gradually decreased from 2 to 0,

which helps to maintain a balance between the new areas and the solutions. When$|A| > 1$, a random search agent is selected for position update, whereas when$|A| < 1$, the best solution is utilized. Depending on the value of p, the algorithm switches between spiral or circular movement modes. Termination occurs upon meeting a predefined criterion.

3. WOA-based Task Scheduling Methods

Recent studies have been directed for task scheduling based on WOA for efficient resource management in cloud computing systems. Sreenu et al. [4] presented a task scheduling (TS) algorithm called W-Scheduler, which combined a multi-objective model and WOA for cloud computing environments. The multi-objective model measures the fitness value by considering CPU cost, memory cost, makespan, and budget cost. Evaluation results demonstrated that the W-Scheduler achieved a decreased makespan of 7 and a decreased average cost of 5.8. The experiments are conducted with limited resources, which may not accurately represent large-scale cloud environments. Chen et al. [5] proposed a WOA algorithm to unravel the optimization problem and an improved approach called Improved WOA for Cloud task scheduling (IWC). IWC has presented two optimization strategies such as a nonlinear convergence factor and adaptive population size. The IWC approach is implemented and the performance is measured through simulations of up to 10,000 tasks. The IWC achieved a time cost of 897 and a load cost of 1567 for optimal task scheduling plans. It also performed more efficiently at utilizing the system resources for large-scale and small-scale tasks. Ni et al. [6] introduced a multi-objective task scheduling strategy based on Gaussian cloud-whale optimization (GCWOAS2) for cloud computing environments. WOA based on the GCWOA is developed to improve randomness and avoid premature convergence. The GCWOAS2 strategy combined the above contributions to optimize the task and resulted in 0.2623s execution time, 0.14489s running time and 0.3302 CPU rate for 100 tasks. Yet, the model lacks complex scenarios with task dependencies and heterogeneous resources. Jia et al. [7] introduced an improved whale optimization algorithm (IWC-TS) for efficient TS in cloud computing environments. The model introduced an inertial weight strategy and improved WOA's local search ability and convergence. The results demonstrated that the model achieved 0.008$ of economic cost, 1.3ms of time consumption, and 0.55 of load for 1000 tasks. Yet, the model lacks balance in exploration and exploitation capabilities.

Ababneh et al. [8] introduced a hybrid grey wolf and whale optimization (HGWWO) algorithm that integrated the Grey Wolf Optimizer (GWO) and WOA to address the cloud task scheduling problem effectively. The HGWWO algorithm integrated the strengths of both metaheuristics and mathematical models for the fitness function, encompassing makespan, resource utilization, energy, latency, and task weights. The results showed that the HGWWO achieved 95s of makespan, 90% of resource utilization, 9827s of cost and 1923W of energy consumption. The limitation is that the chosen fitness limitations and parameter values directly impact the optimization value quality. Sanaj et al. [9] presented a Map-Reduce Framework and a Genetic Algorithm-based Whale Optimization Algorithm (MRF-GA-WOA). The model used Maximized Raleigh QuoFisher Linear Discriminant Analysis (MRQFLDA) for feature extraction and reduction. The MRF-GA-WOA algorithm increased convergence and optimized the scheduling process by integrating the WOA with the GA. Yet, scalability and robustness for larger or more complex cloud computing environments remain unexplored. Manikandan et al. [10] proposed a hybrid Whale optimization algorithm-based Mutation-based Bee algorithm (MBA) called HWOA-MBA. The fitness function that combines makespan, energy consumption, and total power cost is optimized by using WOA, and the scheduling problem is modelled as a multi-objective optimization problem. The model resulted in 37542s for execution time, 96% for throughput and 1340 for computational cost. The proposed model effectively schedules tasks by considering task and VM priorities while minimizing energy consumption.

Chhabra et al. [11] introduced an energy-aware bag-of-tasks scheduling approach using a Hybrid oppositional Differential Evolution-enabled Whale Optimization Algorithm (h-DEWOA). The algorithm used chaotic maps and OBL techniques to create an initial population, which upgraded the variety of scheduling solutions. The model evaluated with the work logs such as CEA-Curie and HPC2N, the makespan was decreased to 10,488.06s and 11,450.91s, with reduced energy consumption of 6739.39W and 7097.58W for the initial set. Yet, the algorithm exhibited longer execution times. Mangalampalli et al. [12] introduced a prioritized energy-efficient task scheduling algorithm for cloud computing using a whale optimization algorithm (PEETS-WS). The model measured priorities for tasks and VMs based on task length, processing capacity and electricity cost. The approach resulted in 1867.2s for makespan, 4145 for total energy cost and 11.56 for energy consumption. The model lacked testing on real cloud workload traces. Mangalampalli et al. [13] also developed a multi-objective trust-aware TS algorithm (MOTSWAO) using WOA in cloud computing. MOTSWAO assigns task priorities based on task size and VM capacities and sets VM priorities based on the cost of electricity. The MOTSWAO decreased energy consumption by 22-43%, makespan by 17-39%, boosted success rate by 23-68%, process completion efficiency by 29-84%, availability by 13-60%, and total running time by 10-38% for various workloads along with optimizing trust. Yet, the model faced challenges in complexity and computational difficulty in tuning algorithm parameters. Mangalampalli et al. [14] also proposed a SLA-aware TS algorithm in cloud computing using WOA (SLA-WOA). By calculating priorities for tasks and VMs, the algorithm effectively maps high-priority tasks to high-priority VMs, thereby reducing the overall job completion time and the number of SLA breaches. SLA violations are decreased for PSO, ACO, GA, and W-scheduler up to 63.42%, 23.33%, 55.51% and 40.1% for BigDataBench workloads and 56.76%, 42.17%, 35.29% and 24.53% for random workloads. The scheduler WOA not only minimized completion times but also significantly reduced SLA violations.

Chakraborty et al. [15] presented a new variant of the WOA called Elite-Based WOA (EBWOA) to handle the limitations of the conventional WOA, such as restricted exploration capabilities and premature convergence. The model removed the search prey phase and used encircling prey and bubble-net attack phases, which employed a local elite solution during exploration, using a global best solution during exploitation. The model resulted in 10.792s for makespan and 7038.24W for energy consumption, respectively. The model limitation includes it only considering encircling prey and bubble-net attack phases which may restrict exploration. Hosny et al. [16] proposed a Refined Whale Optimization Algorithm (RWOA), an optimizing multi-user dependent task offloading in an edge-cloud computing environment. The model was evaluated based on performance metrics, namely completion time, energy consumption, cost usage, and fitness. RWOA outperformed and achieved optimized fitness by 52.7%. Yet, the model's

Computational complexities improve with the number of tasks and users. Khan et al. [17] presented a Parallel Enhanced Whale Optimization Algorithm (PEWOA) for scheduling independent tasks on heterogeneous VMs in the cloud. PEWOA incorporated an updated encircling manoeuvre, parallelization and an adaptive bubble net attacking technique that enhanced diversity, avoided local optima and improved convergence. The model resulted in 0.74% in resource utilization, 1693.166ms in makespan, 0.605bps in throughput, 0.098s in response time and 1872.500ms in execution time. Gupta et al. [18] introduced a Multi-Objective Whale Optimization-Based Scheduler (WOA-Scheduler) for efficient TS in cloud computing environments. The methodology involved integrating WOA with CloudSim, initializing a whale population representing task allocations, and iteratively updating positions through exploration and exploitation phases and improved solutions. The results indicated the WOA scheduler's effectiveness in adapting to varying cloudlet counts, VM capacities, realistic task distributions, and user-defined objective weights. The model resulted in a cost of 6.625$, execution time of 12.125 seconds and load balancing rate of 1.6154. The model achieved a balance between minimizing cost and execution time while maintaining an acceptable load balance. Yet, the model lacked dynamic resource provisioning strategies.

Table 1 summarizes the advantages and disadvantages of the discussed WOA-based task scheduling methods. The discussion of WOA-based task scheduling methods from the above previous research works highlights the strengths and weaknesses of WOA compared to the other variants of WOA algorithms in this context.

Table 1. Comparison of WOA-based Task Scheduling Methods

Authors	**Method**	**Advantages**	**Disadvantages**
Sreenu et al. [4]	W-Scheduler	Minimized makespan and cost.	Poor resource utilization and energy consumption.
Chen et al. [5]	IWC	Better resource management.	Slow convergence.
Ni et al. [6]	GCWOAS2	Reduced task execution time, and improved resource utilization.	Poor handling of task dependencies and heterogeneous resources.
Jia et al. [7]	IWC-TS	Reduced response time.	Imbalance between exploration and exploitation capabilities.
Ababneh et al. [8]	HGWWO	Better makespan, cost, and resource utilization.	Poor selection of fitness constraints and parameter values for faster convergence.
Sanaj et al. [9]	MRF-GA-WOA	Improved the convergence rate	Less scalable for complex cloud computing environments.
Manikandan et al. [10]	HWOA-MBA	Effective handling of VM priorities and minimized power consumption.	Ignores the total time for scheduling the task.
Chhabra et al. [11]	h-DEWOA	Reduced energy consumption.	Longer execution times.
Mangalampalli et al. [12]	PEETS-WS	Prioritized task and VM mapping with reduced energy for data centres.	Less effective on dynamic workload traces.
Mangalampalli et al. [13]	MOTSWAO	Minimized task completion time and execution time with faster convergence.	High memory utilization and less effective on dynamic workloads.
Mangalampalli et al. [14]	SLA-WOA	Minimized makespan and SLA breaches.	High computational complexity.
Chakraborty et al. [15]	EBWOA	Faster convergence and better balance in exploitation.	Neglects the exploration phase.
Hosny et al. [16]	RWOA	Improved multi-user dependent task offloading.	High computational complexity.
Khan et al. [17]	PEWOA	Higher scalability.	Ineffective for dynamic environment.
Gupta et al. [18]	WOA-Scheduler	Better balanced trade-off between decreasing cost and execution time while maintaining an acceptable load balance.	Less suitable for dynamic resource provisioning strategies.

It is inferred that the WOA-based hybrid optimization methods provided better task scheduling performance since they solved the problem of convergence at local optimum solutions. However, the complexity of many WOA-based algorithms is a major problem. Also, the complexity of WOA for handling the scheduling of a larger number of tasks is a concerning limitation.

4. PERFORMANCE COMPARISON

The discussed research works have performed experiments for justifying efficient task scheduling using CloudSim 3.0.2. The workload, involving a combination of interactive and batch processing tasks, encompasses tasks associated with managing and processing accounting records for the 128 nodes. Details include the runtime, number of nodes, user commands and start time. The commonly used performance metrics are Energy Consumption, Execution Time, Response Time, and Cost. The available results for the metrics by the discussed methods are summarized in Table 2.

The results summarized in Table 2 show that the WOA-based task scheduling methods were efficient in task scheduling for different scenarios. Based on the comparison results in those research studies, the WOA-based task scheduling methods outperformed the other task scheduling methods, especially the other metaheuristic algorithms-based task scheduling methods. Results and analysis illuminated the performance of each WOA variant concerning predefined metrics. Variants showcased diverse strengths and weaknesses, influenced by factors such as convergence speed, solution quality, and computational efficiency. Contextualizing findings within task scheduling optimization underscored the practical implications of WOA-based algorithms, yet limitations and avenues for further research were acknowledged. Hybridization emerged as a promising avenue for enhancing WOA-based task scheduling methods. By integrating WOA with complementary techniques such as local search heuristics or problem-specific strategies, hybrid methods have the potential to mitigate weaknesses and capitalize on strengths inherent in both WOA and its counterparts. When compared to standalone methods of the WOA variant, the hybrid model provides higher performance, increased computational efficiency and robustness across diverse problem instances.

The research studies conducted experiments in various scenarios, involving different numbers of tasks and virtual machines (VMs). The results summarized in Table 2 indicate that WOA-based task scheduling methods generally performed well across a range of scenarios, with specific metrics such as energy consumption, execution time, response time, and cost being key performance indicators. Energy consumption was a critical metric in scenarios where it was measured. For instance, HGWWO [8] and EBWOA [15] showed significant energy usage, highlighting the importance of energy efficiency in cloud environments.

Table 2. Performance Results of WOA-based Task Scheduling Methods

Methods	No. of Tasks	No of VMs	Energy (W)	Execution time (s)	Response time (s)	Cost ($)
W-Scheduler [4]	400	40	-	7.76	8.25	5.8
IWC [5]	1000	40	-	3.876	2.334	0.22
GCWOAS2 [6]	1000	40	-	0.2623	4.865	-
IWC-TS [7]	1000	100	-	0.543	1.3	100
HGWWO [8]	500	32	1.923	125.6	95	188
MRF-GA-WOA [9]	500	50	-	255	220	-
HWOA-mBA [10]	100	40	-	336.57	375.42	1340
h-DEWOA [11]	400	48	6739.39	120.05	104.88	-
PEETS-WS [12]	1000	100	11.56	964.2	103.42	4145
MOTSWAO [13]	1000	30	112.34	119.1	95.62	-
SLA-WOA [14]	1000	20	-	200.45	189.95	1653
EBWOA [15]	100	10	7038.24	12.34	10.792	-
RWOA [16]	200	10	362.36	18.765	21.05	5.33
PEWOA [17]	1000	32	-	0.098	187.25	-
WOA-Scheduler [18]	50	5	-	12.125	11.453	265.3

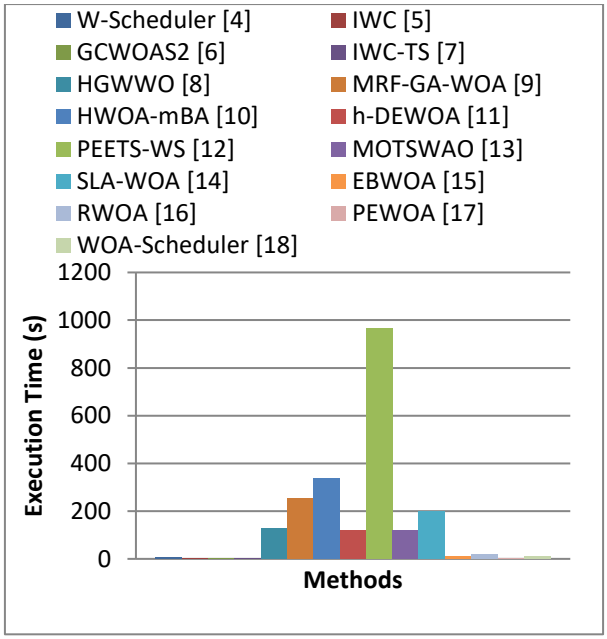

Figure.1. Execution Time Comparison

In Figure 1, execution time varied significantly among the methods. For example, IWC-TS [7] achieved a notably low execution time (0.543s) for 1000 tasks, showcasing the efficiency of the method in handling large workloads.

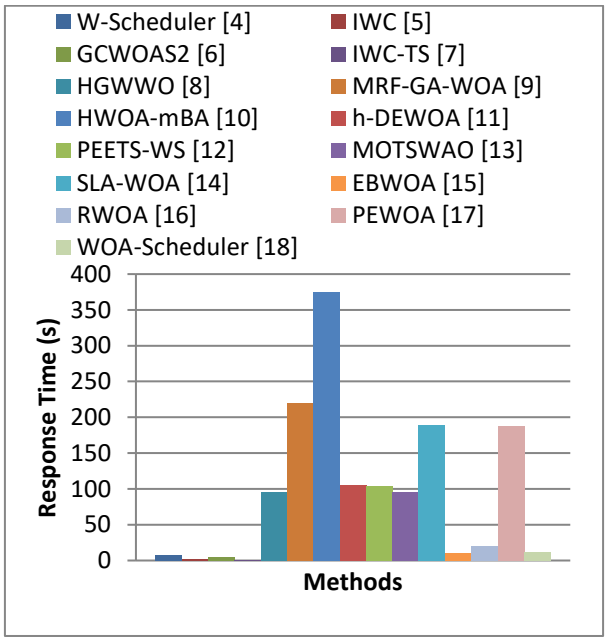

Figure.2. Response Time Comparison

Response time was another crucial metric. Methods like IWC [5] and GCWOAS2 [6] demonstrated relatively low response times, indicating their effectiveness in minimizing delays in task processing. Cost was a variable factor across different methods, with some like IWC [5] achieving a minimal cost of $0.22, while others like PEETS-WS [12] incurred much higher costs, reflecting the trade-offs between performance and economic efficiency.

The analysis indicates that WOA-based methods generally outperformed other metaheuristic algorithms in task scheduling efficiency. For example, methods like GCWOAS2 [6] and IWC-TS [7] delivered superior performance in terms of execution and response times, making them suitable for scenarios requiring rapid task processing. The various WOA variants displayed different strengths and weaknesses, often influenced by factors like convergence speed, solution quality, and computational efficiency. For instance, while some methods like MRF-GA-WOA [9] showed excellent convergence in terms of execution time, others like HWOA-mBA [10] struggled with significantly higher execution times.

The research highlighted hybridization as a promising strategy to enhance WOA-based methods. By integrating WOA with complementary techniques such as local search heuristics or problem-specific strategies, hybrid models like MOTSWAO [13] were able to mitigate some of the inherent weaknesses of standalone WOA, offering improved performance, computational efficiency, and robustness across various problem instances.

The findings underscore the practical utility of WOA-based task scheduling methods in cloud computing environments, particularly in scenarios requiring optimized resource allocation and task management. However, the performance of these methods can vary depending on specific workload characteristics and environmental factors. Despite the promising results, the research also acknowledged limitations in the existing WOA-based methods, such as the potential for high energy consumption and the need for improved cost efficiency. Future research could explore further hybridization, dynamic adaptation techniques, and the application of WOA in emerging cloud computing paradigms like edge computing architectures.

5. CONCLUSION

This review paper has given a detailed overview of utilizing standalone WOA variants and hybrid WOA methods in task scheduling within cloud environments. Through an elaborate and extensive study and examination of various WOA methods in task scheduling, several observations and insights have been developed. These adaptations exhibit the versatility of WOA and its potential to suit varying requirements and constraints of cloud-based scheduling scenarios. Against the traditional optimization algorithm, the different WOA-based approaches have illustrated the competitive performance in solution quality, convergence speed, adaptability to dynamic environments and robustness. These models have demonstrated their effectiveness in reducing execution time, minimizing energy consumption, improving resource utilization, and optimizing cost-related objectives. In cloud computing systems, the WOA has great potential for optimizing and enhancing the performance of task scheduling methods. The insights from the survey serve as a valuable resource and information for practitioners and researchers to guide the development of innovative scheduling methods that harness the full potential of WOA for optimizing resource utilization, enhancing the system performance and advancing techniques in cloud computing.

While the current applications of WOA in cloud computing task scheduling are impressive, several avenues for future research and development remain unexplored. Future research could focus on developing advanced hybrid models that integrate WOA with emerging optimization techniques, such as machine learning-based predictive models or quantum-inspired algorithms. These integrations could enhance WOA's ability to handle increasingly complex and dynamic cloud environments. As cloud environments continue to grow in size and

complexity, there is a need to scale WOA-based methods effectively. Research could explore distributed and parallelized WOA algorithms capable of handling large-scale task scheduling in real-time, ensuring that performance gains are maintained even as the system scales. Developing dynamic adaptation mechanisms that allow WOA-based models to adjust their strategies in real-time based on changing workloads or system conditions could further enhance their applicability in highly volatile cloud environments. Future studies could focus on refining WOA-based methods to align with green computing initiatives, emphasizing energy efficiency and sustainability. This could involve optimizing energy consumption not just at the task level, but across the entire cloud infrastructure, including data centers and network resources. Combining WOA with cost-effective energy models could lead to the development of task scheduling algorithms that balance performance and environmental impact, offering cloud providers more sustainable and economically viable solutions. As cloud systems become more complex, ensuring security and reliability in task scheduling is crucial. Future research could explore how WOA-based methods can be extended or hybridized with security-focused algorithms to address potential vulnerabilities and ensure reliable task execution. Enhancing the resilience of WOA-based scheduling models against system failures or attacks could be another promising direction, ensuring continuous and reliable service in cloud environments.

6. REFERENCES

[1] Arunarani, A. R., Manjula, D., and Sugumaran, V. 2019. Task scheduling techniques in cloud computing: A literature survey. Future Generation Computer Systems, 91, 407-415.

[2] Houssein, E. H., Gad, A. G., Wazery, Y. M., and Suganthan, P. N. 2021. Task scheduling in cloud computing based on meta-heuristics: review, taxonomy, open challenges, and future trends. Swarm and Evolutionary Computation, 62, 100841.

[3] Mirjalili, S., and Lewis, A. 2016. The whale optimization algorithm. Advances in engineering software, 95, 51-67.

[4] Sreenu, K., and Sreelatha, M. 2019. W-Scheduler: whale optimization for task scheduling in cloud computing. Cluster Computing, 22, 1087-1098.

[5] Chen, X., Cheng, L., Liu, C., Liu, Q., Liu, J., Mao, Y., and Murphy, J. 2020. A WOA-based optimization approach for task scheduling in cloud computing systems. IEEE Systems Journal, 14(3), 3117-3128.

[6] Ni, L., Sun, X., Li, X., and Zhang, J. 2021. GCWOAS2: multi-objective task scheduling strategy based on Gaussian cloud-whale optimization in cloud computing. Computational Intelligence and Neuroscience, 2021, 1-17.

[7] Jia, L., Li, K., and Shi, X. 2021. Cloud computing task scheduling model based on improved whale optimization algorithm. Wireless Communications and Mobile Computing, 2021, 1-13.

[8] Ababneh, J. 2021. A hybrid approach based on grey wolf and whale optimization algorithms for solving cloud task scheduling problem. Mathematical Problems in Engineering, 2021, 1-14.

[9] Sanaj, M. S., and Prathap, P. J. 2021. 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, 37, 3199-3208.

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

[11] Chhabra, A., Sahana, S. K., Sani, N. S., Mohammadzadeh, A., and Omar, H. A. 2022. Energy-aware bag-of-tasks scheduling in the cloud computing system using hybrid oppositional differential evolution-enabled whale optimization algorithm. Energies, 15(13), 4571.

[12] Mangalampalli, S., Swain, S. K., and Mangalampalli, V. K. 2022. Prioritized energy efficient task scheduling algorithm in cloud computing using whale optimization algorithm. Wireless Personal Communications, 126(3), 2231-2247.

[13] Mangalampalli, S., Karri, G. R., and Kose, U. 2023. Multi-Objective Trust aware task scheduling algorithm in cloud computing using Whale Optimization. Journal of King Saud University-Computer and Information Sciences, 35(2), 791-809.

[14] Mangalampalli, S., Swain, S. K., Karri, G. R., and Mishra, S. 2023. SLA Aware Task-Scheduling Algorithm in Cloud Computing Using Whale Optimization Algorithm. Scientific Programming, 2023.

[15] Chakraborty, S., Saha, A. K., and Chhabra, A. 2023. Improving whale optimization algorithm with elite strategy and its application to engineering design and cloud task scheduling problems. Cognitive Computation, 1-29.

[16] Hosny, K. M., Awad, A. I., Khashaba, M. M., Fouda, M. M., Guizani, M., and Mohamed, E. R. 2023. Optimized multi-user dependent tasks offloading in edge-cloud computing using refined whale optimization algorithm. IEEE Transactions on Sustainable Computing.

[17] Khan, Z. A., Aziz, I. A., Osman, N. A. B., and Nabi, S. 2024. Parallel Enhanced Whale Optimization Algorithm for Independent Tasks Scheduling on Cloud Computing. IEEE Access.

[18] Gupta, S., and Singh, R. S. 2024. User-defined weight-based multi-objective task scheduling in cloud using whale optimization algorithm. Simulation Modelling Practice and Theory, 102915.