STM JOURNALS

Research & Reviews:
Discrete Mathematical Structures

ISSN: 2394-1979
Volume 13, Issue 2, 2026
May–August
DOI Journal: 10.37591/RRDMS

https://journals.stmjournals.com/rrdms

Research RRDMS

Task Scheduling in Cloud Computing using Hippopotamus Optimization Algorithm

Rekha S.[1,*], Bhargavi K.[2], Dinesha H.A.[3]

Abstract

Cloud computing, which provides remote clients with on-demand services, has emerged as a crucial component of contemporary technology. It is still difficult to schedule tasks effectively in such diverse and dynamic situations. Motivated by the hippopotamus's balanced exploration and exploitation behavior, this research suggests a unique work scheduling method utilizing the hippopotamus optimization algorithm (HOA). In order to maximize resource usage and throughput while minimizing makespan and execution cost, the suggested HOA-based scheduler dynamically assigns jobs to virtual machines. As demonstrated by simulation results, HOA outperforms other existing methods like the intelligent and interpretable rule-based metaheuristic task scheduling (IRMTS), the henry gas–harris hawks comprehensive-opposition (HGHHC) algorithm, and the optimised AI-Driven swarm-based enhanced task scheduling model (OASE-TSM) in terms of execution time, load balancing, and throughput. HOA is ideal for large-scale and real-time cloud systems because to its adaptability and lightweight design. Future research will concentrate on expanding HOA's scalability for sustainable cloud computing and adding energy-aware scheduling.

Keywords: Cloud computing, task scheduling, hippopotamus optimization algorithm (HOA), resource utilization, makespan, throughput, load balancing, metaheuristic algorithms, real-time scheduling, energy- aware scheduling

INTRODUCTION

One of the more important technological developments in modern computing is cloud computing. Using this cloud technology we can solve on demand requests from the remote clients. Nowadays, this cloud computing becomes a part of our daily life. Hence to utilize the resource in cloud becomes difficult because of dynamic and heterogeneous nature of cloud environments [1]. We have to build a model which will reduces the execution cost and the task completion time that is our motto. Accurate managing and scheduling client tasks has become a major concern [2]. In a cloud context, task scheduling is the process of allocating certain tasks to virtual machines in order to minimizes the execution time, maximizes resource utilization, minimizes the execution cost and increases the throughput [3].

This job scheduling is categorized as an optimization problem that is NP-hard [4]. Traditional task scheduling method not addressing the intricacy and fluctuating cloud tasks [5]. This drawback has led to the increasing use of metaheuristic algorithms, which is the true inspiration from the natural and biological strategy to provide nearest solutions within a time frame [6]. The latest addition to nature inspired techniques is the hippopotamus optimization algorithm. This method was developed based on the behavior of hippopotamuses, as their approach to locating food

***Author for Correspondence**
Rekha S.
E-mail: rekhanisarga@gmail.com

[1]Student, Department of Computer Science and Engineering Siddaganga Institute of Technology, Tumakuru, Karnataka, India
[2]Assistant Professor, Department of Computer Science and Engineering Siddaganga Institute of Technology, Tumakuru, Karnataka, India
[3]Assistant Professor, Department of Computer Science and Engineering Siddaganga Institute of Technology, Tumakuru, Karnataka, India

Received Date: May14, 2026
Accepted Date: May 19, 2026
Published Date: May 30, 2026

Citation: Rekha S., Bhargavi K., Dinesha H.A. Task Scheduling in Cloud Computing using Hippopotamus Optimization Algorithm. Research & Reviews: Discrete Mathematical Structures. 2026; 13(2): 22–29p.

involves a balanced exploration of the food [7]. And also in dangerous situations like some animal attacks to this hippopotamus it will exploit the strategy in efficient way. And its behavior in aquatic and on land is highly remarkable [8].Hence we could say HOA mimics hippopotamus behavior in balancing explore and exploit is interesting, and we are able to adapt in task scheduling to get optimal solution.Task scheduling using HOA algorithm approach can upgrade the system's performance by minimizing the makespan, execution time, maximize the resource utilization and system throughput more effectively [9]. Compare to traditional approach this HOA based approach more convenient and emerged as a powerful optimization technique in cloud environment.

Furthermore, HOA has excellent flexibility in dynamic cloud environments, effectively managing diverse workloads and heterogeneous resources [10]. HOA optimizes work allocation, system scalability, and reliability by skillfully striking a balance between exploration and exploitation [11]. Because of this, HOA is especially well-suited for real-time applications and expansive cloud infrastructures, where performance depends on reducing latency and maximizing resource utilization [12]. Additionally, HOA ensures more consistent optimization outcomes by lowering the likelihood of local optima trapping.

RELATED WORK

In [13], Kaur et al. presented a task scheduling model that selects virtual machines according to essential resources using neural networks and optimization. By learning from input information and producing initial task-resource mappings, the Artificial Neural Network (ANN) models the link between tasks and available resources. By determining the best starting weights prior to training, adjusting the ANN's hyperparameters, or immediately optimizing the task scheduling output after the ANN creates an initial schedule, Moth Flame Optimization (MFO) improves the ANN's performance and leads to more effective and flexible job distribution. Through the reuse of learnt solution patterns, the method drastically cuts down on execution time. However, still rely significantly on meticulous parameter adjustment and lacks validation on real-time cloud systems, which could restrict its adaptability and scalability in real-world deployments, even though it successfully handles convergence issues in MFO.

In [14], Barut et al. presented a rule-based task scheduling approach that make use of machine learning and previous metaheuristic methods to make faster judgments in time- sensitive cloud environments. During the semi-offline stage, metaheuristic algorithms are used to address job scheduling difficulties, and the results are saved in a pattern archive. These are then grouped together, and interpretable rules are extracted using machine learning.When a new issue emerges during the online phase, the system instantly chooses the optimal solution pattern from the archive using these principles, allowing cloud systems to schedule tasks quickly and efficiently. Through the reuse of learnt solution patterns, the method drastically cuts down on execution time. However, IRMTS may have trouble adjusting to completely new job scenarios that aren't addressed in the pre-generated solution archive, which would restrict its responsiveness in extremely dynamic or unpredictable contexts, even while it improves efficiency and interpretability.

In [15], Mangalampalli et al. employed a technique for cloud job scheduling known as grey wolf optimization. The Multi- Objective task scheduling grey wolf optimization (MOTSGWO) algorithm optimizes cloud work scheduling by imitating the hunting behavior and social hierarchy of grey wolves. It finds the top three leaders, comes up with possible solutions, and keeps others informed by following them. This iterative approach makes real-time scheduling decisions and adjusts to shifting cloud circumstances. The method outperformed more conventional techniques like Ant Colony Optimization (ACO), GA, PSO, and Cuckoo Search (CS) when evaluated in CloudSim and confirmed using both synthetic and real-world workload traces. However, the study does not take network latency, security, or fault tolerance into account, is restricted to simulation without real-world deployment, and lacks scalability analysis. Furthermore, its adaptability in extremely dynamic cloud environments may be impacted by static parameter values and the absence of dynamic tweaking.

In [16], Alkaam et al. created a hybrid scheduling method HGHHC that blends three cloud computing strategies. Harris Hawks Optimization (HGSO) aids in investigating a broad variety of possible task schedules, but Harris Hawks Optimization (HHO) acts as a local search to refine answers and avoid getting caught up in poor schedules. By considering their opposite values, Comprehensive Opposition-Based Learning (COBL) further enhances weaker solutions, improving the likelihood of discovering superior ones. When combined, these methods enable more efficient job distribution to virtual machines by balancing exploration and exploitation. The algorithm was evaluated using cloudsim with synthetic and real-world workloads, showing better performance than existing approaches. However, although the technique showed encouraging results in simulations, it is not validated in the real world, does not handle dynamic scheduling or energy usage, and provides little information about computational overhead and scalability.

In [17], Abualigah et al. suggested a hybrid algorithm, Improved Aquila Optimizer (IAO) that blends cloud-based particle swarm optimization with the Aquila Optimizer. During the scheduling process, a transition mechanism is employed to intelligently switch between Aquila Optimizer (AO) and Particle Swarm Optimization (PSO) based on the diversity of solutions and the algorithm's progress. The algorithm was tested across multiple task scenarios (600– 2000 tasks) and outperformed standard methods like Genetic Algorithm (GA), PSO, AO, and Differential Evolution (DE) in terms of expected completion time. However, while IAO showed improved scheduling performance, the study lacks real-world cloud deployment, does not evaluate energy efficiency or resource utilization, and focuses narrowly on completion time without analyzing broader system-level impacts.

PROPOSED HOA ALGORITHM FOR TASK SCHEDULING IN CLOUD COMPUTING

Figure 1 illustrates a cloud-based task scheduling architecture that effectively assigns and manages computing jobs by utilizing the proposed hippopotamus optimization algorithm. Initially, a number of users (Users 1 through i) submit their tasks, which are compiled into a cloud task list. They are then placed in a task queue and given time to be processed. The central decision-making component of the system is the HOA Task Scheduler, where virtual agents called "hippos" (hippo 1 to hippo P) develop several possible scheduling methods (Schedule 1 to Schedule P).The two primary operations that the scheduler does to hone these methods are explore, which looks at a significant quantity of potential outcomes, and exploit, which improves the most promising ones.

To gauge its effectiveness, each method is put through a fitness assessment that considers variables like processing time and resource usage. The best option has been identified as the cloud broker receiving the most efficient schedule. This component controls resource allocation and ensures load balancing by allocating the jobs to the relevant virtual machines (VM1 to VMm). Essentially, by proactively modifying task allocations to maximize cloud resource utilization and minimize latency, the HOA-driven approach improves overall system performance.

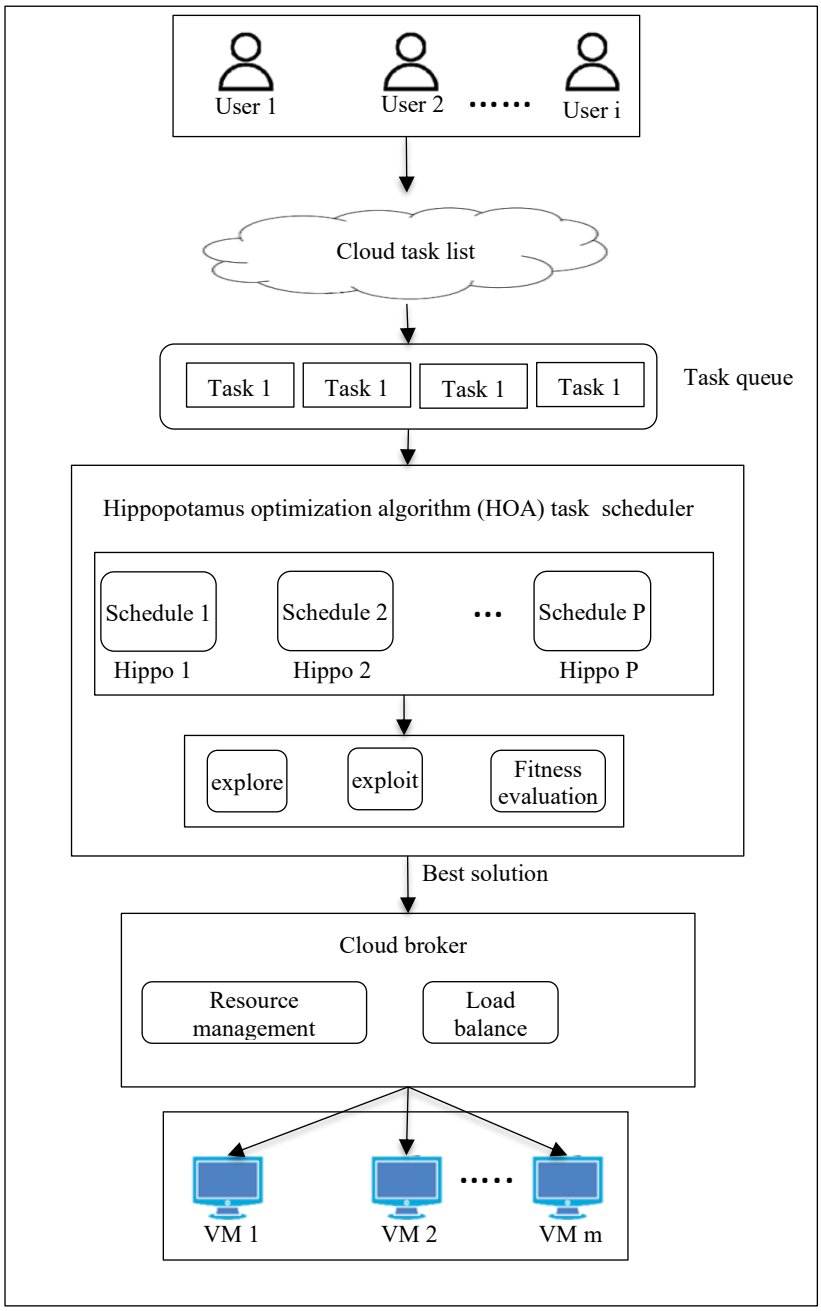


Figure 1. Proposed architecture of HOA based task scheduling system in cloud computing

Algorithm 1. Working of the proposed HOA algorithm for task scheduling in cloud computing
1: **Start**
2: **Input** T, V, P, I_{max}, $r1$, $r2$, α, β, γ, δ
// T: Tasks, V: Virtual Machines, P: Population size, I_{max}: Max iterations
//r_1 , r_2: Control parameters; α, β, γ, δ: Weights for fitness objectives
3: **Output:** X_{best} , the best task scheduling solution with related metrics.
4: Initialize a population of P hippopotamuses (schedules)
$X = \{X_1, X_2, \ldots, X_P\}$
- Each $X_i = [X_1, X_2, \ldots, X_n]$, where $X_j \in \{1, \ldots m\}$ ($task\ T_j \rightarrow VM_{xj}$)
5: $for\ i = 1\ to\ P\ do$
6: Evaluate Fitness $F(X_i)$ using:

$Makespan(X_i) = max\sum(W_j/P_k)$ over all VMs
$Cost(X_i) = \sum(ExecutionTime\_VM_k \times Cost\_rate_k)$
$Energy(X_i) = (ExecutionTime\_VM_k \times Energy\_rate_k)$
$LoadBalance(X_i) =$
$standard\ deviation\ of\ VM\ workloads$
$F(X_i) = \alpha \times Makespan + \beta \times Cost + \gamma \times Energy + \delta \times LoadBalance$
7: ***if*** $F(X_i) < F(X_{best}) then$
8: $X_{best} \leftarrow X_i$ // Update the best solution discovered thus far.
9: **end if**
10: **end for**
11: ***for*** $iteration = 1\ to\ I_{max}\ do$
12: ***for*** $i = 1\ to\ P\ do$
13: Produce a random number $r \rightarrow [0,1]$
14: ***if*** $r < 0.5\ then$
15: // Phase of Exploration
16: ***for*** $j = 1\ to\ n\ do$
17: **// Encourage variety occasionally by assigning people at random**
18: ***if*** $rand() < 0.3 then$
19: $X_i[j] = random\ integer\ in\ [1, m]$
20: **else**
21: $X_{i[j]} = X_{i[j]} + r_1 \times (X_{besy[j]} - X_{i[j]}) + r_2 \times rand()$
22: $X_{i[j]} = round(X_{i[j]})\ and\ clamp\ to\ [1, m]$
23: **end if**
24: **end for**
25: **else**
26: // Phase of Exploitation
27: ***for*** $j = 1\ to\ n\ do$
28: $\epsilon \leftarrow small\ random\ number\ \in [-1,1]$
// Adjust the solution in light of the optimal c ourse of action.
29: $X_{i[j]} = X_{i[j]} + \varepsilon \times sign(X_{i[j]} - X_{best[j]})$
30: $X_{i[j]} = round(X_{i[j]}) and\ clamp\ to\ [1, m]$
31: **end for**
32: **end if**
33: Re-evaluate Fitness $F(X_i)$ (with LoadBalance term)
34: ***if*** $F(X_i) < F(X_{best})\ then$
35: $X_{best} \leftarrow X_i$ // Update if improved solution found 36: **end if**
37: **end for**
38: **end for**
39: **Return** X_{best}, $Makespan(X_{best})$, $Cost(X_{best})$, $Energy(X_{best})$, $LoadBalance(X_{best})$
// The optimal final result along with all important assessment metrics
40: **Stop**

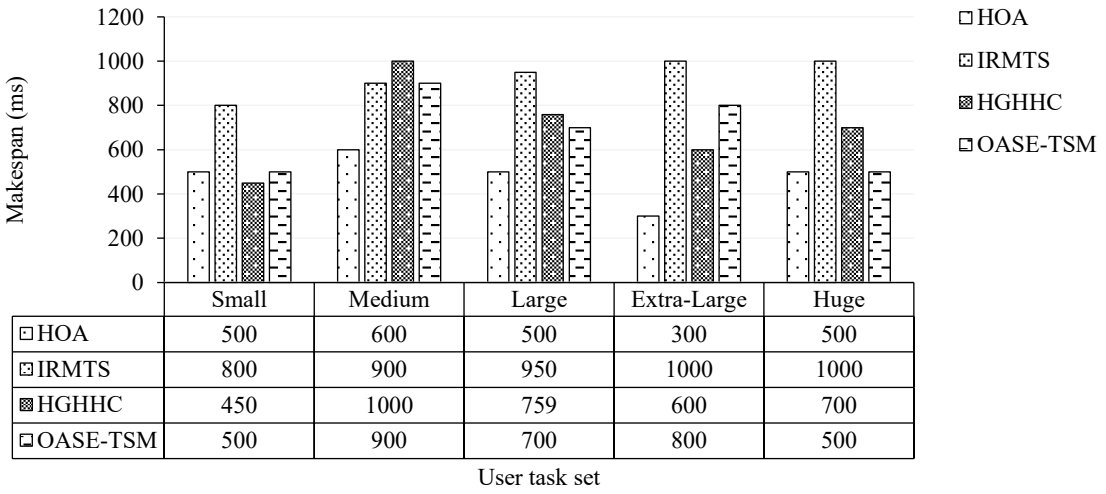
RESULTS

Figure 2 illustrates the relative effectiveness of several scheduling strategies with respect to makespan. By carefully balancing search and swiftly modifying job distribution with its hybrid approach and dynamic updates, HOA lowers makespan. IRMTS results in an increased makespan because to its

excessive number of rules, complexity, and delayed adaption. Because it is difficult to tweak and computationally expensive, the HGHHC algorithm has a longer makespan. OASE-TSM's greater convergence time and scalability problems result in a bigger makespan.

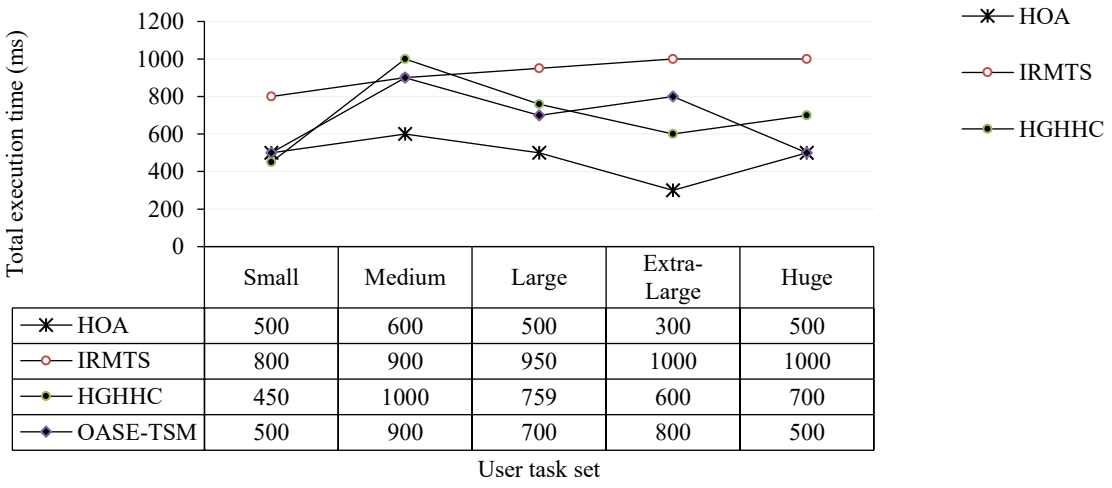
Figure 3 compares the execution times of several scheduling approaches. Fast task scheduling in HOA is made possible by low computational complexity and fast convergence, which shortens execution times. Longer execution times in IRMTS are caused by limited generalization and local optimization risk. Because of the possibility of premature convergence and scalability problems with large or dynamic task sets, HGHHC execution times are longer. OASE-TSM's complicated implementation and significant resource usage result in longer execution times.

Figure 4 shows the resource usage of several scheduling techniques. Distributed execution, task priority awareness, and a high degree of solution variety all improve resource usage in HOA. Complex debugging and a heavy knowledge engineering burden in IRMTS result in less efficient use of resources. HGHHC uses fewer resources due of reproducibility overhead, a limited scope of the evaluation, and a lack of QoS. OASE-TSM's risk of overfitting and unpredictable performance under dynamic loads result in decreased throughput.



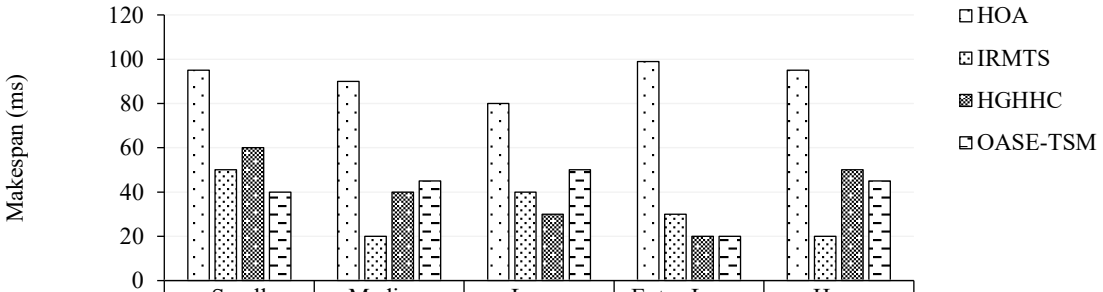
	Small	Medium	Large	Extra-Large	Huge
HOA	500	600	500	300	500
IRMTS	800	900	950	1000	1000
HGHHC	450	1000	759	600	700
OASE-TSM	500	900	700	800	500

Figure 2. Comparative makespan performance of different scheduling techniques.



	Small	Medium	Large	Extra-Large	Huge
HOA	500	600	500	300	500
IRMTS	800	900	950	1000	1000
HGHHC	450	1000	759	600	700
OASE-TSM	500	900	700	800	500

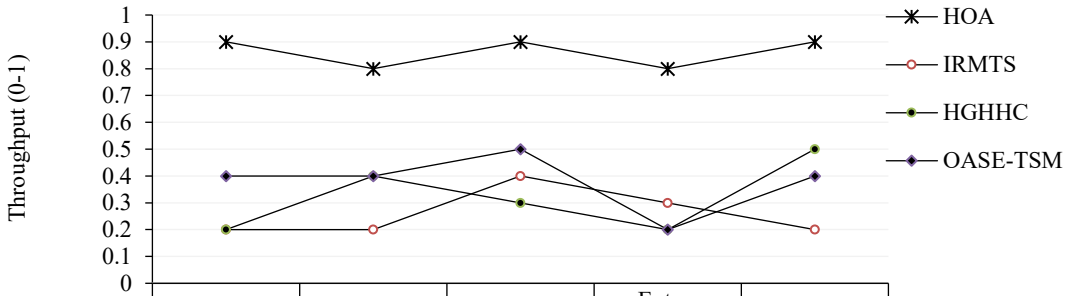
Figure 3. Comparative execution time performance of different scheduling techniques.



	Small	Medium	Large	Extra-Large	Huge
HOA	95	90	80	99	95
IRMTS	50	20	40	30	20
HGHHC	60	40	30	20	50
OASE-TSM	40	45	50	20	45

User task set

Figure 4. Comparative resource utilization performance of different scheduling techniques.



	Small	Medium	Large	Extra-Large	Huge
HOA	0.9	0.8	0.9	0.8	0.9
IRMTS	0.2	0.2	0.4	0.3	0.2
HGHHC	0.2	0.4	0.3	0.2	0.5
OASE-TSM	0.4	0.4	0.5	0.2	0.4

User task set

Figure 5. Comparative throughput performance of different scheduling techniques.

In Figure 5, the throughput performance of various scheduling schemes is shown. Throughput in HOA is improved by increased energy efficiency, smooth cloud integration, and strong resilience to changing workloads. IRMTS is less suited for real-time applications, which restricts how quickly it can complete tasks. Furthermore, the high expenses of implementation and upkeep further diminish efficiency, which lowers throughput. Throughput is decreased in HGHHC due to load imbalance on heterogeneous resources and challenges managing dynamic jobs. OASE-TSM has reduced throughput due to latency in decision-making and maintenance complexity.

CONCLUSION

The proposed HOA works better than current methods like IRMTS, HGHHC, and OASE-TSM regarding cloud job scheduling. HOA uses cloud resources effectively, dynamically adjusts to changing conditions, and strikes a balance between exploration and exploitation. According to experimental results, HOA increases throughput, lowers makespan, executes faster, and uses more resources. These enhancements are the result of HOA's strong load balancing system, flexible scheduling approach, and lightweight architecture. HOA is a viable option for real-time and large- scale cloud computing systems because it prevents early convergence, minimizes computational overhead, and guarantees appropriate

task-resource mapping, in contrast to traditional and hybrid techniques. In order to improve sustainability and adaptability in big cloud infrastructures, future work will concentrate on incorporating energy-aware scheduling and real-time scalability advancements into the HOA architecture.

REFERENCES

1. Kumar MS, Reddy KG, Donthi RK. SSKHOA: Hybrid metaheuristic algorithm for resource aware task scheduling in cloud-fog computing. Int J Inf Technol Comput Sci. 2024 Feb;16(1):1–13. doi:10.5815/ijitcs.2024.01.01.
2. Sa'ad S, Muhammed A, Abdullahi M, Abdullah A. An optimised cuckoo-based discrete symbiotic organisms search strategy for tasks scheduling in cloud computing environment. arXiv [Preprint]. 2023 Nov. Available from: arXiv:2311.15358.
3. Abdulrazzaq DR, Shati NM, Hoomod HK. Task scheduling in a cloud environment based on meta-heuristic approaches: A survey. Iraqi J Sci. 2024 Feb;65(2):33–45.
4. Amiri MH, Hashjin NM, Montazeri M, Mirjalili S, Khodadadi N. Hippopotamus optimization algorithm: A novel nature-inspired optimization algorithm. Sci Rep. 2024 Feb;14(1):5032.
5. Amiri MH, Hashjin NM, Montazeri M, Mirjalili S, Khodadadi N. Hippopotamus optimization algorithm: A novel nature-inspired optimization algorithm. Sci Rep. 2024 Feb;14(1):5032.
6. Pei S, Sun G, Tong L. An improved hippopotamus optimization algorithm based on adaptive development and solution diversity enhancement. 2025.
7. Abualigah L, Diabat A, Sumari P, Gandomi AH, Mirjalili S. Hippopotamus optimization algorithm: A new nature-inspired metaheuristic optimization algorithm. Comput Ind Eng. 2023;179:109298.
8. Alkaam A, Abualigah L, Diabat A, Mafarja H. A hybrid task scheduling algorithm based on Henry's gas solubility optimization, Harris hawks optimization and comprehensive opposition-based learning for cloud computing. J Grid Comput. 2024;21(1):1–27.
9. Abualigah L, Mafarja H, Diabat A, Mirjalili S, Gandomi AH. Recent advances and applications of metaheuristic algorithms in cloud computing: A comprehensive review. Cluster Comput. 2023;26:1297–1325.
10. Kumar S, Verma A. An efficient task scheduling algorithm based on improved genetic algorithm in cloud computing environment. Comput Electr Eng. 2022;95:108080.
11. Aljarah I, Faris H. Optimizing task scheduling in cloud computing using hybrid metaheuristic algorithms. Appl Soft Comput. 2023;136:110102.
12. Jena RK. Task scheduling in cloud environment using soft computing techniques: A survey. J King Saud Univ Comput Inf Sci. 2023;35(3):321–339.
13. Kaur S, Singh J, Bharti V. An optimised AI-driven swarm-based enhanced task scheduling model for cloud computing environment. Int J Cloud Comput. 2025;14(1):25–53.
14. Barut C, Yildirim G, Tatar Y. An intelligent and interpretable rule-based metaheuristic approach to task scheduling in cloud systems. Knowl Based Syst. 2024;284:111241.
15. Mangalampalli S, Karri GR, Kumar M. Multi objective task scheduling algorithm in cloud computing using grey wolf optimization. Cluster Comput. 2023;26(6):3803–3822.
16. Alkaam NO, Sultan AM, Hussin MB, Sharif KY. Hybrid Henry Gas-Harris Hawks comprehensive-opposition algorithm for task scheduling in cloud computing. IEEE Access. 2025.
17. Abualigah L, Elaziz MA, Khodadadi N, Forestiero A, Jia H, Gandomi AH. Aquila optimizer based PSO swarm intelligence for IoT task scheduling application in cloud computing. In: Integrating meta-heuristics and machine learning for real-world optimization problems. Cham: Springer International Publishing; 2022. p. 481–497.

