Al-Rafidain Journal of Computer Sciences and Mathematics (RJCSM)

www.csmj.uomosul.edu.iq

Enhancing Cloud Resource Allocation with a Multi-Objective SARSA Whale Optimization Algorithm (MO_SAWOA)

Hiba Qasim[1], Huda Abd Alali[2] and Soukaena Hassan[3]

[1,2]College Of Computer Science, Mustansiriyah University, Baghdad, Iraq
[3]College Of Computer Science, University of Technology, Baghdad, Iraq
Email: hibaqasimjaber@uomustansiriyah.edu.iq[1] ,huda_it@uomustansiriyah.edu.iq[2] and soukaena.h.hashem@uotechnology.edu.iq[3]

Article information

Article history:
Received 14 April, 2026
Revised 20 May, 2026
Accepted 28 May, 2026
Published 25 June, 2026

Keywords:
Cloud computing, reinforcement learning, resource allocation.

Correspondence:
Hiba Qasim
Email: hibaqasimjaber@uomustansiriyah.edu.iq

Abstract

As we know, cloud computing has become an increasingly popular solution to deliver scalable and effective solutions to the needs of today's modern applications. One of the greatest problems in this context is the efficient allocation of resources to the different needs of numerous users and applications. Reinforcement learning (RL) is an approach that can adapt dynamically to changing environments and is a promising solution. In this paper, the SARSA (state action reward state action) RL algorithm is combined with the whale optimization algorithm (WOA) to propose a new resource allocation approach for cloud computing based on a multi objective optimization. We are going to refer to this method as MO SAWOA. It assists the SARSA algorithm to faster converge and prevents it from being trapped in the local optima. The proposed system handles complex multi objective resource allocation issues by improving load balancing, reducing makespan, saving costs, and enhancing user experience in dynamic cloud environments. It can also be used for real time systems (RTS) like public clouds like Amazon Web Services (AWS).

DOI: 10.33899/rjcsm.v20i1.60674,

1. Introduction

The research Dynamic resource allocation is an important aspect of cloud system, where a system can automatically allocate and scale resources according to the current load and requests [1]. Cloud computing enables customers worldwide to access computing, storage, and network services seamlessly, using web browsers and any kind of devices [2][3]. In the field of cloud computing, resource allocation has been one of the most researched topics and many researchers have made substantial contributions to solve this important problem. Real time changes in workload are too dynamic to be tackled using conventional allocation strategies, like heuristic or rule-based approaches. This results in inefficiencies that impact system performance, load balance, cost and execution time [4].

Dynamic load balancing is a technique used mainly to enhance cloud performance by optimizing work scheduling and distribute workloads across virtual machines (VMs). The idea is that it is dynamically allocating resources based on the computed load on the VMs. To maximize the resource allocation efficiency, the proposed approach is a hybrid between reinforcement learning (RL) and whale optimization algorithm (WOA). The aim is to balance the load efficiently with the quality of service (QoS) metrics like makespan optimization, cost savings, balanced CPU usage, efficient memory utilization and bandwidth management. The proposed approach is implemented in Java using the CloudSim toolkit.

(1) presented in this thesis tackles the following problems:

- Adaptive load balancing in cloud system.
- Balancing competing goals like lower costs, faster delivery times and optimal resource utilization.

- Solve implementation difficulties associated with reinforcement learning (RL) and hybrid algorithms.
- Providing flexibility in managing the growing load and expanding out cloud services.

(2) Its objectives are to achieve the following:

- Optimize task scheduling and load distribution in cloud.
- Improve the efficiency of the cloud in general.
- Balance conflicting quality of service (QoS) requirements while at the same time effectively load balancing.
- Make use of the whale optimization algorithm (WOA) to improve the allocation system by integrating RL with it.

Java and CloudSim will be used to validate the effectiveness of the proposed model.

(3) The following is the paper's primary novelty:

- In this paper, we introduce MO SAWOA, a novel integration of SARSA algorithm with Whale Optimization Algorithm (WOA), to solve a three objective cloud resource allocation problem of minimizing makespan, cost and load imbalance. MO-SAWOA differs from the earlier RL–metaheuristic hybrids which use RL only as an approximation to the policy, and those which use metaheuristics only to re optimize occasionally, by continuously coupling a SARSA based critic with the search process of WOA. This makes it flexible to provide an adaptive steering of the swarm towards Pareto optimal resource allocations even in the presence of non-stationary workloads.
- Multi objective approach with weights that are dynamically adapted to deal with multiple conflicting objectives. The Wahle optimization algorithm (WOA) was optimized by reinforcement learning to allocate resources in cloud computing so that it saves cost, enhances load balance, and improves resource efficiency.
- Saving cost for both user and provider.

2. Related Works

Improving resource allocation and load distribution is an NP-hard problem [5]. Many studies have proposed optimization strategies for cloud environments across various areas, including system power consumption, scheduling, load balancing techniques to reduce execution time, and resource allocation.

The researchers in this study employed a reinforcement learning (RL) algorithm to optimize the load distribution and save energy. The results obtained by the proposed system using the actor-critic (AC) algorithm indicated that it would be possible to decrease the workload by about 20%, thereby preserving the quality of service [6].

in this study, reinforcement learning techniques (Q learning and linear regression) were employed to develop an effective resource allocation system for cloud computing environment. The results of the study show that the proposed approach can help to achieve a significant reduction in operational costs, with a saving of about 77.7% and a reduction of 30% in service violations [7].

This study proposed a hybrid approach for resource allocation in cloud environment by combining the particle swarm optimization (PSO) and whale optimization algorithm (WOA). The approach pursued multiple objectives: reducing energy consumption, processing time, and costs. The Sipht dataset was used to test the proposed method. The proposed system was better than Round Robin and ant colony optimization (ACO) algorithms [8].

The Wa3C algorithm has been used for resource allocation and the proposed multi objective system tries to consider energy consumption as well as response time. The results obtained with the simulated workloads demonstrate that the proposed resource allocation system improves energy consumption and response time [9].

MO-SAWOA is different in several aspects than the approaches described above. Unlike existing techniques, which either employ RL for approximation of the policy, or apply metaheuristics for infrequent re-optimization, it adaptively uses SARSA-based critic and whale optimization algorithm (WOA) continuously to guide the search process under non-stationary workloads. MO-SAWOA particularly focuses on three goals: makespan, cost and load imbalance, and, as stated, obtains dynamic weight adjustment to deal with conflicts between these goals, but this is crucially dependent on good tuning of the exploration-exploitation trade-off.

3. Mathematical Formulations as a Markov Decision Process

In In cloud data centers, the optimization problem of dynamic resource allocation is represented as:

3.1 A Markov- Decision- Process (MDP):

The (S, A, R) defines the MDP in which:
S: is mean the state of containing by the proposed system, such as physical device and VM.
A: action containing potential decisions such as cloudlet assignment to vm.
R: current reward achieved for doing action in state(s).
The goal is to identify the best policy (π^*) that make higher expected of cumulative reward:

$$\pi^* = \text{argmax}\ \pi\ [\Sigma^t\ \gamma^t R(s^t, a^t)] \ldots. (1)$$

where the discount element is represented by $\gamma \in [0, 1]$. This equation makes it possible to apply reinforcement learning techniques to discover the best resource allocation strategies that decrease makespan and cost waste while preserving excellent performance and load balancing (LB) of data centers.

3.2 Objective function

Multi-objective optimization problems typically involve multiple fitness functions. Several non-dominated (Pareto-based) approaches exist for solving such challenges, but there is no single perfect solution [10]. In this paper, we consider the following three objectives:

The first objective is makespan, also known as total task execution time, which is the time the system takes to complete all submitted tasks (i.e., the completion time of the last task).

$$MS = \max (CT) \quad (2)$$

where CT denotes cloudlet completion time [11].

The second objective is the cost incurred by processing a job, which can be determined from the costs of CPU, memory, and bandwidth usage. Equation (3) calculates the total cost of the virtual machines.

$$\text{Cost}_z^r = \text{Cost memory} + \text{Cost storage} + \text{Cost cpu} + \text{Costs BW} \quad (3)$$

where (Cost_z^r) represent the Cost of vms for rth make a request for zth user Cost memory indicates the memory- cost, (Coststorage) demonstrates the storage cost, (Costcpu) shows processing unit cost and CostBW denotes the bandwidth cost [12].

In a heterogeneous cloud computing system, load balancing is crucial for achieving optimal resource usage and quality of service (QoS), which is the third objective. Load balancers help allocate resources to workloads in a fair and balanced manner, thereby maximizing resource utilization and customer satisfaction at the lowest possible cost [13].

$$AL = \frac{1}{M}\sum_{i=1}^{M} Lvmj \quad (4)$$

Where M means the total number of (VMs).

3.3 Fitness function:

is produced by computing each individual fitness function's weighted sum approach [14]. (5) displays the suggested fitness function (F):

$$F = (Y1 * F1) + (Y2 * F2) + (Y3 * F3) \quad (5)$$

where the makespan (MS), cost, and load balance are denoted by Y1, Y2 and Y3, where $Y \in [0,1]$.

4. PROPOSE SYSTEM METHODOLOGY

The propose system comprises four layers cloud data centers shown in **Figure 1**.

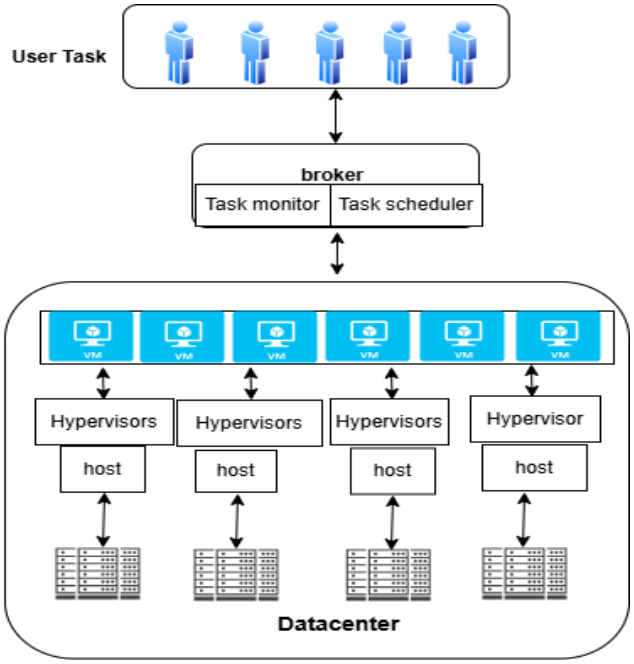


Figure 1. The proposed resource allocation system.

Users submit requests via the internet. After the tasks are sent to the broker, it distributes them among virtual machines in a way that ensures efficient load balancing, maximizes resource utilization at the lowest cost, and provides the fastest response time. Once executed, the tasks are sent back to the broker to fulfill the user's request. The three most commonly used evaluation criteria are loading balancing among virtual machines, makespan, and cost. In this work, we use the SARSA and MO-SAWOA approach to help the system predict and make decisions on the best way to optimize resource allocation.

4.1 Reinforcement Learning (RL)

One such method for adaptive resource management is reinforcement learning (RL), a branch of machine learning (ML). Unlike traditional rule-based or static resource allocation methods, reinforcement learning algorithms allow systems to learn and adjust their resource management strategies over time based on environmental feedback. The trade-off between exploration and exploitation is a challenge specific to reinforcement learning [15][16]. Figure 2 illustrates the general structure of RL.

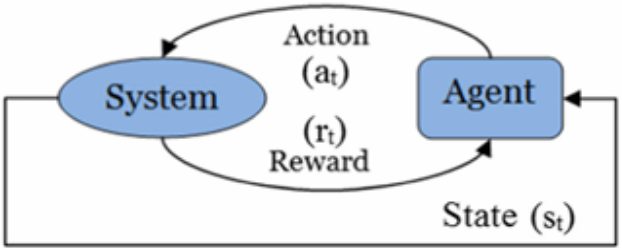


Figure 2. Interaction of a (reinforcement learning) agent with its environment.

4.1.1 Sarsa Algorithm

The SARSA algorithm is an example of an on-policy reinforcement learning method. When λ = 0, it functions similarly to the Q-learning algorithm. Equation (6) shows the update rule for SARSA:

$$Q(st, at) = Q(st + at) + \propto [r + \Upsilon . Q(st + 1, at + 1) - Q(st + at)] \quad (6)$$

At state s_t, the agent selects an action, transitions to state s_{t+1}, and receives a reward r_{t+1}. This process repeats over multiple episodes until the agent reaches the terminal state.

4.2 Whale Optimization algorithm (WOA)

It is a novel meta-heuristic strategy inspired by natural phenomena, specifically the hunting patterns of humpback whales. The process involves locating the target and encircling it while moving in a spiral pattern (resembling the number 9). The WOA technique [17] consists of three steps:

- search for prey:

Depending on the prey's location, whales search for it randomly. This corresponds to the exploration phase of the algorithm, where the whale optimization algorithm (WOA) performs a global search when A≥1. The following mathematical model represents this initial movement:

$$D = |C . x_{rand} - X(t)| \quad (7)$$

$$X(t + 1) = x_{rand} - A . C \quad (8)$$

Where:
D represents the distance between the current whale and a randomly selected whale. A and C are coefficient vectors. X(t) is the current position of the whale, and X_ "rand" is the random whale's position. The vectors A and C can be computed according to equations:

$$A = 2a . r - a \quad (9)$$

$$C = 2 . r \quad (10)$$

To enable switching between exploration and exploitation, a variable a is used, which decreases linearly from 2 to 0, adjusting the balance between exploration and exploitation.

- Encircling the prey

Once the whales have located and identified their prey, they start to surround them.

$$X_i^{t+1} = X_i^{*t} - A . D \quad (11)$$

where X is the position vector and X_i^(t+1) is the position of the best solution found thus far.

- Bubble net attack:

As it spirals, the whale eventually shrinks the bubble net to catch its prey. This motion is modeled using equation (12):

$$X^{ti+1} = \begin{cases} X^{ti} - A . D \; If \; p < 0.5 \\ D . e^{bl} . \cos(2\pi l) + X^{ti} \; if \; p > 0.5 \end{cases} \quad (12)$$

where b is to determine the spiral's form and D is the i-th whale's space from the prey (the best solution found thus far). A random number among [-1,1] is called l. Additionally, the whales emit air bubbles that ascend to the surface of the seawater, as seen in **Figure 3.**

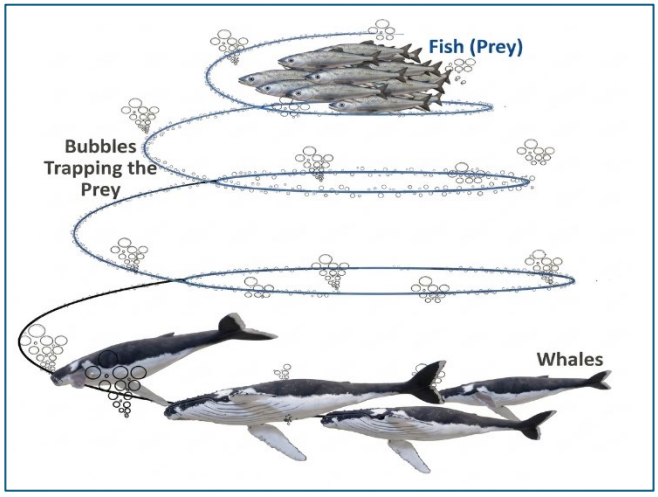


Figure 3. Whale movement and bubble net.

4.3 Hybrid ALGORITHM

The hybrid SA-WOA (MO-SAWOA) approach solves a multi-objective cloudlet-to-VM allocation problem by combining two well-known optimization concepts: SARSA (reinforcement learning) for informed initialization, and WOA for refined scheduling and resource assignment. The method improves initial feasibility, avoids local optima,

speeds up convergence, reduces makespan, and cuts costs across multiple objectives. The approach follows a two-phase design:

Initialization phase (SARSA-based Q-value initialization) Use the SARSA agent's Q-table to seed initial cloudlet-to-VM assignments, rather than random allocation. Optimization phase (WOA-based task scheduling and resource assignment) better initial population reduces the search space that the optimizer (WOA) must explore, leading to quicker convergence to high-quality solutions, reduced risk of poor local optima.

Incorporating Q-values into the WOA steps biases the search toward regions of the solution space that have historically performed well, while still allowing exploration to avoid premature convergence. **Figure 4** shows propose system flowchart.

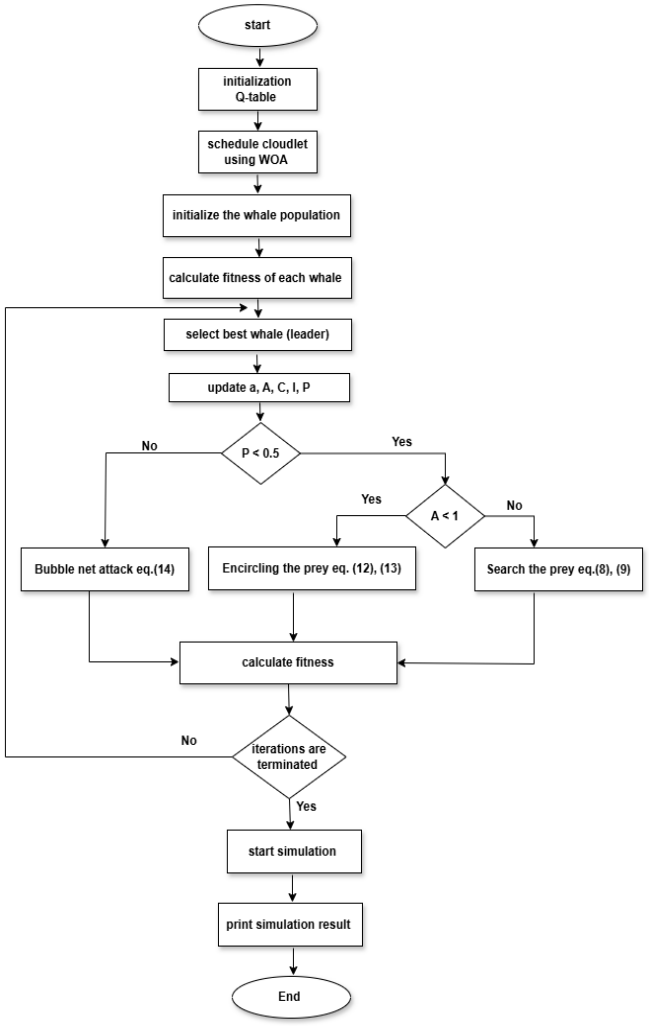


Figure 4. show propose resource allocation system hybrid approach flowchart.

5. Discussions and Experimental Evaluation

This section presents the experimental results and the parameter settings used. The simulation was built and developed using CloudSim 3.0.3. The experiments were run on a PC with an Intel Core i7-8565U CPU running at 1.8 GHz and 8 GB of RAM. **Table 1** shows the parameter configuration. The allocation efficiency of the proposed method was evaluated using NASA datasets.

Table 1. Parameter settings of proposed system.

No.	Item	Parameter	Value
1	Cloudlet	Number	100-1000
2		number	30-150
		VmScheduler	Time Shared
		RAM	2 GB
	Low performance	mips	1000
	Medium performance	Mips	2000
	High performance	mips	4000
3	Datacenter	number	3
4	Hosts	host no.	2 PM

The efficiency and behavior of virtual machines (VMs) during the simulation are influenced by these parameters, which define the processing resources and network characteristics of the simulated environment.

5.1 Experimental results based on the number of cloudlets

The results of the resource allocation algorithms (SARSA and MO-SAWOA) that have been provided have been evaluated by taking into account the average of ten independent executions for each dataset independently. Table 2 shows the makespan (MS), load balance (LB), and cost obtained based on experimental results in comparison to SARSA and MO-SAWOA approaches. Figures 4, 5, and 6 display the corresponding graph for Table 2. As the number of tasks rises, vms stays constant at 30 for both datasets and comparing algorithms.

Table 2. MS, LB and cost evaluation for different cloudlet number.

Data set	**No. of task**	**MS**		**LB**		**COST**	
		sarsa	**MO-SAWOA**	**sarsa**	**MO-SAWOA**	**Sarsa**	**MO-SAWOA**
NASA	200	30.76	5.53	8.32	2.042	46.12	40.221
	400	47.85	9.19	13.94	3.017	81.11	69.144
	600	66.23	12.66	20.77	3.786	124.62	110.94
	800	77.44	16.55	27.16	4.921	168.12	140.10
	1000	98.52	20.18	33.92	5.364	210.13	188.14

As the total amount of tasks rises from 200, 400, 600, 800, and 1000, **Table 2** demonstrates. MS first rises appropriately

for the datasets and compared methods. Second, the results' most notable finding is that the suggested hybrid metaheuristic algorithm MO-SAWOA consistently and overwhelmingly outperforms sarsa, showing the lowest LB value among the comparable algorithms for varying numbers of cloudlets. This is demonstrated by the LB values decreasing as the number of cloudlets increases. Lastly, MO-SAWOA makes sure that all resources are used efficiently so that the system as a whole completes its tasks as quickly and economically as possible. The NASA dataset costs are comparatively low, ranging from 43 to 218 as cloudlets rise, according to the table**. Figure 5** illustrates MS rise according to cloudlet rise and **Figure 6** for Load balance and **Figure 7** showing cost.

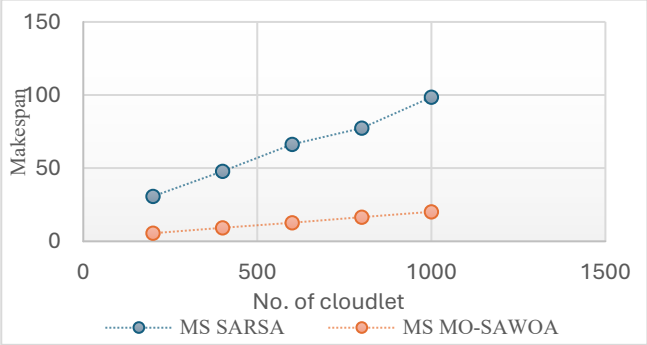
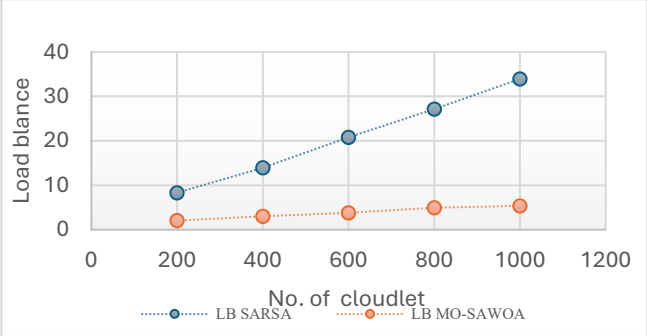


Figure 5. MS of NASA dataset.

From figure above, we evaluated makespan with different workloads, and the results show that the MO_SAWOA consistently yields a smaller MS than SARSA at all cloudlet counts (e.g., 200: 5.532 vs 30.763; 1000: 20.185 vs 98.525).



Figures 6. LB of NASA dataset.

Figure 5, show the blue line (LB SARSA) rises more than the red line (LB MO_SAWOA) as workload grows, MO_SAWOA shows much lower LB values than SARSA at every point (e.g., 200: 2.042 vs 8.327; 1000: 5.364 vs 33.921).

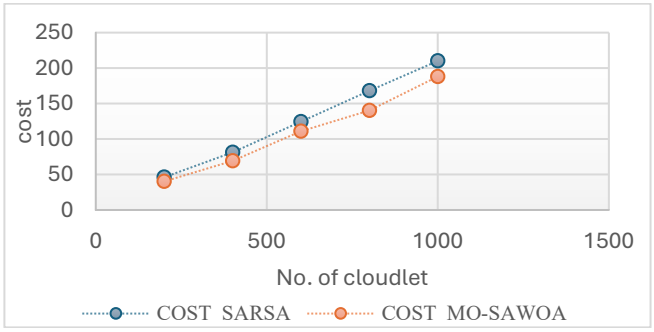


Figure 7. Cost of NASA dataset.

MO_SAWOA is achieving cost savings compared to standard SARSA MO_SAWOA costs are lower than SARSA across all cloudlets (e.g., 200: 40.221 vs 46.127; 1000: 188.148 vs 210.131) as described in figure 7.

5.2 Experimental results based on the number of vms

This simulation aims to investigate how the allocation outcomes vary with the number of virtual machines (VMs). The MS, LB, and cost measure findings for SARSA and MO-SAWOA comparison for different numbers of virtual machines are displayed numerically in **Table 3**. The matching graph for Table 3 is shown in Figures 8,9 and 10.

Table 3. MS, LB and cost evaluation for different vms

Data set	No of vm	MS		LB		Cost	
		sarsa	MO-SAWOA	sarsa	MO-SAWOA	sarsa	MO-SAWOA
NASA	30	59.33	11.61	17.08	3.571	96.65	90.55
	60	47.49	8.45	12.98	2.363	99.81	92.11
	90	46.15	6.51	10.96	1.789	100.1	97.26
	120	45.85	5.84	9.136	1.394	100.9	99.18

Table 3 demonstrates that the proposed MO-SAWOA consistently outperforms the other techniques for the NASA datasets. The table indicates that the MS values decrease dramatically as the number of virtual machines increases. The algorithm's performance improves when more virtual machines are added, resulting in greater load distribution and overall efficiency.

As can be seen from Fig. 8, the execution time of (MO-SAWOA) is less than (SARSA) when the number of virtual machines increases, showing the efficiency and flexibility of the algorithm to handle a greater number of jobs.

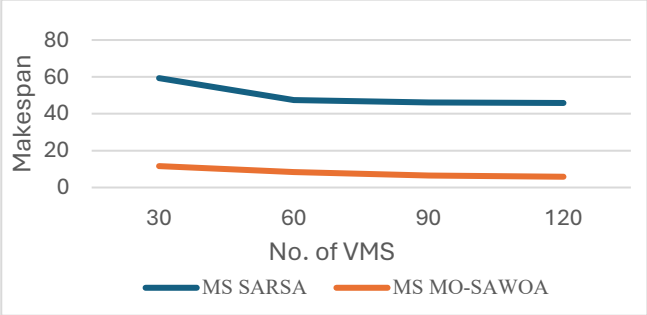


Figure 8. MS of different Vms.

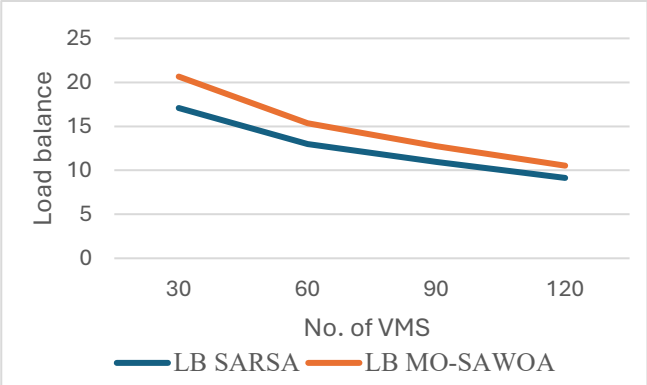


Figure 9. Load balance of different Vms.

In **Figure 9**, it is shown that the load balancing is better as the number of virtual machines increases, but MO_SAWOA is better for task allocation and optimization.

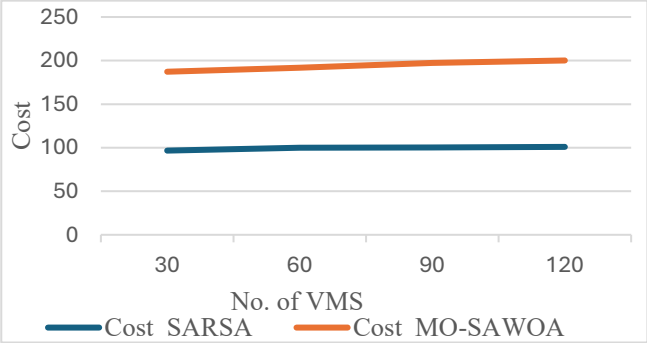


Figure 10. Cost of different Vms.

The cost is seen to increase as the number of virtual machines increases from 30 to 120 as seen in the figure 10. The algorithm processing cost of MO-SAWOA is always less than SARSA algorithm which leads to better performance effectiveness.

6. Existing Methods Comparison

To ensure the effectiveness of the proposed resource allocation system algorithms, a comparison with previously published techniques, including PSO and GWO has been performed [18]. According to MS and LB, performance has been evaluated. 500 cloudlets and various numbers of virtual machines have been considered in this simulation. It should be mentioned that the comparison was made by re-implementing the previous study results in line with the model of cloud computing, datasets, and virtual machines. Table.4 highlights these comparisons.

Table.4: Comparisons of MS evaluation

Data set	No. of vm	PSO	GWO	MO_ QAWOA	MO_ SAWOA
NASA dataset	25	19.915	20.191	13.255	5.063
	50	10.775	11.717	9.670	4.468
	75	7.622	6.804	6.412	3.621
	100	6.093	7.635	5.643	1.875
	Av.	11.101	11.586	8.745	3.756

Generated makespan for PSO for 25, 50 ,75, 100 vm is 19.915, 10.775, 7.622, 6.093 respectively. Generated makespan for GWO with 25, 50 ,75, 100 vms is 20.191, 11.717, 6.804, 7.635 respectively. Makespan generated for MO_QAWOA with same number of vms is 13.255, 9.670, 6.412, 5.643 respectively. Makespan generated for MO_SAWOA is 5.063, 4.468, 3.621, 1.875 respectively.

There is another comparison that was made with [19] to evaluate two objective makespan and cost for different number of tasks (100, 500 and 1000). As shown in **Table 5**.

Table 5: Evaluation of Makespan of NASA dataset

Dataset	No. of cloudlet	DQN	A2C	MO-QAWOA	MO-SAWOA
NASA dataset	100	942.14	765.64	4.855	4.213
	500	1089.26	1107.26	11.251	10.644
	1000	1437.58	1756.93	20.945	20.185
	Av.	3468.86	3629.83	37.051	35.042

There is another comparison that was made with [20-22] to evaluate cost for (500) cloudlets. As shown in **Table 6.**

Table 6. Comparisons of cost of NASA dataset

Dataset	No. of vm	DQN	MO_QAWOA	MO_SAWOA
NASA dataset	100	5039	38.86	32.22
	500	7205	81.22	40.75

Table 6 shows the cost numerical results for MO_SAWOA and the comparison algorithms. It allows obvious that (MO_SAWOA) has obtained the greatest cost savings when compared to (DQN) and (MO_QAWOA).

Conclusion

The proposed system is flexible and dynamic in handling multi-objective resource allocation in cloud environment. It employs a combination of SARSA and the whale optimization algorithm (WOA). SARSA combined with WOA allows for proper balance between exploration/exploitation and global search. This hybrid approach reduces the risk of getting trapped in local optima and accelerates convergence to high-quality allocation policies. MO-SAWOA results in a better balance between the aspects of performance, efficiency and fairness.

Using CloudSim environment and NASA data, experiments were performed to observe the performance of the proposed algorithm for varying number of cloudlets (100 to 1000) with varying number of VMs (30 to 120). The proposed method balances task loads according to available system resources, and better load balancing, reduces makespan, reduces cost, increases throughput, and provides a higher average expected return than SARSA algorithm.

As future work, we aim to extend this task allocation approach to other scenarios such as fog computing and test the performance of MO-SAWOA in real world testbeds like AWS or AZUR.

Conflict of interest

None.

References

[1] Abdulrazzaq, D. R., Shati, N. M., & Hoomod, H. K. (2023, May). Bi-objective task scheduling based on heuristic initialization of the jellyfish search algorithm in cloud computing. In 2023 3rd International Scientific Conference of Engineering Sciences (ISCES) (pp. 25-30). IEEE.

[2] Abdulrazzaq, D. R., Shati, N. M., & Hoomod, H. K. (2024). Task scheduling in a cloud environment based on meta-heuristic approaches: A survey. Iraqi Journal of Science, 65(2), 1001-1023.

[3] Khan, F. A., Gumaei, A., Derhab, A., & Hussain, A. (2019). A novel two-stage deep learning model for efficient network intrusion detection. IEEE Access, 7, 30373–30385.

[4] Khraisat, A., Gondal, I., Vamplew, P., & Kamruzzaman, J. (2019). Survey of intrusion detection systems: Techniques, datasets and challenges. Cybersecurity, 2(1), 1–22.

[5] Saleh, H. H. (2016). Increasing security for cloud computing by steganography in image edges. Al-Mustansiriyah Journal of Science, 27(4), 63-70.

[6] Khudhair, H. A. (2025). Intelligent formation protocol: An approach for enhancing energy efficiency and network performance in wireless sensor networks. Al-Mustansiriyah Journal of Science, 36(1), 46-55.

[7] Gong, Y., Huang, J., Liu, B., Xu, J., Wu, B., & Zhang, Y. (2024). Dynamic resource allocation for virtual machine migration optimization using machine learning. arXiv preprint arXiv:2403.13619.

[8] Arvindhan, M., & Kumar, D. R. (2023). Adaptive resource allocation in cloud data centers using actor-critical deep reinforcement learning for optimized load balancing. International Journal on Recent and Innovation Trends in Computing and Communication, 11(5s), 310-318.

[9] Panwar, R., & Supriya, M. (2024). Rlpraf: Reinforcement learning-based proactive resource allocation framework for resource provisioning in cloud environment. IEEE Access, 12, 95986-96007.

[10] Bansal, S., & Aggarwal, H. (2024). An efficient workflow scheduling in cloud–fog computing environment using a hybrid particle whale optimization algorithm. Wireless Personal Communications, 137(1), 441-475.

[11] Kumari, S., & Mishra, D. (2025). Adaptive, efficient and fair resource allocation in cloud datacenters leveraging weighted A3C deep reinforcement learning. arXiv preprint arXiv:2506.00929.

[12] Kusuma, G. S., & Devi, M. (2025). Optimized resource management and security enhancement in fog computing using advanced Q-learning approaches. Engineering, Technology & Applied Science Research, 15(3), 23965-23971.

[13] Chen, Y., Ganapathi, A. S., Griffith, R., & Katz, R. H. (2010). Analysis and lessons from a publicly available Google cluster trace (Tech. Rep. UCB/EECS-2010-95). EECS Department, University of California, Berkeley.

[14] Kruekaew, B., & Kimpan, W. (2022). Multi-objective task scheduling optimization for load balancing in cloud computing environment using hybrid artificial bee colony algorithm with reinforcement learning. IEEE Access, 10, 17803-17818.

[15] Dhari, A., & Arif, K. I. (2017). An efficient load balancing scheme for cloud computing. Indian Journal of Science and Technology, 10(11), 1-8.

[16] Malti, A. N., Benmammar, B., & Hakem, M. (2022). QoS based task scheduling algorithm in cloud computing. E3S Web of Conferences, 351, Article 01014. EDP Sciences.

[17] Zhou, J., Lilhore, U. K., Hai, T., Simaiya, S., Jawawi, D. N. A., Alsekait, D. M., ... & Hamdi, M. (2023). Comparative analysis of metaheuristic load balancing algorithms for efficient load balancing in cloud computing. Journal of Cloud Computing, 12(1), 1-21.

[18] Bibal Benifa, J. V., & Dejey, D. (2019). Rlpas: Reinforcement learning-based proactive auto-scaler for resource provisioning in cloud environment. Mobile Networks and Applications, 24(4), 1348-1363.

[19] Natesan, G., & Chokkalingam, A. (2020). Multi-objective task scheduling using hybrid whale genetic optimization algorithm in heterogeneous computing environment. Wireless Personal Communications, 110(4), 1887-1913.

[20] Jain, P., & Sharma, S. K. (2023). A load balancing aware task scheduling using hybrid firefly salp swarm algorithm in cloud computing. International Journal of Computer Networks and Applications, 10(6), 914-925.

[21] Mangalam Palli, S. S., Karri, G. R., Mohanty, S. N., Ali, S., Khan, M. I., Abdullaev, S., & AlQahtani, S. A. (2024). Multi-objective prioritized task scheduler using improved asynchronous advantage actor critic (A3C) algorithm in multi cloud environment. IEEE Access, 12, 11354-11377.

[22] Pan, J., Wei, Y., Meng, L., & Meng, X. (2025). A dual scheduling framework for task and resource allocation in clouds using deep reinforcement learning. Journal of King Saud University - Computer and Information Sciences, 37(5), Article 81.