

EFFECTIVE LOAD BALANCING IN CLOUD COMPUTING USING A HYBRID OPTIMIZATION ALGORITHM

Anushree Kaushik, Gulista Khan, Priyank Singhal*

Teerthanker Mahaveer University, Moradabad, U.P.
India

* Corresponding Author, e-mail: anushree.scholar@tmu.ac.in

Abstract: Effective load balancing is a critical challenge in cloud computing environments, where dynamic and unpredictable workloads necessitate efficient resource allocation to ensure optimal performance and energy efficiency. This paper proposes a hybrid optimization algorithm, combining Quantum-Gray Wolf Optimization (Q-GWO) and Particle Swarm Optimization (PSO) to enhance load balancing in cloud computing. The Q-learning technique enhances the decision-making process by selecting appropriate virtual machines (VMs) based on their predicted fitness values. The hybrid GWO-PSO algorithm is then used to update positions and velocities, ensuring optimal resource utilization. Empirical evaluations demonstrate the superiority of the Q-GWO-PSO algorithm over traditional methods like Q-Learning Modified PSO (QMPSO) and Modified PSO (MPSO). Specifically, Q-GWO-PSO achieved a makespan of 7074.11 milliseconds (ms), throughput of 2129.16 requests per second (req/ms), a Standard Deviation (SD) of 0.05479, energy utilization of 150.979-kilo joule (KJ), and migrated 306 tasks, showcasing its robust and scalable solution for load balancing.

Key words: Load balancing, cloud computing, hybrid optimization, VMs.

1. INTRODUCTION

A more recent development in Internet-based computing is cloud storage, which makes use of shared computers with numerous people's Personal Computers (PCs) connected to a single network to carry out certain tasks as needed [1]. In the context of cloud computing and heavily internet-based applications was initially designed to satisfy consumer demands for computing resources access or to allow them to get cloud services only if needed [2]. Moreover, cloud computing could handle a wide range of services, depending on the platforms and services that customers require. Cloud computing combines parallel and distributed computing while sharing resources, including hardware and software.

Users only need a connection to the internet to use cloud computing and cover the cost of their usage; users don't have to purchase any platforms or software to use it. One of the technology models used to manage data centers, servers, and VMs is called

Infrastructure as a Service (IaaS). It is a service provided by the cloud that provides a computer or cloud server for processing and storing data. Customers could use a rental server with any kind of operating system or application, and there are no hardware maintenance or operational charges. Another benefit of IaaS is that it gives the user access to servers located nearby because the cloud infrastructures are operated on VMs. Quality of Service (QoS) and Service Level Agreements (SLAs) are two types of services that are user-centric. The method of payment for the services is dictated by the user's agreements with the cloud services providers (CSPs)

By further supporting consumers using cloud services, cloud computing allows CSPs to supply powerful computing resources to data centres. A VM uses up the random-access memory (RAM) of the host computer, hard drive, or control processing unit (CPU). The requested resource needs the resources on the VM to function. The resource allocation on the cloud network is, therefore, uneven and numerous. VMs have both pre-emptive and non-pre-emptive connections to resources, and certain VMs could be unable to access the resources VMs require. The cloud environment and all cloud entities collaborate. For instance, cloud auditors serve as the cloud's police, making sure that CSPs' services are honest and of the highest calibres.

Traditional load-balancing techniques have numerous difficulties due to the wide range of application requirements, fluctuating workloads, and different resource capacities.

To address these challenges, this research proposes the use of an efficient hybrid optimization technique for cloud computing load balancing.

- The research contributes to computing in the cloud by proposing a novel hybrid cloud scheduling algorithm.
- The research was performed by a hybrid cloud scheduling algorithm integrating GWO and PSO to address the NP-hard problem of resource allocation in cloud computing.
- The model was analysed to efficiently schedule tasks of varying sizes within cloud data centres, prioritizing them based on duration and runtime.
- The proposed solution involved optimizing resource utilization while considering factors like time, memory, and CPU, ultimately improving task completion efficiency in cloud infrastructures.

This research project's scope includes both theoretical creation and the use of the hybrid optimizing method in practice. The structure of this work is described as follows: In Section 1, the introduction provides a concise overview of the topic and explains why it is important. Previous studies are reviewed and analysed in Section 2. Section 3 examines the framework's potential consequences. To help with better understanding, the next part, section 4, summarizes the results and provides a brief explanation. Section 5 contains not just some views on possible future work but also the study's conclusion.

2. LITERATURE REVIEW

This review examines load balancing in cloud computing, focusing on the necessity for improved methods in dynamic environments. It discusses the application of optimization methods like Particle Swarm Optimization and Genetic Algorithm,

highlighting their benefits in addressing dynamic load balancing challenges. The analysis concludes with the suggestion of an improved hybrid optimization technique designed specifically for cloud settings, which promises better resource and performance utilization.

Recent studies contribute significantly to this field. [3] proposes a Long Short-Term Memory (LSTM) network that incorporates one-dimensional Convolutional Neural Networks (CNNs) and utilizes LSTM modules to capture temporal information. In contrast, a CNN module used data on the VM workload to extract complex differentiating features. The CNN-LSTM model was trained using a PSO-GA approach, which improved the model's performance where the Memory: 0.036, Processing power: 0.37, Storage: 0.25, and Memory: 0.03 were the multivariate analysis's material adverse effect (MAE) results. [4] presented the dragonfly (DF) and particle swarm optimization (PSO) algorithms provided the most efficient resource for load balancing after comparing the PSO and the Dragonfly method and was found that the DPSO strategy outperformed the others and the response time with DPSO was a mere 0.4 seconds, which was shorter than the other model, according to the data from 100 users.

Further, [5] found a new hybrid strategy that used two steps to break down the problem into its parts through a clustering technique whereby data centers organized servers that have comparable utilization rates into smaller groups. The distribution between clusters is controlled at the data center level by load balancing and task scheduling. With this strategy, got the best average makespan of 1.18, the lowest migration ratio of 12%, and the most reduced amount of service level agreement (SLA) breaches of 8%. In their study, GERALD and DRP (2024) [6] proposed the improved particle swarm optimization-firefly (FIMPSO), a metaheuristic load balancing in cloud computing extensively studied, which reduced response time, optimized resource used, and improved fault tolerance. The improved particle swarm optimization-firefly (FIMPSO) approach had the lowest make-span durations across all task sizes, with a minimum of 29 units for small tasks. FIMPSO had the highest average throughput across all workloads and suggested the best task completion.

A method for increasing clustering efficiency by integrating reinforcement learning with the robust hybrid lyrebird falcon optimization (HLFO) algorithm is suggested in [7]. HLFO combined the lyrebird and falcon optimization algorithms to improve load balancing. The HLFO model showed the best performance energy used CPU balance better memory usage task prioritizations taken 4.387614347 units per 10 VMs and 5.598419936 for 50 VMs, while balanced CPU utilization delivered 0.006 and 0.0105. Following this, [8] suggested a novel weight-assigned load-balancing round-robin algorithm. This algorithm incorporated five server metrics that have been carefully selected to effectively balance the virtual machines' three-tier web apps' burden. These experimental findings indicated that this strategy improved average reaction times by 12.5% as compared to the base algorithm and by 22.3% when compared to the round-robin algorithm. When compared to the baseline method, the average time to respond was improved by 20.7%, while when compared to the round-robin algorithm, it was improved by 21.4%.

In another study, [9] examined a bio-inspiration improved lion optimization meta-heuristic approach that has been developed to solve the load balancing concern that was

present in the world of cloud computing. When comparing the meta-heuristic benchmark EMAMBO_LB, BOA_LB, and CSO_LB, it was discovered that the throughput was improved by 7%, 9%, and 13%, respectively. The ILOA_LB demonstrated an improved tolerance to failure even as the workload in the nodes grew. [10] introduced a hybridization of improved MPSO with an improved Q-learning algorithm where the QMPSO enhanced make span, throughput, energy utilization, and task waiting time during load balancing. The hybrid QMPSO model has a high throughput (5.46) and lower energy use in VMs and tasks (173.80 and 222.45). In a previous study, [11] proposed a hybridized heuristic technique in metaheuristic algorithms that used two hybrid techniques for the HDD-PLB framework using the ant colony optimization (ACO) and hybrid heterogeneous earliest finish time (HEFT). The average makespan of Cyber shake workflow input utilized Hybrid PEFT-ACO was 7.6 times better than Hybrid HEFT-ACO. The study by [12] examined the HLBGA algorithm in the heterogeneous-high algorithm, which yielded the best results in comparison to the other algorithms. As an approach that combined hierarchical interpolation, local basis (HILB) and genetic algorithm (GA), the HLBGA algorithm that has been presented performed better than the previous algorithms. Compared to HILB and GA, HLBGA had a makespan improvement of 8% and 48.3%, load deviation improvement of 34.3% and 85%, and utilization improvement of 3.4% and 40%, respectively.

3. RESEARCH METHODOLOGY

Each Research methodology is considered when examining the idea of planned architecture shown in Figure 1. Scheduling is essential in cloud computing so that users can complete the work on time. Therefore, choosing the right optimal resource is essential, and it's a non-deterministic polynomial (NP)-hard problem. An algorithm for a hybrid cloud scheduling-based algorithm is suggested by integrating GWO and PSO. Q-learning is a reinforcement approach used for further scheduling and prediction. Scheduling tasks receive a variety of jobs in cloud infrastructure, varying in size. The goal of the suggested model's metaheuristic algorithms is to finish tasks inside the cloud data centre. Assigning tasks to the virtual machines is the main objective of the suggested method. The suggested solution uses a metaheuristic approach to regulate the jobs' priority based on their duration and runtime. After the tasks have been assigned priorities, these tasks are arranged in the task queue. The hybrid algorithm carries out the job scheduling procedure and is dependent on data centre attributes like time, memory, and CPU.

The work's goal function was created by quality-of-service measurements such as availability, throughput, and response time.

The duration of time required by a service to reply to a certain kind of inquiry is known as its response time. Throughput is the fastest possible speed at which a request could be processed. The percentage of a service or system that is up and running and available to users is known as its availability.

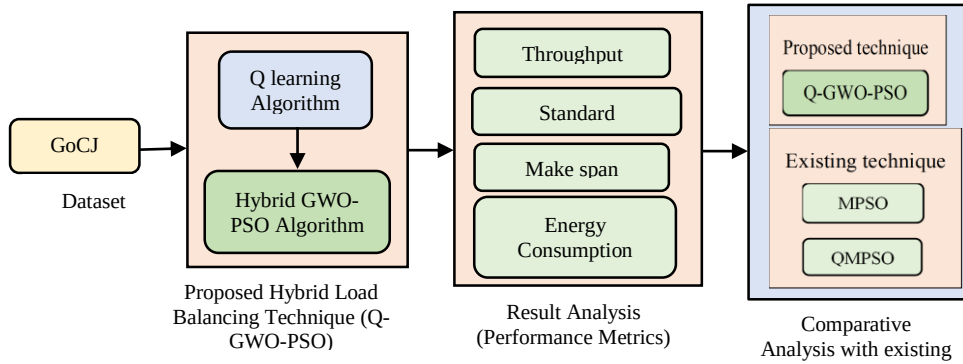


Figure 1. Proposed Methodology

The following summarizes the work's main goal:

$$\text{Objective Function} = \max(\text{AvgRT}, \text{Throughput}, \text{Availability}) \quad (1)$$

The highest possible value of the system's average reaction time, throughput, as well as availability sets the performance threshold that the system strives to meet. The system's performance metrics can be balanced and kept above a certain acceptable threshold if this is done [13].

4. PROPOSED HYBRID LOAD BALANCING TECHNIQUE

4.1. Q learning algorithm

One algorithm used in Reinforcement Learning (RL) is the Q-learning algorithm. Agents could learn from and adapt to their environment through the machine learning technique known as reactive learning (RL) by altering their state in response to environmental feedback, which could result in rewards or penalties. RL's primary goal is to teach the agent by having it make mistakes and learn from its surroundings. By a state, the agent could receive information about the environmental conditions and make decisions that impact the environment in order to maximise rewards and learn from past errors. In an unpredictable environment, agents adopt Markov's decision process (MDP), which might occasionally provide chaotic results.

Every state contains a set of actions, given that set of state $S = \{s_1, s_2, s_3, \dots, s_n\}$ is in the environment. Through the transition process, the $A = \{a_1, a_2, a_3, \dots, a_m\}$ agent chooses an action $A_t \in a$ at the time t in state $s_t \in S$ to move to the next state $s_{t+1} \in S$ and obtains a reward r_{t+1} from the surroundings. The state's choice of action determines the Q-value function. The Q-value functional is supposed to go to the optimal state as well as obtain the greatest possible total estimated rewards in the environment provided the agent in the state s_t .

$$Q(s_t, a_t) = (1 - \alpha) Q(s_t, a_t) + \alpha [r_t + \gamma \max_{a'} Q_t(s_{t+1}, a_{t+1})] \quad (2)$$

Where r_t is the rewards or penalties for acting in the state s_t , α represents the learning rates, and γ ($0 < \gamma < 1$) represents the discount rate along with the effect of the prior action on the subsequent state. Making a Q-table with all potential states, Q-values, and suitable actions is the first step towards determining the Q-value. The Q-

learning algorithm incorporates the greedy technique to find the optimal state using user experience as input and then uses that value to determine the Q-value.

$$Q_{t+1}(s_t, a_t) = Q(s_t, a_t) + \alpha [r_t + \gamma \max_{a't} [Q_t(\delta(s_t, a_t, a't))] - Q_t(s_t, a_t)] \quad (3)$$

Where δ is the transitional function, and α is the learning rate, which is determined by using the formula $\alpha = 1/(1 + \text{the total number of visit to state } s_t)$. The best possible future value is approximated by the maximum of a and the product of $\max_a Q(s_{t+1}, a)$. An activity that could be performed in the cloud to allocate resources among virtual machines is load balancing [13].

4.2. Hybrid GWO-PSO Algorithm

The Particle Swarm Optimization algorithm (PSO) and the Grey Wolf Optimizer (GWO) hybridized involves integrating the exploration and exploitation capabilities of both metaheuristic algorithms to enhance search efficiency and convergence speed. The hybrid method aims to improve the quality of solutions along with the convergence performance of optimization problems by striking a balance between the exploration of promising areas and the discovery of the solution area. The best aspects of both variants have been combined.

The following equation is used to update the particle's position and velocity:

$$v_j = \omega * [v_j^t + c_1 r_1 (X_1 - x_j) + c_2 r_2 (X_2 - x_j^t) + c_3 r_3 (X_3 - x_j^t)] \quad (4)$$

$$x_j^{t+1} = x_j^t + v_j^{t+1} \quad (5)$$

The velocity as well as position are randomly initialized in the first stage of the suggested algorithm. These initializations randomly initialize a population of N particles with dimensions equal to the number of tasks for N particles. Once the parameters have been initialized, the loop is executed for a predefined maximum number of iterations (maxIter) [14].

4.3. Q-GWO-PSO Algorithm

The suggested framework for scheduling jobs in the cloud is shown in Figure 2. When the Users request tasks, the system receives these requests and uses a hybrid GWO-PSO metaheuristic approach to optimise load balancing. It describes the data centre using a hybrid GWO-PSO metaheuristic approach and Q-learning to optimize load balancing, scheduling, and fitness value prediction. So, Q-learning is then used to select appropriate virtual machines (VMs) and predict their fitness value for the tasks. Finally, a task scheduler allocates the tasks to the VMs balancing. So, Q-learning is then used to select appropriate virtual machines (VMs) and predict their fitness value for the tasks. Finally, a task scheduler allocates the tasks to the VMs.

Proposed Algorithm

- Step 1. Initialize the cloud environment and virtual machines
- Step 2. Initialize Q-table with random values
- Step 3. Initialize GWO population
- Step 4. Initialize PSO population
- Step 5. Set maximum iterations
- Step 6. Set convergence criteria

- Step 7 for iteration = 1 to maximum iterations:
 Step 8 Update GWO positions using the GWO algorithm
 Step 9 Update PSO positions using the PSO algorithm
 Step 10. Evaluate the fitness of GWO and PSO solutions
 Step 11. Select the best solution from GWO and PSO populations
 Step 12. Use Q-learning to update Q-values based on the selected solution
 Step 13. Update Q-table
 Step 14. Select an action based on the updated Q-table (e.g., allocate VM to a server)
 Step 15. Monitor system performance (e.g., Response time, Throughput, Availability)
 Step 16. if convergence criteria are met:
 Step 17. break
 Step 18. Update exploration-exploitation tradeoff for Q-learning
 Step 19: Update GWO and PSO parameters (e.g., alpha, beta, inertia weight, velocity)
 Step 20: End For
-

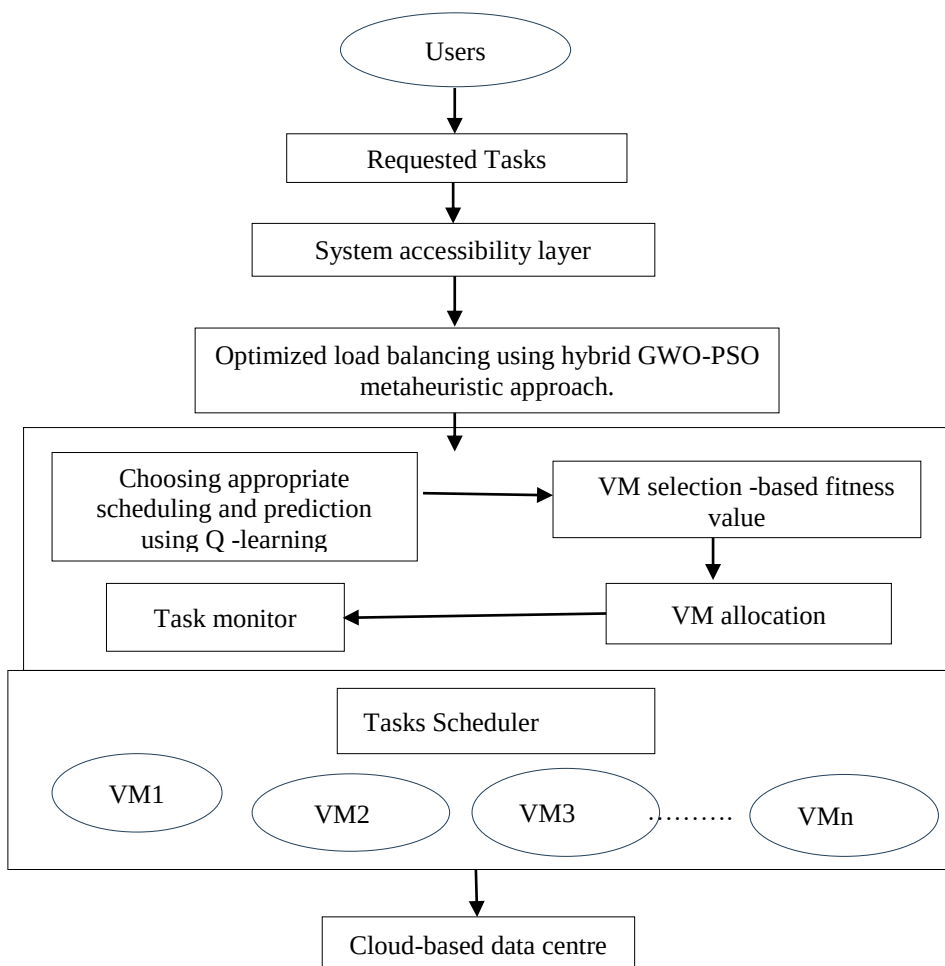


Figure 2. Proposed Architecture

5. RESULT ANALYSIS

This section details the method for experimentally evaluating the suggested load-balancing strategy in a cloud computing environment. The details of the proposed cloud environment are detailed in the simulated environment, and the performance matrices cover each key matrix. Table 1 displays the parameters that are used throughout the experimental setup.

Table 1. Experimental Setup

S. No.	Parameters	Values
1.	Cloud environment type	Simulated
2.	Sim tool kit	Cloud sim 4.0
3.	Software version	3.11.1
4.	PMs	1
5.	VMs	10 to 300
6.	Number of Tasks	500 to 2000

5.1. Performance Metrics

Several measures were used to evaluate the algorithm's performance: Makespan, energy utilisation, standard deviation for load balancing, and throughput for evaluation of the efficacy of external services.

• Throughput

Throughput is the maximum speed at which a service can process requests. Throughput can be calculated using the formula:

$$T = \frac{N}{t} \quad (5)$$

Where N is the number of requests processed, and t is the time taken to process these requests.

• Standard Deviation

In the context of effective load balancing, standard deviation is a statistical measure that quantifies the variation or dispersion of load (such as tasks, processes, or data) across different servers or nodes in a computing environment

$$\sigma = \sqrt{\sigma^2} = \sqrt{\frac{1}{n} \sum_{i=1}^n (Li - \mu)^2} \quad (6)$$

• Makespan

Makespan refers to the total amount of time required to complete all tasks or activities in a system. The makespan can be calculated using the following formula:

$$\text{Makespan} = \max_{i \in \{1, 2, \dots, n\}} C_i \quad (7)$$

where: C_i is the completion time of task i ; n is the total number of tasks.

• Energy Consumption

In a cloud architecture, managing energy consumption primarily aims to reduce total power use. Energy consumption in cloud infrastructure can be quantified using the following.

$$E = P \times T \quad (8)$$

where E is the total energy consumed, P is the power consumption rate, and T is the time duration over which the power is consumed.

5.2. Results

Performance metrics for various Virtual Machines (VMs) identified by their VM IDs. Each VM is evaluated based on three key parameters (k_1 , k_2 , and k_3), with all k_1 values set to 1.00, indicating uniformity in this parameter. The Makespan, which represents the total execution time, ranges from 7052.08 ms to 7880.21 ms, highlighting the time required to complete tasks. Throughput, measured in requests per millisecond (req/ms), varies between 5.17 and 9.92, reflecting the processing efficiency of each VM. The Standard Deviation (SD), indicating the variability in load distribution, shows diverse values, with some VMs having low variability (0.14) and others having higher variability (0.89). Energy utilization, expressed in kilojoules (KJ), spans from 15.01 KJ to 148.43 KJ, providing insight into energy consumption. The Load Balance column marks all VMs as "true", indicating their compliance with the load balancing criteria.

Figure 3(a) shows how much energy different VMs use when they share the load, and Figure 3(b) displays the same information for three different algorithms: QMPSO, MPSO, and the brand-new Q-GWO-PSO method for tasks. Figures 3(a) and 3(b) show that, when comparing the three traditional algorithms, QMPSO, MPSO, and the suggested Q-GWO-PSO, the suggested method uses the least amount of energy while balancing the load.

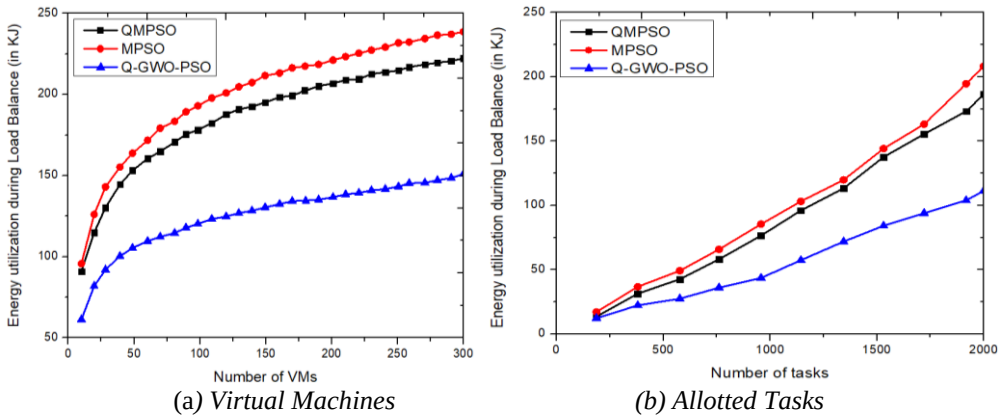


Figure 3. Energy Consumption during Load Balancing in KJ

Makespan evaluates the user's reaction time for a specific task. Since response time influences the Quality of Service (QoS) in cloud computing, the service provider is able to guarantee the quality of service to the customer. Figure 4(a) displays several virtual machines (VMs), while Figure 4(b) displays various jobs that extend the Makespan. Figures 4(a) and 4(b) demonstrate a significant decrease in energy consumption and makespan when comparing the suggested Q-GWO-PSO to MPSO and QMPSO. Thus, by comparing its energy consumption and makespan to its counterpart algorithm, the suggested Q-GWO-PSO effectively accomplished load balancing and stability.

The standard deviation measures the load balance in the network. Smaller standard deviations suggest better load balance. Figure 5(a) shows load balancing performance

by standard deviation. The Q-GWO-PSO algorithm's standard deviation is initially lower than QMPSO and MPSO, but it reduces with time. At 4300 ms, both QMPSO and the proposed Q-GWO-PSO have similar standard deviation values. The standard deviation of QMPSO is higher than the planned Q-GWO-PSO when $t \geq 4300$ decreases. The suggested Q-GWO-PSO quickly obtains optimal resources, allowing for residual processing power.

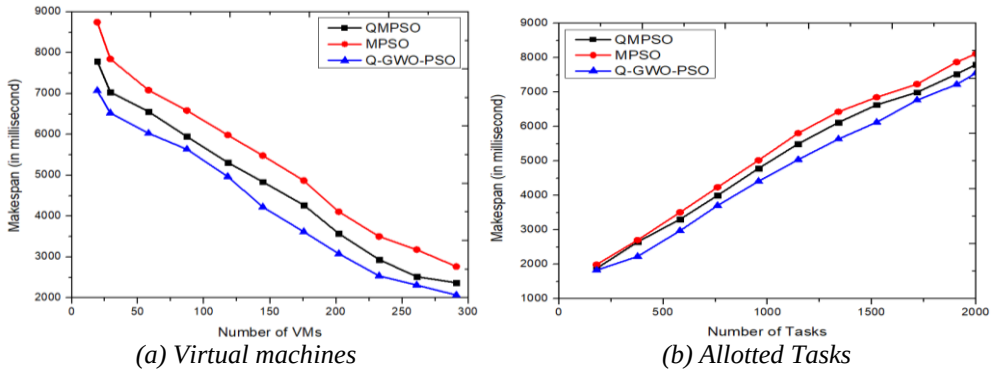


Figure 4. Makespan

The investigation shows that the suggested Q-GWO-PSO method outperforms the competition in terms of resource utilisation capabilities and efficient load balancing. The evaluation was based on the number of tasks that were successfully moved. The task is being moved because the physical host could not provide the requested resource. Figure 5(b) shows the results of an evaluation of the task migration with respect to the number of tasks. In comparison to QMPSO and MPSO, the suggested Q-GWO-PSO migrates fewer tasks.

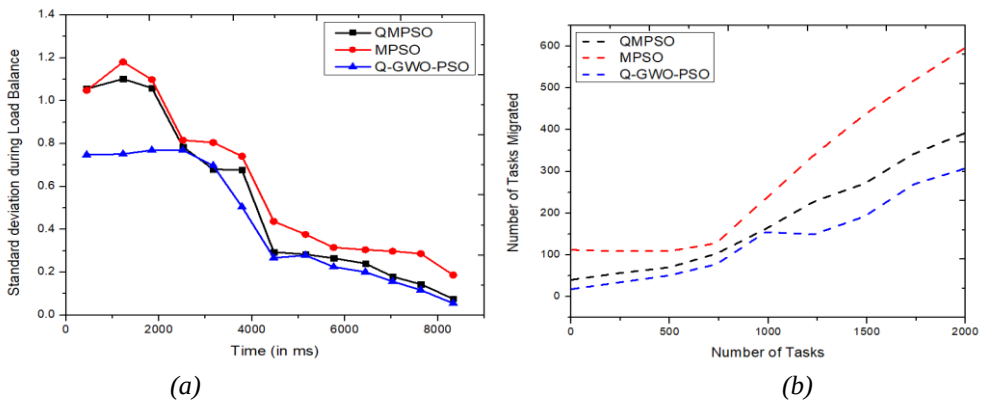


Figure 5. (a) Standard Deviation (b) Tasks Migrated.

Figure 6(a) displays the experimental results for throughput analysis. Q-GWO-PSO testing. Figure 5 demonstrates that Q-GWO-PSO's ext Kernal service performs better than its competitor in the beginning stages. As time goes on, the performance of the external service becomes stable; at $t = 1500$, it's nearly identical for all three algorithms.

Eventually, Q-GWO-PSO's external service outperforms MPSO and QMPSO in terms of stability and efficiency.

The number of tasks required to accumulate resources on the physical host is the primary determinant of load balancing. Since more tasks are being allocated to single high processing-power VMs, the degree of imbalance decreases as the amount of idle time for the tasks increases, as shown in figure 6(b). Having only one task in the ready queue can increase task waiting time. Less waiting time for all activities is possible with higher processing power and more virtual machines (VMs).

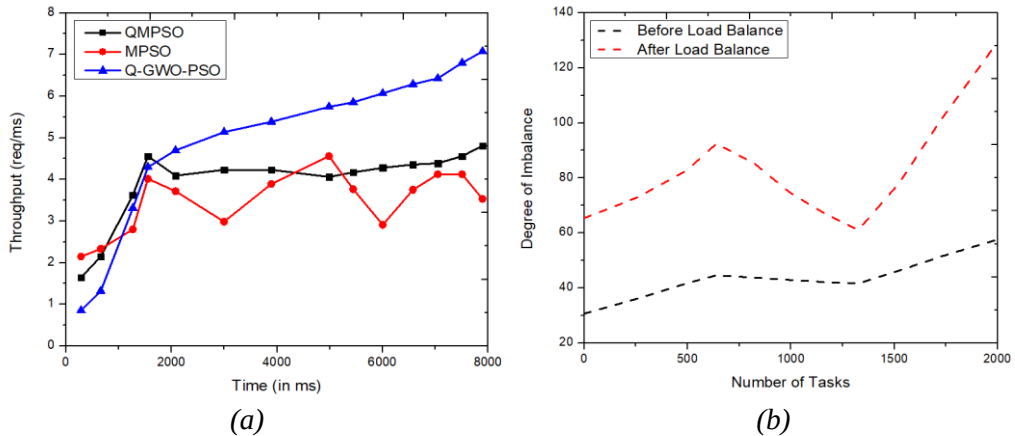


Figure 6. (a) Throughput and (b) Degree of Imbalance

6. CONCLUSION

Cloud computing relies on efficient load balancing and job scheduling to ensure optimal resource utilization. This study proposes a hybrid optimization algorithm, Q-GWO-PSO, integrating Q-learning-based Grey Wolf Optimizer (Q-GWO) with Particle Swarm Optimization (PSO) to enhance VM selection and task scheduling. Q-learning predicts VM fitness for informed decision-making, while the hybrid GWO-PSO optimizes position and velocity updates. Experimental results demonstrate that Q-GWO-PSO achieves a makespan of 7074.11 ms, throughput of 2129.16 req/ms, standard deviation of 0.05479, energy utilization of 150.979 kJ, and 306 task migrations, outperforming QMPSO and MPSO. The results confirm the algorithm's effectiveness in dynamic cloud environments, offering improved performance, reduced energy consumption, and enhanced resource management.

REFERENCES

- [1] Abdalla, et.al. Advantages to disadvantages of cloud computing for small-sized business. *7th International Symposium on Digital Forensics and Security (ISDFS)*, pp. 1-6., 2019 IEEE.
- [2] Haji, et.al. Impact of cloud computing and internet of things on the future internet. *Technology Reports of Kansai University*, vol.62, no. 5, 2020, pp.2179-2190.
- [3] Surianarayanan, et.al. Essentials of cloud computing. *Cham: Springer International Publishing*, 2019.
- [4] Parast, F.K., Sindhav, Ch., Nikam, S. et al. Cloud computing security: A survey of service-based models. *Computers & Security*, vol.114, March 2022, art. 102580.

- [5] Yang et.al, Subscription or pay-as-you-go: Optimally purchasing IaaS instances in public clouds. *2018 IEEE International Conference on Web Services (ICWS)*, pp. 219-226, 2018 IEEE.
- [6] Surbiryala, Jayachander, Chunming Rong. Cloud computing: History and overview. *2019 IEEE Cloud Summit*, pp. 1-7. IEEE, 2019.
- [7] Simaiya, et.al. A hybrid cloud load balancing and host utilization prediction method using deep learning and optimization techniques. *Scientific Reports* 14, no. 1, 2024. 1337.
- [8] Mohapatra, et.al. DPSO: A hybrid approach for load balancing using dragonfly and PSO algorithm in cloud computing environment. *EAI Endorsed Transactions on Internet of Things*, 10, 2024.
- [9] Elsakaan, et.al. A novel multi-level hybrid load balancing and tasks scheduling algorithm for the cloud computing environment. *The Journal of Supercomputing*, 2024, pp. 1-41.
- [10] Gerald, et.al. Metaheuristic algorithm-based load balancing in cloud computing. *Journal of Theoretical and Applied Information Technology*, vol.102, no.5, 2024.
- [11] Khan, Ahmad Raza. Dynamic load balancing in cloud computing: Optimized RL-based clustering with multi-objective optimized task scheduling. *Processes*, vol.12, no. 3, 2024, p.519.
- [12] Adewojo, Adekunbi A., Julian M. Bass. A novel weight-assignment load balancing algorithm for cloud applications. *SN Computer Science*, vol.4, no. 3, 2023, p.270.
- [13] 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*, 10, 2022, pp.17803–17818.
- [14] Shehadeh, H. A., Shagari, N. M. A hybrid grey wolf optimizer and sperm swarm optimization for global optimization. In *Handbook of intelligent computing and optimization for sustainable development* 2022, pp. 487–507. Wiley.

Information about the authors:

Anushree Kaushilk is Ph.D. Research scholar at Teerthanker Mahaveer University.

Prof. (Dr.) Gulista Khan is Professor in the Department of Computer Science and Engineering at Teerthanker Mahaveer University. Having more than 18 Years of teaching and research experience. Her interests include cloud computing, load balancing, optimization algorithms, and energy-efficient resource management.

Dr. Priyank Singhal is Associate Professor in the Department of Computer Science and Engineering at Teerthanker Mahaveer University. Having more than 15 Years of teaching and research experience. His interests include cloud computing, load balancing, optimization algorithms, and energy-efficient resource management.

Manuscript received on 17 April 2026