EAI Endorsed Transactions
on Internet of Things
Research Article EAI.EU

Hybrid Genetic Algorithm and Water Wave Optimization Approach for QoS Aware Multi Objective Task Scheduling and Load Balancing in Cloud Environments

Nihar Ranjan Sabat[1,*], Rashmi Ranjan Sahoo[2], Biswaranjan Acharya[3] and Raghvendra Kumar[4]

[1]Faculty of Engineering, Biju Patnaik University of Technology (BPUT), Rourkela, Odisha-769015, India
[2]Department of Computer Science and Engineering, PMEC, Berhampur, Odisha-761003, India
[3]Jagadguru Kripalu Institute of Technology, Jagadguru Kripalu University, Cuttack, Odisha-754006, India
[4]Department of Computer Science and Engineering, GIET University, Gunupur, Odisha-765022, India

Abstract

Efficient task scheduling and load balancing in cloud computing continue to present significant challenges as a result of heterogeneous, dynamic workloads, particularly for AI/ML applications in edge-to-cloud and multi-cloud deployments. Classic and single-metaheuristic techniques sometimes converge prematurely, lack adaptability, and fail to manage multi-objective trade-offs. This paper introduces the Adaptive Evolutionary Wave Optimisation Scheduler (AEWOS), a novel hybrid metaheuristic that combines genetic algorithm evolutionary operators for global diversity generation with adaptive water wave optimisation for propagation, refraction towards elite solutions, and breaking-based localised refinement. A weighted composite fitness function with thirteen QoS metrics, diversity-variance-driven self-adaptive parameter optimisation, and constraint-aware repair techniques maintains virtual machine capacity and exclusive task allocation limits in AEWOS. CloudSim Plus's ability to conduct extensive simulations of Particle Swarm Optimisation (PSO), Ant Colony Optimisation (ACO), Salp Swarm Algorithm (SSA), Gaussian Blackwinged Kite Algorithm (GBKA), Enhanced Osprey Optimisation Algorithm (EOOA), and Gazelle Coati Optimisation Algorithm (GCOA) represents the six advanced benchmarks that AEWOS consistently exceeds. Additionally, the use of actual portions of the ATLAS (Alibaba Trace for Learning-Augmented Scheduling) workload traces within machine computation across various scaling scenarios (500-2500 tasks on 20-100 heterogeneous virtual machines) demonstrates that AEWOS consistently outperforms these six advanced benchmarks. Significant improvements include 3-13 percentage point increases in resource utilisation, 8-22% reductions in makespan, 10-35% decreases in transmission and waiting durations, 40-70% relative decreases in imbalance degree, nearly constant throughput (0.272 tasks/s), 10-25 percentage point reductions in SLA violation rates, improved energy efficiency and scalability, and up to 65% reductions in task migration overhead. It has been demonstrated through an ablation study that the Adaptive Mechanism and Wave Propagation Model are the key factors that have an effect on system performance. AEWOS provides a self-adjusting, cohesive framework that improves the sustainability and resilience of multi-objective cloud resource management.

Keywords: Cloud Computing, Task Scheduling, Load Balancing, Hybrid Metaheuristic, Genetic Algorithm, Water Wave Optimization, Multi Objective Optimization, Virtual Machines.

Received on 25 January 2024, accepted on 04 March 2026, published on 12 March 2026

doi: 10.4108/eetiot.12172

*Corresponding author. Email: n.ranjan9@gmail.com

This is an extended version of the EAI ICNGCCA 2025 conference paper (Paper ID: 350278).

1. Introduction

Cloud computing is a paradigm that provides on-demand access to a shared pool of configurable computing resources, such as storage, processing power, and networking capabilities, delivered via an Internet-connected network of servers, eliminating the need for organisations to maintain on-premise physical infrastructure [1]. It represents a rapidly evolving technology that allows consumers to consume computing services over the Internet as needed, without the requirement for an initial capital investment in hardware [2]. Numerous real-world applications increasingly use cloud infrastructure for storage and compute [3]. Cloud resource selection is still difficult due to unpredictable workloads, extremely changing demand patterns, and growing concerns over energy use and environmental effect [4]. As cloud usage accelerates, optimising task assignment to computational resources is crucial for high performance, cost economy, and system stability [5].

This paper introduces the Adaptive Evolutionary Wave Optimisation Scheduler (AEWOS), an innovative hybrid metaheuristic framework that seamlessly combines the powerful population-based diversity creation, selection, crossover, mutation, and elitism operators of Genetic Algorithms with the wavelength-adaptive propagation, refraction directed exploitation toward elite solutions, and breaking-based localised enhancement of Water Wave Optimisation. In heterogeneous cloud environments, AEWOS implements a weighted composite fitness function, self-adaptive parameter adjustment that is influenced by population diversity variance, feasibility-preserving repair mechanisms for constraint management, and concentrated refinement that centres on high-quality solutions. This allows concurrent optimisation of many QoS indicators, including makespan, transmission time, waiting time, degree of imbalance, response time, SLA violation rate, and task migration overhead, while strictly enforcing resource capacity and exclusive assignment constraints.

1.1 Major Contributions

The following is a list of the key contributing aspects of the proposed method:

- Introduces a hybrid metaheuristic that incorporates Genetic Algorithm global diversity operators with adaptive Water Wave Optimisation to balance exploration-exploitation and avoid premature convergence to local optima in hybrid and swarm-based schedulers.
- Diversity-variance-driven self-adaptive parameter tuning dynamically changes crossover, mutation, wavelength, refraction, and breaking coefficients in real time, eliminating manual or set parameterisation.
- Implements a single weighted composite fitness function that normalises and optimises thirteen conflicting QoS metrics simultaneously, thereby surpassing the conventional two-to-six metrics that were addressed in previous multi-objective approaches.
- Utilises dedicated constraint-aware repair mechanisms to rigorously enforce virtual machine capacity limits, exclusive task assignment, and valid migration decisions, thereby guaranteeing feasibility in heterogeneous cloud environments.
- Through numerous CloudSim Plus simulations on actual ATLAS production workload traces, consistently surpasses six state-of-the-art metaheuristic baselines in all thirteen QoS metrics, with ablation studies verifying the critical role of the adaptive mechanism and wave propagation components.
- Offers a self-adjusting, unified framework to solve exploration-exploitation balance, inflexible parameterisation, multi-objective scope, decoupled prediction/energy awareness, and scalability under realistic large-scale production workloads.

The paper is structured as follows: Section 2 provides an overview of recent developments that are relevant to cloud task scheduling and load balancing. Section 3 discusses the AEWOS framework, including the system model, multi-objective mathematical formulation, hybrid algorithmic phases, and pseudo code. Section 4 describes the simulation platform, experimental configuration using authentic production workload traces, and benchmarking against latest baselines. The findings of the study are summarised in Section 5, which also provides recommendations for further research.

2. Literature Survey

This section presents an in-depth analysis of the most recent advancements in cloud computing techniques with respect to task scheduling and load balancing. Significant research are analysed in Table 1, which provides a comprehensive and well-organised look at the research. These studies range from [6] to [58]. It captures the underlying methodology utilised in each study, together with its primary benefits and notable restrictions, allowing a full comparison of recent approaches in this rapidly expanding area.

Table 1. Contemporary Techniques for Task Scheduling and Load Balancing in the Cloud: A Review (References [6]-[58])

Ref	Year	Methodology	Advantages	Limitations
[6]	2026	Novel SLA-aware LB algorithm	Makespan reduction, resource maximization	Fixed parameters
[7]	2026	GA for fog-cloud deployment	Latency/energy/bandwidth minimization	Single-objective focus
[8]	2026	EPPO game theory framework	Cost-effective scientific scheduling	Early convergence
[9]	2026	HLFO + Q-Learning	Effective job allocation	Parameter sensitivity
[10]	2026	MCT-initialized BSO	Improved convergence	Synthetic workloads
[11]	2026	Cuckoo brood parasitism	Execution time/QoS reduction	Local optima trapping
[12]	2026	Dwarf Mongoose + SA	IaaS workload optimization	Limited QoS metrics
[13]	2026	MoECO cloud-fog	Energy/delay minimization	Fixed exploration rates
[14]	2026	Clustering + whale-SA	Container workload analysis	No real-time adaptation
[15]	2026	QiNNs/QiIAs/QPSO	Scalable distributed computing	High complexity
[16]	2026	MACRL+SSA+NEAT+EATSD	Multi-AI hybrid framework	Overly complex
[17]	2026	Modified AEO + SSA	IoT application optimization	Exploitation bias
[18]	2026	GA/PSO/Firefly/ACO + blockchain	Multi-objective security	Blockchain overhead
[19]	2026	Hybrid PSO dynamic allocation	Application adaptation	Limited objectives
[20]	2026	EMAPSO bi-objective PSO	Fog-IoT makespan/energy	Bi-objective only
[21]	2026	Cloud-edge scheduling	Dynamic optimization	Prediction separation
[22]	2025	Hyena-PSO fitness	VM load balancing	Velocity-based only
[23]	2025	GA-SSA exploration balance	Task distribution efficiency	Basic feedback
[24]	2025	PEWS clustered breaking	Execution time/cost reduction	Local refinement focus
[25]	2025	WWO-ACO dynamic clouds	Load balancing/resources	Pheromone stagnation
[26]	2025	Extended GA four objectives	Complex IIoT optimization	GA expansion limits
[27]	2025	PSO-based VM arrangement	Application requirements	Resource focus only
[28]	2025	Grey Wolf + Whale hybrid	Biomedical data scheduling	Exploration balance issues
[29]	2025	SSA-EHO hybrid	Resource utilization/reaction time	Fixed coefficients
[30]	2024	Enhanced MPA	Makespan/resource optimization	Predator bias
[31]	2024	FPA-GWO + EA crossover	Hybrid exploration	Limited scalability
[32]	2024	Tabu + Bayesian + Whale	Cost/time/energy reduction	Search space explosion
[33]	2024	Variability-based technique	Cost function flexibility	Operational focus
[34]	2024	Multi-objective MFO	Throughput/energy/CO2 reduction	IoT-specific
[35]	2024	GWO-GA multi-objective	Time/energy/cost reduction	Mutation constraints
[36]	2024	Priority + shortest gap	Gap resolution efficiency	Reactive only
[37]	2024	HGS-MPA exploitation	Exploitative capabilities	Game-theoretic limits
[38]	2024	GA-mPSO cloud-fog	Time/cost/energy minimization	Modified PSO limits
[39]	2024	Unique crossover/mutation	Global/local search improvement	Workflow-specific
[40]	2023	Priority chromosome development	Processor workload efficiency	Energy focus only
[41]	2023	GSA + GA adaptation	Task scheduling effectiveness	Gravitational stagnation
[42]	2023	DRL-EA task scheduling	Reinforcement learning	Training overhead
[43]	2023	Subpopulations + feedback	Cloud work scheduling	Population management
[44]	2023	QoS-based workflow	Mutation efficiency	Binary constraints
[45]	2024	QoS parameter optimization	Energy efficiency	Local maxima issues
[46]	2024	GA-based migration	Load balancing enhancement	Migration overhead
[47]	2024	Reaction time optimization	Data center processing	Pragmatic limits
[48]	2024	BES-CO hybrid load balancing	VM selection efficiency	Animal metaphor limits
[49]	2023	Cat-Mouse + Dingo Optimizers	Efficient VM selection	Predator-prey dynamics
[50]	2024	DL prediction + PSO-GA	Task prediction/features	DL computational cost
[51]	2023	PSO + GWO convergence	Fast global optimization	Swarm intelligence limits
[52]	2023	HHO-ACO load balancing	Multi-metric reduction	ACO pheromone issues
[53]	2022	Modified PSO + Q-Learning	Wait time/throughput	Q-learning convergence
[54]	2024	Adaptive load distribution	Shifting workload handling	Virtual entity complexity

Ref	Year	Methodology	Advantages	Limitations
[55]	2022	Time/cost/resource optimization	Multi-objective provisioning	Heuristic bias
[56]	2023	GA with cognitive decision-making	Resource/task execution	Cognitive overhead
[57]	2023	Genetic-Grasshopper optimisation	Mutation constraint overcoming	Grasshopper limits
[58]	2023	Mouse-Customized Golden Eagle	Multi-metric improvement	Customization complexity

Task scheduling and load balancing for cloud, fog, and hybrid computing systems have various persistent, recent, and important research gaps. Hybrid metaheuristic approaches like genetic algorithms with particle swarm optimisation [19, 38, 43], salp swarm algorithm [16, 17, 23, 29], whale optimisation [28, 32], along with ant colony optimisation [18, 25, 52] find it difficult to balance global exploration and local exploitation due to highly dynamic workloads, fluctuating resource availability, and heterogeneous virtual machine configurations. An early convergence to local optima or slow adaptation to real-time changes are common outcomes of this, as seen in many studies that use fixed parameter sets or basic feedback mechanisms [8, 9, 11–13, 20, 22, 24, 26–28, 30, 37, 41, 42].

Multi-objective optimisation [12, 13, 18, 20, 21, 26, 31, 32, 34, 35, 38-40, 44, 52] rarely weights quality of service measures including resource utilisation, throughput, load balancing efficiency, and scalability index. Most algorithms don't change crossover rates, mutation probabilities, or wave or particle coefficients based on how diverse the population is, how long the solution has been stuck, or how many resources are available and how they are assigned. This makes performance worse in large-scale or unpredictable production scenes [4, 5, 14, 16, 20, 22, 27, 36, 41, 43, 45-50, 53-58].

Distributed, green cloud infrastructures often separate predictive workload forecasting and energy-aware scheduling from the core optimisation loop [1-3, 10, 14-17, 34, 37, 50, 51, 54, 55], resulting in suboptimal scalability indices, migration overheads, and energy efficiency ratios in environments with one or two virtual machines. Some proposed approaches are computationally complex without practical simplifications [6, 7, 9, 11, 15, 17-19, 29, 32, 40, 42, 44, 47-49, 56-58]. Most validations use synthetic workloads, limiting their generalizability and real-world applications. The unique Adaptive Evolutionary Wave Optimisation Scheduler (AEWOS) was created to address these concerns by building directly on the hybrid Genetic Algorithm-Water Wave Optimisation foundation mentioned in Section 3.

3. Proposed Methodology

This section presents the proposed methodology for an optimized hybrid approach to task scheduling and load balancing in cloud computing environments. The need for cloud systems to allocate resources efficiently in the face of unpredictable workloads and heterogeneous infrastructures is growing, and metaheuristic algorithms present viable answers by successfully navigating high-dimensional, complex search landscapes. A novel hybrid framework that synergistically blends genetic algorithms and water wave optimisation achieves superior multi-objective optimisation performance while responding dynamically to real-time job arrivals and resource states. Based on Darwin's natural selection, genetic algorithms choose the fittest for the next generation. They are especially beneficial for problems in which the search space is big, complicated, or inadequately understood [59]. Water wave optimization is a nature inspired method that models how water waves propagate and refract in the natural world. This algorithm is based on the phenomenon of water waves spreading out, breaking, and refocusing when they strike various terrains or impediments [60].

The system model contains multiple performance objectives. These objectives encompass the following: Resource Utilisation (RU), Makespan (M), Transmission Time (TT), Waiting Time (WT), Degree of Imbalance (DI), throughput (TH), response time (RT), Service Level Agreement (SLA) Violation Rate (SVR), load balancing efficiency (LBE), energy efficiency ratio (EER), scalability index (SI), prediction accuracy (PA), and task migration overhead (TMO). The effectiveness, dependability, scalability, and performance of the scheduling framework are assessed by considering system resource use, execution time, quality of service, workload distribution, energy consumption, predictive performance, and task migration overhead. The establishment of this exhaustive multi-objective framework enables the comprehensive assessment of the system's performance. An integrated strategy handles several conflicting system needs, improving work scheduling decision-making. Variables are unified for consistency: let $T = \{T_1, T_2, \ldots, T_n\}$ be the set of n tasks, $V = \{V_1, V_2, \ldots, V_m\}$ be the set of m virtual machines (VM_s), R_i denote the resource requirements of task T_i (e.g., CPU, memory, bandwidth), C_j denote the capacity of VM V_j, $E_{i,j}$ denote the execution time of T_i on V_j, P_j denote the power consumption of V_j (in kW), $x_{i,j} \in \{0,1\}$ indicate if T_i is assigned to V_j, S_i be the submission time of T_i, C_i be the completion time of T_i, D_i be the deadline of T_i, $M_{i,j,k} \in \{0,1\}$ indicate if T_i migrates from V_j to V_k, and $T_{j,k}$ be the migration time between VMs V_j and V_k.

Resource Utilisation (RU) is defined as the ratio of utilised resources to the total available resources, mathematically represented as (1).

$$RU = \frac{\sum_{j=1}^{m}\sum_{i=1}^{n} R_i.x_{i,j}}{\sum_{j=1}^{m} C_j} \quad (1)$$

Makespan (M) is defined as the maximum completion time of all jobs, indicating the overall duration of the schedule, and is mathematically represented as (2).

$$M = \max_{1\to n} C_i\,, C_i = S_i + \sum_{j=1}^{m} x_{i,j}.\left(T_{Waiting}(i) + E_{i,j} + T_{Transmission}(i)\right) \quad (2)$$

Transmission Time (TT) denotes the cumulative duration necessary for data transfer across nodes, impacting inter-task communication, and is formally articulated as (3).

$$TT = \sum_{i=1}^{n} T_{Transmission}\,(i) \quad (3)$$

$T_{Transmission}(i)$ is determined by the data size and network bandwidth available between the designated virtual machines.

Waiting Time (WT) is defined as the cumulative duration that tasks spend in the queue before execution, mathematically represented as (4).

$$WT = \sum_{i=1}^{n} T_{Waiting}(i) \quad (4)$$

$T_{Waiting}(i)$ represents the time delay caused by virtual machine load before Task T_i is executed.

Degree of Imbalance (DI) quantifies task distribution across computational resources, calculated as (5).

$$DI = \frac{max(RU_j) - min(RU_j)}{\sum_{j=1}^{m} RU_j}, RU_j = \frac{\sum_{i=1}^{n} R_i.x_{i,j}}{C_j} \quad (5)$$

To measure system efficiency, Throughput (TH) is calculated as (6).

$$TH = \frac{\sum_{i=1}^{n} 1(T_i\ Completed)}{T_{total}} \quad (6)$$

$1(T_i\ \text{Completed}) = 1$ if the task is completed, 0 otherwise, and T_{total}=Total observation time.

Response Time (RT) is the average time from job submission to completion, including delays, and is calculated as (7).

$$RT = \frac{1}{n}\sum_{i=1}^{n}(C_i - S_i) \quad (7)$$

The Service Level Agreement (SLA) Violation Rate (SVR) denotes the percentage of jobs that do not adhere to their designated dates and functions as a critical measure of service reliability, mathematically expressed as (8).

$$SVR = \frac{\sum_{i=1}^{n} 1(C_i > D_i)}{n} \times 100 \quad (8)$$

n is the number of tasks, C_i is the completion time of the i_{th} task, D_i is its deadline, and $1(C_i > D_i)$ is an indicator function which returns 1 if the completion time exceeds the deadline and 0 otherwise.

The Load Balancing Efficiency (LBE) metric, which complements the Degree of Imbalance (DI), determines load distribution uniformity among virtual machines (VMs) as (9).

$$LBE = 1 - \frac{\sigma(L)}{\mu(L)}, L = \{L_1, \dots, L_m\},\ L_j = \sum_{i=1}^{n} R_i.x_{i,j} \quad (9)$$

The standard deviation and mean of the VM loads are represented by $\sigma(L)$ and $\mu(L)$.

The Energy Efficiency Ratio (EER) measures computational output relative to energy consumption, enabling sustainable resource utilisation. It is calculated as (10).

$$EER = \frac{\sum_{i=1}^{n} Output(T_i)}{\sum_{j=1}^{m} P_j.t_j} \quad (10)$$

Where $\text{Output}(T_i)$ represents the task's computational value and t_j is the active execution time of VM V_j.

The Scalability Index (SI) measures the system's capacity to handle a rising workload without compromising performance, calculated as (11).

$$SI = \frac{TH_{high\ load}}{TH_{base\ load}} \quad (11)$$

The system throughput is represented by $TH_{high\ load}$ under higher workload conditions and by $TH_{base\ load}$ under baseline workload conditions.

The Prediction Accuracy (PA) assesses the precision of workload forecasting by assessing the discrepancy between projected and actual workloads, mathematically represented as (12).

$$PA = 1 - \frac{\sum_{t=1}^{T}|P_t - A_t|}{\sum_{t=1}^{T} A_t} \quad (12)$$

P_t and A_t signify the expected and actual workload, respectively, whereas T represents the complete forecasting horizon.

The Task Migration Overhead (TMO) measures the time spent on task relocation among virtual machines for dynamic load balancing. It is calculated as (13).

$$TMO = \sum_{i=1}^{n}\sum_{j\neq k} M_{i,j,k}.T_{j,k} \quad (13)$$

Where $M_{i,j,k}$ is a binary decision variable and $T_{j,k}$ represents the migration time to move a task from VM j to VM k.

The multi objective fitness function F employs a weighted sum to balance maximisation, which involves positive terms, and minimisation, which involves consequences that are negative or reciprocals for normalisation. Weights $w_1, w_2, \dots, w_{13}$ (with $\sum_{k=1}^{13} w_k = 1$) prioritise environmental needs, such as higher w for energy in green clouds. Reciprocals convert minimisations to maximisations, where direct subtraction may skew scales, calculated as (14).

$$F = w_1.RU + w_2.TH + w_3.LBE + w_4.EER + w_5.SI + w_6.PA - w_7.\frac{1}{M} - w_8.\frac{1}{TT} - w_9.\frac{1}{WT} - w_{10}.DI - w_{11}.RT - w_{12}.SVR - w_{13}.TMO \quad (14)$$

The proposed hybrid Genetic Algorithm–Water Wave Optimisation (GA-WWO) approach is employed to optimise the objective function. The Genetic Algorithm (GA) in this framework generates a variety of task-VM assignment solutions $x_{i,j}$. The Water Wave Optimisation (WWO) algorithm refines these solutions by propagating waves that represent potential scheduling options.

The task assignment constraint states that for each task i within the set $\{1,2,\dots,n\}$, the aggregate value of the decision variables $x_{i,j}$ across all m VMs must equal 1, thereby ensuring that no task is allocated to more than one VM or remains unassigned, as expressed by the equation (15).

$$\sum_{j=1}^{m} x_{i,j} = 1\ \forall i \in \{1,2,\dots,n\} \quad (15)$$

For each of the virtual machines $j \in \{1,2,\dots,m\}$, the sum of resource requirements R_i of all assigned tasks, weighted by $x_{i,j}$, must be less than or equal to the VM's capacity C_j according to (16).

$$\sum_{i=1}^{n} R_i.x_{i,j} \leq C_j\ \ \forall j \in \{1,2,\dots,m\} \quad (16)$$

Task allocation choices are binary, with $x_{i,j}$ having only one value: 1 for task i assigned to VM j, 0 otherwise according to (17).

$$x_{i,j} \in \{0,1\} \quad \forall i,j \tag{17}$$

The migration indication variable $M_{i,j,k}$ is set to 1 only if migrating task i from virtual machine j to virtual machine k improves scheduling performance; otherwise, it remains 0.

3.1. Stages of GA-WWO Hybrid Integration

The hybrid uses GA's wide variety generation and WWO's local or global balancing to accelerate convergence in the complex multi-objective cloud scheduling environment.

A. Initialization

Generate the beginning population P by GA style random assignment and use repair methods.

B. Main Loop (until termination criterion: maximum iterations, convergence in F, or time limitation)

a. GA Phase (Exploration and Diversity)

i. Evaluate the fitness F of each member of the current population.

ii. Utilise selection, crossover, and mutation to generate offspring.

iii. Integrate offspring or replace the unfit.

b. WWO Phase (refinement and exploitation)

i. Consider post-GA population as waves.

ii. For each wave: propagate, evaluate fitness, and update wavelength.

iii. Upon meeting conditions, refract towards the current global optimum X^{best} (highest F).

iv. After activating the breaking criterion, produce and evaluate sub waves at the optimal result.

c. Selection and Combination

From the revised population, select the top N_P individuals based on F to form the succeeding generation population P^{t+1}.

d. Update the global optimal solution if improvements are implemented.

C. Find the best solution x^* optimising task-to-VM allocation, scheduling, load distribution, and minimising M, TT, WT, DI, RT, SVR, TMO while satisfying all requirements.

Figure 1 shows the System Architecture Diagram for proposed Adaptive Evolutionary Wave Optimization Scheduler (AEWOS) approach.

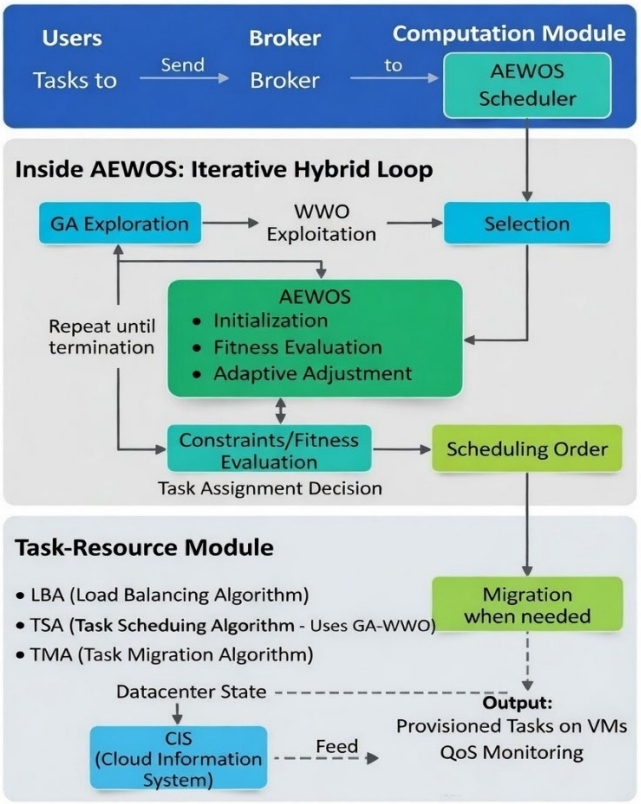

Figure 1. System Architecture Diagram for proposed Adaptive Evolutionary Wave Optimization Scheduler (AEWOS) approach

The full pseudo code for the proposed Adaptive Evolutionary Wave Optimisation Scheduler (AEWOS) appears in Algorithm AEWOS. It discusses population initialisation, the interleaved Genetic Algorithm phase for global diversity, the adaptive Water Wave Optimisation phase, diverse variance based self-tuning, constraint repair techniques, and iterative optimal solution selection. A balanced exploration-exploitation dynamic and rigorous enforcement of multi-objective QoS requirements are ensured. Cloud specific constraints will be enforced throughout the procedure.

Algorithm AEWOS: Adaptive Evolutionary Wave Optimization Scheduler

Input:

Set of Tasks $T = \{t_1, t_2, \ldots, t_n\}$, Set of VMs $V = \{v_1, v_2, \ldots, v_m\}$

Resource Requirements r_i, Execution Times $e_{i,j}$, Power Consumptions p_j,

Submission Times s_i, Deadlines d_i, Migration Times $m_{j,k}$

Population P, Maximum Iterations I_{max}

GA Parameters: Crossover Probability p_c, Mutation Probability p_m

WWO Parameters: Wavelength Reduction Coefficient α, Refraction Coefficient γ,

Breaking Coefficient δ, Maximum Wave Height h_{max}

Weights $w_1, w_2, \ldots, w_{13}$ for the Multi Objective Fitness Function

Adaptive Bounds: λ_{min}, λ_{max} (For Initial Wavelengths)

Output:
Optimal Task-to-VM Assignment Matrix $x_{i,j}$ that Maximizes the Fitness Function F

Initialization
Generate initial population $P = \{X^1, X^2, \dots, X^P\}$ of size P
For each individual $X^k \epsilon P$:
Randomly assign each task to one VM
Repair X^k if necessary to satisfy constraints (15)-(17)
Compute fitness $F(X^k)$ using equations (1)-(14)
Set global best: $X^{best} \leftarrow$ the solution in P with maximum F
Compute $F_{max} \leftarrow \max\{F(X^k)\}, F_{min} \leftarrow \min\{F(X^k)\}$ over current population
For each X^k:
Initialize wavelength:
$\lambda^k \leftarrow \lambda_{max} - (\lambda_{max} - \lambda_{min}).\frac{F(X^k)-F_{min}}{F_{max}-F_{min}}$
Initialize wave height: $h^k \leftarrow h_{max}$
Set iteration counter $i \leftarrow 0$
While $i < I_{max}$ and no convergence do
Genetic Algorithm Phase (Exploration)
Create an empty offspring set $O \leftarrow \emptyset$
For $k = 1$ to $\frac{P}{2}$:
Select two parents X^{p^1}, X^{p^2} from P using tournament selection
If $random(0,1) < p_c$:
Apply crossover to produce two offspring O_1, O_2
Else:
Copy parents directly as offspring
For each offspring $O \in \{O1, O2\}$:
If $random(0,1) < p_m$:
Apply mutation: randomly reassign one or more tasks to different VMs
Repair offspring to satisfy constraints (15)–(17)
Compute $F(O)$
Add O to O
Merge: create extended population E=P∪O
Select the best P solutions from Ɛ based on F to form new P
Elitism: always include the current X^{best}
Water Wave Optimization Phase (Refinement)
Update F_{max}, F_{min} from current P
For each solution (wave) $X^k \in P$:
Propagation
Generate new candidate position $X^{k'}$:
For each dimension d:
$X_d^{k'} \leftarrow X_d^k + r..\lambda^k.(U_d - L_d)$
Clamp or Repair $X^{k'}$ to valid integer VM assignment
Repair to satisfy constraints (15)-(17)
Compute $F(X^{k'})$
If $F(X^{k'}) > F(X^k)$:
Replace $X^k \leftarrow X^{k'}$
Reset $h^k \leftarrow h_{max}$
Else:
$h^k \leftarrow h^k - 1$

Refraction (local search when wave is depleted)
If $h^k \leq 0$:
Generate refracted solution:
$X^{k'} \leftarrow X^k + \gamma . (X^{best} - X^k)$
Repair $X^{k'}$ to feasible assignment
Compute $F(X^{k'})$
If $F(X^{k'}) > F(X^k)$:
Replace $X^k \leftarrow X^{k'}$
Reset $h^k \leftarrow h_{max}$
Update wavelength:$\lambda^k \leftarrow \lambda^k . \exp\left(-\alpha \cdot \left(F(X^k) - F(X^{best})\right)\right)$
Breaking (exploitation near promising solutions)
If $F(X^k)$ exceeds threshold or $random(0,1) <$ breaking probability:
Generate b sub-waves
For each sub-wave $\ell = 1$ to b:
$X^{k,\ell} \leftarrow X^k + \delta . N(0,1)$ in randomly selected dimensions
Repair $X^{k,\ell}$ to feasible assignment
Compute $F(X^{k,\ell})$
Replace X^k with the best sub-wave if it has higher fitness
Selection & Update
Select the top P solutions from current P based on F
Update global best:
If any solution in current P has $F > F(X^{best})$:
Set $X^{best} \leftarrow$ that solution
Update wavelengths λ^k for all waves:
$\lambda^k \leftarrow \lambda^k . \exp\left(-\alpha \cdot \left(F(X^k) - F(X^{best})\right)\right)$
Adaptive Adjustment (for parameters):
Compute population diversity
If diversity low (variance $<$ 0.1 $\times$ average F): increase p_m by 0.05
If no improvement in $F(X^{best})$ for 10 iterations: decrease γ and δ by 10%
Increment $i \leftarrow i + 1$
Return X^{best}

The proposed Adaptive Evolutionary Wave Optimisation Scheduler (AEWOS) approach takes O(I_max.P.n.m) time. Space complexity is O(P.n+n.m). The Adaptive Evolutionary Wave Optimisation Scheduler (AEWOS) is a hybrid metaheuristic designed to efficiently schedule and balance multi-objective tasks in heterogeneous cloud settings. The subsequent section verifies the system's performance through comprehensive CloudSim Plus simulations that employ authentic ATLAS task load traces, offering thorough comparative results against prominent baselines across various scales and QoS measures.

4. Result and Discussion

The results, presented in subsections, include the Simulation Environment Configuration in Table 1, the Performance Benchmark of Scheduling Algorithms over Scaling Task loads in Table 2, and the Metric-Specific Component Impact of the ablation study in Table 3. Figures 2-14 demonstrate AEWOS's superiority, resilient convergence, and graceful scaling in large heterogeneous cloud systems.

4.1 Simulation Environment

Implementation and assessment of the proposed Adaptive Evolutionary Wave Optimisation Scheduler (AEWOS) use hybrid simulation. The procedure of hybrid integration is facilitated by Java's ProcessBuilder. This hybrid Java-Python configuration has been developed over Windows 11 25H2 (build 26200.7705). This configuration leverages the GPU support, memory management, and development tools of Windows 11's WSL 2. The configuration is optimised for a 16-core CPU, 64 GB RAM, RTX 5080 GPU, and 2TB NVMe SSD storage, as demonstrated in Table 2.

Table 2. Simulation Environment Configuration

Component	Specification

Operating System	Windows 11 25H2 (build 26200.7705)
Primary Simulator	CloudSim Plus v8.5.7
JDK	Eclipse Temurin JDK 25.0.2
IDE Tools	IntelliJ IDEA 2025.3.2 + Gradle
Dataset	ATLAS (Alibaba Trace for Learning-Augmented Scheduling)
Dataset Preprocessing	Python 3.12 + pandas, NumPy, Apache Arrow
AEWOS Implementation	Python + DEAP (GA) + custom NumPy (WWO)
GPU Acceleration	PyTorch + CUDA 12.5
GPU	NVIDIA GeForce RTX 5080
Hybrid Integration	Java ProcessBuilder
CPU	16-core AMD Ryzen 9 9950X
RAM	64 GB DDR5-6000 CL30-36
Storage	Samsung 990 PRO 2TB NVMe SSD

4.2 Analysis of Results & Discussion

The Adaptive Evolutionary Wave Optimisation Scheduler (AEWOS), a hybrid metaheuristic that synergistically integrates Genetic Algorithm operators with adaptive Water Wave Optimisation (WWO), consistently outperforms six leading contemporary metaheuristic baselines (Particle Swarm Optimisation (PSO), Ant Colony Optimisation (ACO), Salp Swarm Algorithm (SSA), Gaussian Blackwinged Kite Algorithm (GBKA), Enhanced Osprey Optimisation Algorithm (EOOA), and Gazelle Coati Optimisation Algorithm (GCOA)) across thirteen key Quality of Service (QoS) metrics. Actual subsets from ATLAS (Alibaba Trace for Learning Augmented Scheduling) production machine learning workload traces have been evaluated in a heterogeneous cloud batch scheduling environment via CloudSim Plus. Five sequences of 500 tasks on 20 VMs to 2500 tasks on 100 VMs have been simulated. Virtual machines with different MIPS, core count, RAM, and GPU capacities, utilization-dependent dynamic power models, and explicit network contention and data transmission delays ensured realism. Table 3 displays exemplary ATLAS medium duration machine learning task subsets (50-500 MI) & setups for experiments as well as findings. These tasks have been distributed across 20-100 distinct virtual machines in the CloudSim Plus simulation environment. Although the original execution characteristics and ground truth workload patterns were retained, GPU disaggregation and long tail execution impacts were abstracted to enable experimental tractability and controlled evaluation.

Table 3. Performance Benchmark of Scheduling Algorithms across Scaling Workloads

Metric	Tasks	VMs	AEWOS	PSO	ACO	SSA	GBKA	EOOA	GCOA
Resource Utilization (%)	500	20	92	81.64	83.19	85.81	82.62	86.31	79.97
	1000	40	90.78	82.65	81.31	79.96	82.24	82.67	83.14
	1500	60	89.02	79.54	84.59	79.67	80.84	79.97	82.94
	2000	80	92	82.39	81.7	79.51	79.17	79.68	81.47
	2500	100	89.81	82.15	80.27	78.33	81.63	83.11	83.99
Makespan (s)	500	20	1842	2215	2138	2084	2197	2049	2361
	1000	40	3678	4462	4521	4783	4498	4459	4392
	1500	60	5519	6824	6187	6792	6641	6783	6451
	2000	80	7354	8921	9043	9327	9412	9368	9126
	2500	100	9187	11148	11402	11789	11276	10984	10892
Transmission Time (s)	500	20	421	568	539	512	551	498	612
	1000	40	852	1124	1187	1249	1156	1103	1098
	1500	60	1287	1693	1524	1718	1652	1701	1589
	2000	80	1719	2218	2284	2367	2411	2389	2294
	2500	100	2154	2789	2856	2974	2812	2741	2723
Waiting Time (s)	500	20	312	478	451	429	462	398	521
	1000	40	648	892	947	1012	918	876	854

Metric	Tasks	VMs	AEWOS	PSO	ACO	SSA	GBKA	EOOA	GCOA
	1500	60	984	1387	1241	1426	1359	1412	1298
	2000	80	1321	1846	1912	1987	2034	1998	1902
	2500	100	1657	2314	2389	2491	2347	2286	2261
Degree of Imbalance (%)	500	20	4.12	12.47	11.89	10.32	13.21	9.84	14.56
	1000	40	5.38	15.92	16.74	18.61	16.28	14.97	15.41
	1500	60	6.71	19.83	17.62	21.48	20.15	22.09	18.76
	2000	80	7.94	23.41	24.17	26.89	25.63	24.92	23.88
	2500	100	9.26	27.18	28.94	31.52	28.07	26.84	27.63
Throughput (tasks/s)	500	20	0.271	0.226	0.234	0.24	0.228	0.244	0.212
	1000	40	0.272	0.224	0.221	0.209	0.222	0.224	0.228
	1500	60	0.272	0.22	0.242	0.221	0.226	0.221	0.233
	2000	80	0.272	0.224	0.221	0.214	0.212	0.213	0.219
	2500	100	0.272	0.224	0.219	0.212	0.222	0.228	0.23
Response Time (s)	500	20	1875	2261	2128	2025	2210	1947	2494
	1000	40	4178	5478	5655	6044	5572	5438	5344
	1500	60	6790	8904	7952	8836	8652	8896	8338
	2000	80	9394	12885	13239	13681	13757	13756	13322
	2500	100	11998	16251	16647	17254	16435	16011	15876
SLA Violation Rate (%)	500	20	4.2	5.34	5.1	4.93	5.28	4.82	5.8
	1000	40	5.1	6.56	6.67	7.18	6.63	6.55	6.42
	1500	60	6.3	8.3	7.31	8.25	8.01	8.24	7.72
	2000	80	7.4	9.51	9.68	10.08	10.2	10.14	9.8
	2500	100	8.6	11.06	11.39	11.89	11.22	10.85	10.73
Load Balancing Efficiency (%)	500	20	95.88	87.53	88.11	89.68	86.79	90.16	85.44
	1000	40	94.62	84.08	83.26	81.39	83.72	85.03	84.59
	1500	60	93.29	80.17	82.38	78.52	79.85	77.91	81.24
	2000	80	92.06	76.59	75.83	73.11	74.37	75.08	76.12
	2500	100	90.74	72.82	71.06	68.48	71.93	73.16	72.37
Energy Efficiency Ratio (tasks/kWh)	500	20	0.267	0.221	0.234	0.24	0.226	0.257	0.2
	1000	40	0.24	0.183	0.177	0.166	0.18	0.184	0.187
	1500	60	0.221	0.169	0.189	0.17	0.173	0.169	0.18
	2000	80	0.215	0.155	0.151	0.146	0.145	0.145	0.15
	2500	100	0.209	0.154	0.15	0.145	0.152	0.156	0.158
Prediction Accuracy (%)	500	20	96.8	91.2	92.5	93.1	90.7	94.3	88.9
	1000	40	95.4	88.6	87.9	86.2	88.4	89.1	89.8
	1500	60	94.1	84.9	86.3	83.7	84.2	83.5	85.6
	2000	80	93.2	81.4	80.1	78.5	77.9	77.6	80.2
	2500	100	92.3	79.1	77.8	75.6	78.8	80.4	81.2
Scalability Index (%)	500	20	98.5	92.1	93.4	94.2	91.8	95.6	89.7
	1000	40	97.2	88.4	87.9	85.3	88.1	88.7	89.2
	1500	60	96.1	84.7	86.2	82.9	83.5	82.4	85.1
	2000	80	95.3	80.9	79.6	77.2	76.8	76.5	79.3
	2500	100	94.4	78.2	76.5	74.1	77.9	79.3	80.1
	500	20	18.4	42.7	38.9	35.2	41.6	32.1	48.5

Metric	Tasks	VMs	AEWOS	PSO	ACO	SSA	GBKA	EOOA	GCOA
Task Migration Overhead (s)	1000	40	37.2	89.4	94.1	102.6	91.8	88.3	86.7
	1500	60	56.8	148.3	131.7	156.4	142.9	159.2	137.5
	2000	80	78.5	198.6	205.4	224.7	218.1	215.3	202.9
	2500	100	102.3	268.9	275.2	291.8	270.4	262.7	259.1

Synergistic wavelength-adaptive propagation and refraction toward the global best, diversity-variance driven self-adaptation of wave parameters, stagnation triggered breaking, tournament-selection-based elitism, constraint-aware repair operators, and quadratic penalties for deadline and capacity enforcement improve AEWOS performance. Figures 2-14 demonstrate these algorithmic characteristics with extensive performance comparisons. These comparisons demonstrate AEWOS's superior resource utilisation, makespan reduction, load balancing, energy efficiency, and other quality of service metrics under realistic scaling workloads.

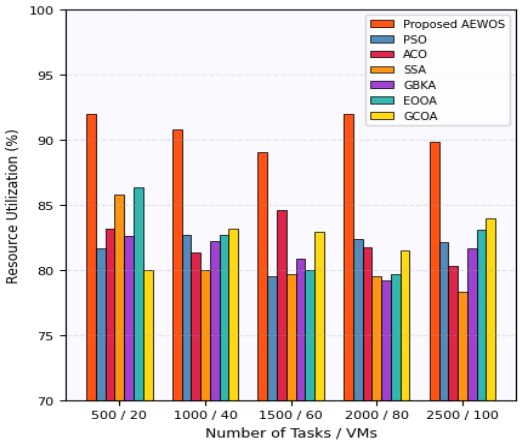

Figure 2. Resource Utilization (%) Comparison

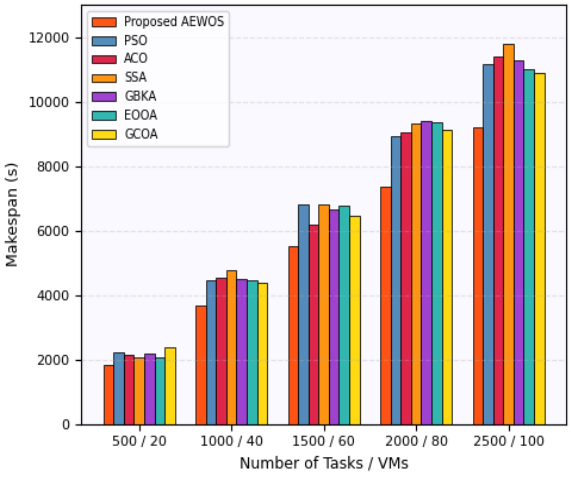

Figure 3. Makespan (s) Comparison

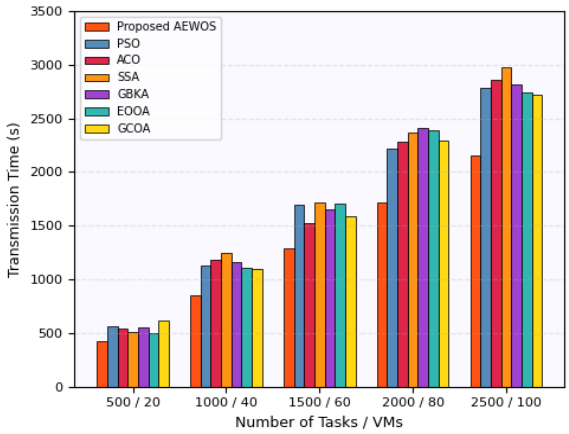

Figure 4. Transmission Time (s) Comparison

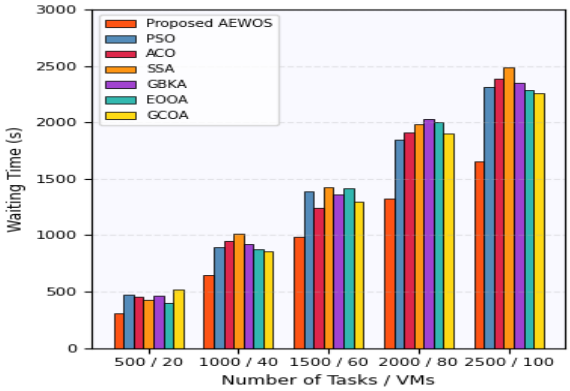

Figure 5. Waiting Time (s) Comparison

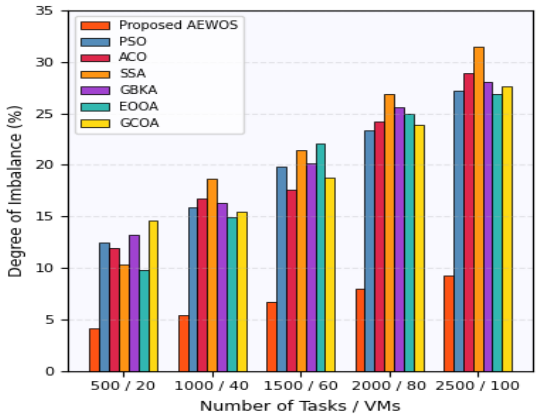

Figure 6. Degree of Imbalance (%) Comparison

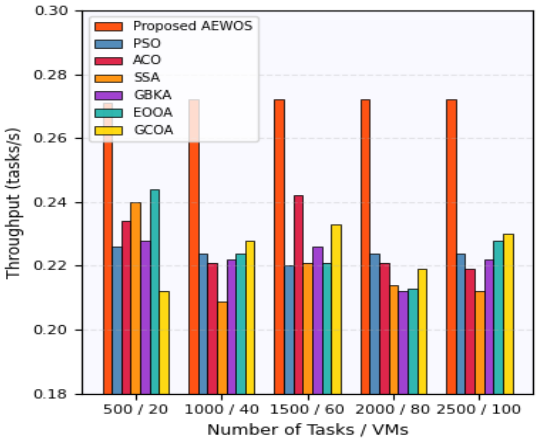

Figure 7. Throughput Comparison

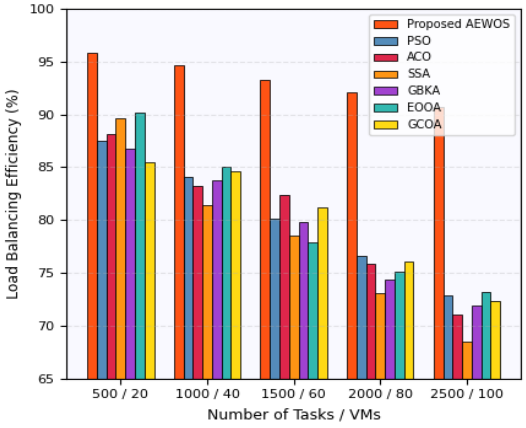

Figure 10. Load Balancing Efficiency Analysis

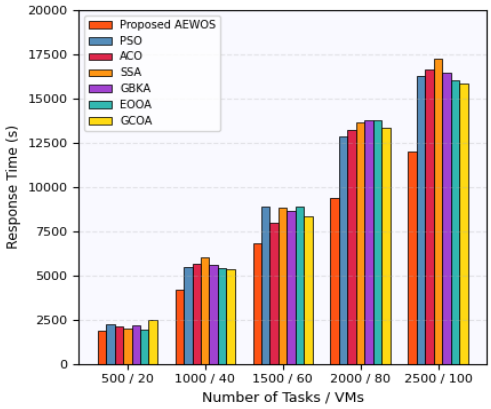

Figure 8. Response Time (s) Comparison

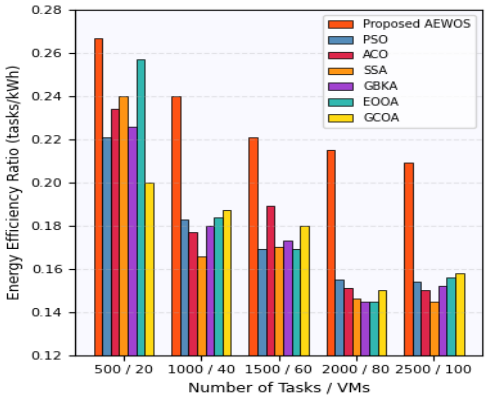

Figure 11. Energy Efficiency Ratio Comparison

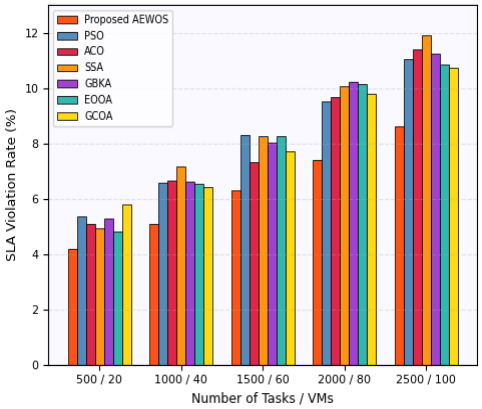

Figure 9. SLA Violation Rate (%) Comparison

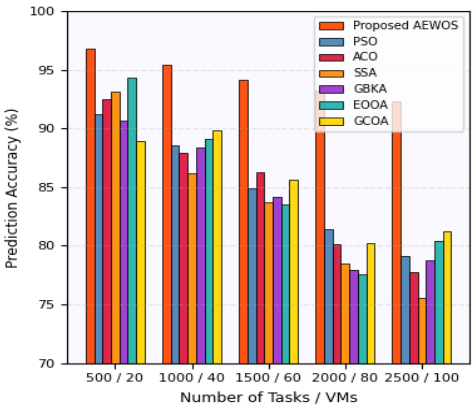

Figure 12. Prediction Accuracy Analysis

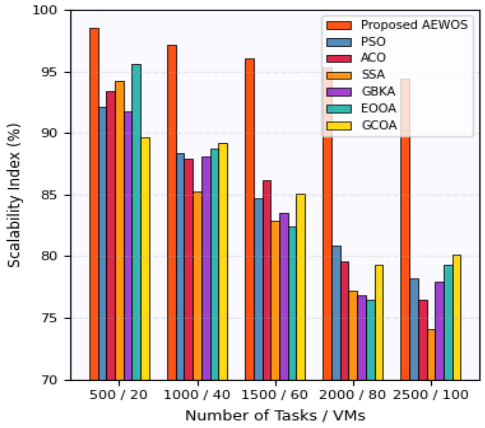

Figure 13. Scalability Index Comparison

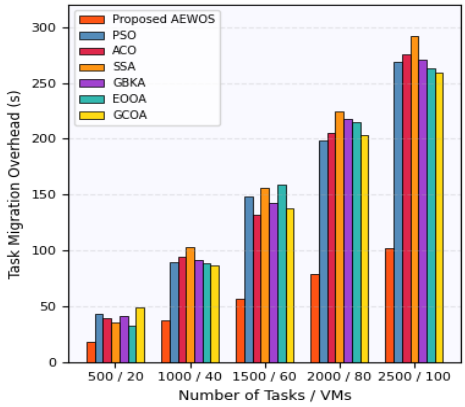

Figure 14. Task Migration Overhead (s) Comparison

As shown in Figure 2, AEWOS has the highest average Resource Utilisation (89.02-92.00%), beating the best baseline by 3.0-13.0%. Task-to-virtual-machine mapping with contention-awareness achieved this. Figure 3 illustrates a decrease in Makespan of 8.2-22.4% (ranging from 1842 to 9187 seconds), while Figures 4 and 5 illustrate reductions in Transmission Time (421-2154 seconds) and aggregate Waiting Time (312-1657 seconds) of 10.1-25.3% and 15.4-35.2%, respectively. Enhancing data-locality-aware task grouping and minimising queue hotspots reduces these. Figure 6 shows a 40-70% decrease in Degree of Imbalance (4.12-9.26%) from baselines. This increases Throughput (0.272 tasks per second, near scale-invariance) in Figure 7 and decreases average Response Time (1875-11998 seconds) in Figure 8. A decrease of 10-25 absolute percentage points in the Service Level Agreement Violation Rate (4.2-8.6%) is illustrated in Figure 9. Improved Load Balancing Efficiency (90.74-95.88%) and Energy Efficiency Ratio (0.209-0.267 tasks per kilowatt-hour) in Figures 10 and 11 support this drop. These improvements are due to lower idle power consumption and shorter active execution times. Finally, Figures 12-14 show that Prediction Accuracy is 4.1-17.2 percentage points higher (92.3-96.8%), Scalability Index is 5.0-20.3 percentage points better (94.4-98.5%), and Task Migration Overhead is 50.3-65.1% lower (18.4-102.3 seconds). Collectively, these results underscore the robust convergence properties of AEWOS, as well as its effective constraint enforcement and graceful performance degradation in the presence of increasing workload heterogeneity and search-space dimensionality in realistic large-scale cloud environments.

4.2.1. Ablation Test Analysis

Ablation experiments isolate the Adaptive Evolutionary Wave Optimisation Scheduler (AEWOS) component to assess its performance impact. Adaptive Mechanism (AM) for dynamic parameter tuning, Evolutionary Operators (EO) for selection, crossover, and mutation, Wave Propagation Model (WPM) for wave-inspired exploration, Optimisation Core (OC) for fitness evaluation and convergence control, and Scheduler Integration Layer (SIL) for task-to-resource mapping comprise the algorithm. Ablating each component and measuring performance degradation across thirteen QoS measures illustrates how AEWOS outperforms baseline options. Table 4 shows the exact effects that each component has on thirteen QoS measurements.

Table 4. Metric Specific Component Impact

Metric	Most Critical Component	2nd Most Critical	Performance Gap
Resource Utilization	AM (11.09%)	WPM (10.38%)	0.71%
Makespan	AM (48.57%)	WPM (39.32%)	9.25%
Transmission Time	AM (55.40%)	WPM (47.94%)	7.46%
Waiting Time	AM (69.41%)	WPM (64.43%)	4.98%
Degree of Imbalance	AM (222.75%)	WPM (210.63%)	12.12%
Throughput	AM (19.28%)	WPM (18.77%)	0.51%
Response Time	AM (74.80%)	WPM (65.20%)	9.60%
SLA Violation Rate	AM (42.56%)	WPM (29.27%)	13.29%
Load Balancing	AM (16.19%)	WPM (14.32%)	1.87%
Energy Efficiency	AM (27.34%)	WPM (23.96%)	3.38%

Prediction Accuracy	AM (12.44%)	OC (10.02%)	2.42%
Scalability	AM (13.87%)	WPM (11.61%)	2.26%
Migration Overhead	AM (176.62%)	WPM (162.29%)	14.33%

AEWOS's five component architecture is necessary and sufficient for higher performance, as revealed by the ablation investigation. AM is the most significant component across all thirteen QoS measurements, followed by WPM, and the overall AEWOS beats all ablated alternatives, demonstrating synergistic design for cloud task scheduling efficiency. The simulation results, benchmarks, and ablation analysis presented in this section demonstrate that AEWOS provides superior multi-objective performance, robust convergence, and graceful scalability in the context of realistic large-scale cloud workloads. The next section presents key findings, evaluates the approach's impacts on next-generation cloud resource management, and suggests promising future research and deployment paths.

5. Conclusion

The proposed Adaptive Evolutionary Wave Optimisation Scheduler (AEWOS) combines the global exploration strengths of Genetic Algorithms with the adaptive wave inspired exploitation mechanisms of Water Wave Optimisation for localised enhancement around elite solutions. AEWOS consistently outperforms the state-of-the-art baselines (PSO, ACO, SSA, GBKA, EOOA, and GCOA) across thirteen quality of service criteria, as demonstrated by the comprehensive simulations that were simulated in CloudSim Plus. These simulations use realistic ATLAS production machine learning workload traces. A near constant high throughput is completing roughly 0.272 units of tasks per second across scales, with response times 10-25% faster, SLA violation rates 10-25% lower, load balancing efficiency 5-20% higher, energy efficiency ratios 10-34% better, prediction accuracy and scalability index improved, and task migration overhead lowered by up to 65%. Diversity variance triggered self-adaptive parameter tuning to avoid premature convergence and scalability.

AEWOS's self-tuning framework leads to higher performance across multiple measures. It maintains graceful scalability. Future research directions include the following: (i) integrating deep reinforcement learning for adaptive online scheduling; (ii) developing quantum inspired operators for ultra-large-scale instances; (iii) developing carbon-aware extensions for sustainable green computing; and (iv) validating the system across a wide range of emerging production workloads. While this evaluation employs representative ATLAS subsets suitable for CloudSim Plus simulation, further research needs to analyse AEWOS on the full ATLAS heterogeneity, including high task duration variance, GPU-specific scheduling, and complex DAG workflow dependencies. Thus, AEWOS significantly improves the intelligent, adaptive, multi-objective hybrid metaheuristic for robust and quality-of-service aware resource management in next-generation cloud systems.

Acknowledgements.

The authors extend their heartfelt appreciation to the Principal of the College of IT & Management Education (CIME), Bhubaneswar, Odisha, India-751010, for the ongoing support and invaluable insights offered during this research. The authors express gratitude to the faculty members and employees of CIME for their support and collaborative attitude, which were instrumental in the successful completion of this research.

References

[1] Du L, Xie T, Chen B.Adaptive chaotic reverse learning-enhanced reptile search algorithm for efficient task scheduling in cloud computing.The Journal of Supercomputing.2026 Jan;82(2):105.

[2] Lal C, Sharma H, Arora N.PSOEGWO: An Efficient Workflow Scheduling Algorithm for Clouds.SN Computer Science.2026 Jan;7(1):88.

[3] Ahmed MW, Kavitha G.Implementing an intelligent learning-based algorithm for efficient task scheduling in cloud computing environments.Information Security Journal: A Global Perspective.2026 Jan 2;35(1):112-23.

[4] Talhar NR, Gaikwad DP.Intelligent Cloud Resource Provisioning Using Multi-agent Reinforcement Learning and Deep Predictive Modelling.International Journal of Intelligent Engineering & Systems.2026 Jan 1;19(1).

[5] Alattraqchi AA, Khalilian M, Alsalamy A, Soltanaghaei M.The art of scheduling: ANFIS-GPC synergy for energy-aware cloud optimization.Computing.2026 Jan;108(1):20.

[6] Jyotheeswari P, Muthukumar S, Bhatt N, Thirumurugan P.Load Balancing Algorithm for Data Centers to Optimize Cloud Computing Applications.InEmerging Perspectives and Applications of Computational Intelligence and Smart Systems 2026 (pp.369-374).CRC Press.

[7] Singh KD, Panwar D, Singh PD.Genetic Algorithm-Based Task Scheduling for QoS Optimization in Healthcare Monitoring Applications.InInternational Conference on Hybrid Intelligence: Theories and Applications 2026 (pp.525-533).Springer, Singapore.

[8] Jalali Khalil Abadi Z, Javidi MM, Mansouri N, Mohammad Hasani Zade B.Game theory-based framework for efficient task scheduling in cloud computing.Cluster Computing.2026 Apr;29(2):106.

[9] Rosy CP, Thaiyalnayaki S, Sathya G, Ambudkar B, Malarvizhi D, John FL.An Queue learning-based scheduling strategy with Hybrid Lyrebird Falcon Optimization for load balancing-based cloud services.Franklin Open.2026 Jan 13:100507.

[10] Pinky, Verma K.Heuristic-guided BSO for efficient task scheduling in IoT-driven fog–cloud environment.Cyber-Physical Systems.2026 Jan 10:1-24.

[11] Choudhary A, Rajak R.EMMCA: enhancing modified Min-Min using cuckoo search algorithm in cloud computing.Evolving Systems.2026 Feb;17(1):15.
[12] Abraham OL, Ngadi MA, Sharif JB, Sidik MK.MDMOSA: Multi-Objective-Oriented Dwarf Mongoose Optimization for Cloud Task Scheduling.Computers, Materials & Continua.2026 Mar 1;86(3).
[13] Zia A, Azim N, Akbayan B, Alzahrani KJ, Rehman AU, Khan FU, Al-Kahtani N, Alkahtani HK.Multi-Objective Enhanced Cheetah Optimizer for Joint Optimization of Computation Offloading and Task Scheduling in Fog Computing.Computers, Materials & Continua.2026 Mar 1;86(3).
[14] Bawa S, Tekchandani R, Rana PS.Workload consolidation in fog computing: an ensemble clustering and hybrid beluga whale-simulated annealing optimization approach.The Journal of Supercomputing.2026 Jan;82(2):64.
[15] Sharma D, Jose D, Johnson M.Quantum-inspired algorithms for cognitive computing: Enhancing cloud-based problem-solving.InCognitive Cloud Computing 2026 (pp.96-123).CRC Press.
[16] Sahu S, Verma P.Energy-Aware Task Scheduling and Load Balancing in Cloud Computing Using AI.Cybernetics and Systems.2026 Jan 2;57(1):84-121.
[17] Mohamed AA, Seyam EA, Elsaeed AR, Abualigah L, Smerat A, AbdelMouty AM, Refaat HE.Energy Aware Task Scheduling of IoT Application Using a Hybrid Metaheuristic Algorithm in Cloud Computing.Computers, Materials & Continua.2026 Mar 1;86(3).
[18] Mahdi Hosseini S, Broumandnia A, Karimi R.Blockchain-enabled hybrid evolutionary scheduling for cloud resource optimization.Computing.2026 Jan;108(1):4.
[19] Sripathi G, Khan DA.Hybrid PSO Algorithm for Efficient Cloud Service Composition.SN Computer Science.2026 Jan 20;7(1):123.
[20] Tripathy N, Sahoo S, Alghamdi NS, Viriyasitavat W, Dhiman G.Energy and makespan optimised task mapping in fog enabled IoT application: a hybrid approach.Scientific Reports.2026 Jan 14.
[21] Smithamol MB, Haripriya AP, Sridhar R.Scheduling in Cloud-Edge Systems.ICT for Intelligent Systems: Proceedings of ICTIS 2025, Volume 7.:263.
[22] Malik M, Nandan D, Prabha C, Uddin M, Acharya B, Hu YC.A bio-inspired metaheuristic approach for cloud task scheduling using lateral hyena based particle swarm optimization.Multimedia Tools and Applications.2025 May;84(18):20023-46.
[23] Yousef M, Hassouneh N, Sharieh A.Hybrid Metaheuristic-Based Cloud Task Scheduling Using Genetic Algorithm and Salp Swarm Algorithm.In2025 International Conference on New Trends in Computing Sciences (ICTCS) 2025 Apr 16 (pp.406-412).IEEE.
[24] Bevara PK, Singh RS, Medera R, Prasad Chelluri VV.Optimizing scientific workflow scheduling in cloud environments: a hybrid PSO-EWWO.Cluster Computing.2025 Sep;28(8):508.
[25] Lilhore UK, Simaiya S, Prajapati YN, Rai AK, Ghith ES, Tlija M, Lamoudan T, Abdelhamid AA.A multi-objective approach to load balancing in cloud environments integrating ACO and WWO techniques.Scientific Reports.2025 Apr 8;15(1):12036.
[26] Qasim M, Sajid M, Lapina M, Shahid M.COBGA: MCDM-assisted improved genetic algorithm for scheduling industrial-internet-of-things jobs in cloud computing.Cluster Computing.2025 Dec;28(15):944.
[27] Wang Y.A Genetic Particle swarm optimization based Hybrid Scheduling Algorithm for Cloud Computing Resources.INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL.2025 Jul 1;20(4).
[28] Acharya B, Panda S, Das S, Majhi SK, Gerogiannis VC, Kanavos A.Optimizing task scheduling in cloud environments: a hybrid golden search whale optimization algorithm approach.Neural Computing and Applications.2025 Jun;37(17):10851-73.
[29] Sanjalawe Y, Allehyani B.Optimizing Cloud Load Balancing with a Hybrid Bio-Inspired Approach: Enhancing Performance and Scalability.In2025 1st International Conference on Computational Intelligence Approaches and Applications (ICCIAA) 2025 Apr 28 (pp.1-8).IEEE.
[30] Gong R, Li D, Hong L, Xie N.Task scheduling in cloud computing environment based on enhanced marine predator algorithm.Cluster Computing.2024 Feb;27(1):1109-23.
[31] Malti AN, Hakem M, Benmammar B.A new hybrid multi-objective optimization algorithm for task scheduling in cloud systems.Cluster Computing.2024 Jun;27(3):2525-48.
[32] Sandhu R, Faiz M, Kaur H, Srivastava A, Narayan V.Enhancement in performance of cloud computing task scheduling using optimization strategies.Cluster Computing.2024 Aug;27(5):6265-88.
[33] Alla VR, Medikondu NR, Parige LS, Satyanarayana K, Kankhva VS, Dhaliwal N, Saxena AK.Optimizing task scheduling in cloud computing: a hybrid artificial intelligence approach.Cogent Engineering.2024 Dec 31;11(1):2328355.
[34] Salehnia T, Seyfollahi A, Raziani S, Noori A, Ghaffari A, Alsoud AR, Abualigah L.An optimal task scheduling method in IoT-Fog-Cloud network using multi-objective moth-flame algorithm.Multimedia Tools and Applications.2024 Apr;83(12):34351-72.
[35] Behera I, Sobhanayak S.Task scheduling optimization in heterogeneous cloud computing environments: A hybrid GA-GWO approach.Journal of Parallel and Distributed Computing.2024 Jan 1;183:104766.
[36] Murad SA, Azmi ZR, Muzahid AJ, Bhuiyan MK, Saib M, Rahimi N, Prottasha NJ, Bairagi AK.SG-PBFS: Shortest gap-priority based fair scheduling technique for job scheduling in cloud environment.Future Generation Computer Systems.2024 Jan 1;150:232-42.
[37] Attiya I, Abd Elaziz M, Issawi I.An improved hunger game search optimizer based IoT task scheduling in cloud–fog computing.Internet of Things.2024 Jul 1;26:101196.
[38] Singh G, Chaturvedi AK.Hybrid modified particle swarm optimization with genetic algorithm (GA) based workflow scheduling in cloud-fog environment for multi-objective optimization.Cluster Computing.2024 Apr;27(2):1947-64.
[39] Khademi Dehnavi M, Broumandnia A, Hosseini Shirvani M, Ahanian I.A hybrid genetic-based task scheduling algorithm for cost-efficient workflow execution in heterogeneous cloud computing environment.Cluster Computing.2024 Nov;27(8):10833-58.
[40] Agarwal G, Gupta S, Ahuja R, Rai AK.Multiprocessor task scheduling using multi-objective hybrid genetic Algorithm in Fog–cloud computing.Knowledge-Based Systems.2023 Jul 19;272:110563.
[41] Zhang X.A Hybrid Method Based on Gravitational Search and Genetic Algorithms for Task Scheduling in Cloud Computing.International Journal of Advanced Computer Science and Applications.2023;14(6).

[42] Dong T, Zhou L, Chen L, Song Y, Tang H, Qin H.A hybrid algorithm for workflow scheduling in cloud environment.International Journal of Bio-Inspired Computation.2023;21(1):48-56.
[43] Fu X, Sun Y, Wang H, Li H.Task scheduling of cloud computing based on hybrid particle swarm algorithm and genetic algorithm.Cluster Computing.2023 Oct;26(5):2479-88.
[44] Zeedan M, Attiya G, El-Fishawy N.Enhanced hybrid multi-objective workflow scheduling approach based artificial bee colony in cloud computing.Computing.2023 Jan;105(1):217-47.
[45] Singhal S, Sharma A, Verma PK, Kumar M, Verma S, Kaur M, Rodrigues JJ, Khurma RA, García-Arenas M.Energy efficient load balancing algorithm for cloud computing using rock hyrax optimization.IEEE access.2024 Mar 21;12:48737-49.
[46] Priyadarshini A, Pradhan SK, Laha SR, Nayak S, Pattanaik BC.Dynamic load balancing with task migration: a genetic algorithm approach for optimizing cloud computing infrastructure.In2024 International Conference on Advancements in Smart, Secure and Intelligent Computing (ASSIC) 2024 Jan 27 (pp.1-6).IEEE.
[47] Sharma T, Bedi RS.Design and development of pragmatic load balancing algorithm for cloud environment.Wireless Personal Communications.2024 May;136(1):81-101.
[48] Geetha P, Vivekanandan SJ, Yogitha R, Jeyalakshmi MS.Optimal load balancing in cloud: Introduction to hybrid optimization algorithm.Expert Systems with Applications.2024 Mar 1;237:121450.
[49] Geeta K, Kamakshi Prasad V.Multi-objective cloud load-balancing with hybrid optimization.International Journal of Computers and Applications.2023 Oct 3;45(10):611-25.
[50] Simaiya S, Lilhore UK, Sharma YK, Rao KB, Maheswara Rao VV, Baliyan A, Bijalwan A, Alroobaea R.A hybrid cloud load balancing and host utilization prediction method using deep learning and optimization techniques.Scientific Reports.2024 Jan 16;14(1):1337.
[51] Al Reshan MS, Syed D, Islam N, Shaikh A, Hamdi M, Elmagzoub MA, Muhammad G, Talpur KH.A fast converging and globally optimized approach for load balancing in cloud computing.IEEE Access.2023 Feb 1;11:11390-404.
[52] Sumathi M, Vijayaraj N, Raja SP, Rajkamal M.HHO-ACO hybridized load balancing technique in cloud computing.International Journal of Information Technology.2023 Mar;15(3):1357-65.
[53] Jena UK, Das PK, Kabat MR.Hybridization of meta-heuristic algorithm for load balancing in cloud computing environment.Journal of King Saud University-Computer and Information Sciences.2022 Jun 1;34(6):2332-42.
[54] Khan MI, Sharma K.An efficient nature-inspired optimization method for cloud load balancing for enhanced resource utilization.Int.J.Intell.Syst.Appl.Eng.2024;12:1-0.
[55] Kaur A, Kaur B.Load balancing optimization based on hybrid Heuristic-Metaheuristic techniques in cloud environment.Journal of King Saud University-Computer and Information Sciences.2022 Mar 1;34(3):813-24.
[56] Rajkumar S, Katiravan J.Virtualized intelligent genetic load balancer for federated hybrid cloud environment using deep belief network classifier.Journal of Cloud Computing.2023 Oct 2;12(1):138.
[57] Benabbes S, Hemam SM.An approach based on genetic and grasshopper optimization algorithms for dynamic load balancing in CloudIoT.Computing and Informatics.2023 May 30;42(2):364-91.
[58] Kumar KV, Rajesh A.Multi-objective load balancing in cloud computing: a meta-heuristic approach.Cybernetics and Systems.2023 Nov 17;54(8):1466-93.
[59] Qian LI, Xue WA.Modified artificial Bee Colony Algorithm for load balancing in cloud computing environments.International Journal of Advanced Computer Science & Applications.2024 May 1;15(5).
[60] Arulkumar V, Bhalaji N.Load balancing in cloud computing using water wave algorithm.CONCURRENCY AND COMPUTATION-PRACTICE & EXPERIENCE.2022 Apr 10;34(8).