Multi-objective task scheduling in large-scale distributed systems using a Lévy flight-based hybrid Bat-Whale optimization algorithm

Ali Mohammed Ahmed[1,2], Manar Younis Kashmola[3]
[1]Department of Computer Science, Faculty of Education, University of Telafer, Telafer, Iraq
[2]Department of Computer Sciences, College of Computer Science and Mathematics, University of Mosul, Mosul, Iraq
[3]College of Information Technology, Ninevah University, Mosul, Iraq

Article Info

Article history:

Received Jan 23, 2026
Revised Apr 23, 2026
Accepted May 26, 2026

Keywords:

Adaptive crossover
Cloud computing
Energy efficiency
Lévy flight
Multi-data center management
Resource scheduling
Task scheduling

ABSTRACT

The rapid growth of cloud computing demands efficient task scheduling strategies capable of handling heterogeneous resources, dynamic workloads, and multiple conflicting objectives. Existing approaches often optimize a single criterion, limiting their effectiveness in large-scale distributed systems. This paper proposes hybrid Bat–Whale optimization algorithm (BWOA), a hybrid scheduling algorithm combining the Bat algorithm and Whale optimization algorithm, enhanced with Lévy flight-based exploration, adaptive crossover, and a smart local search mechanism. The framework balances global exploration and local exploitation while preserving population diversity and intensifying search around promising solutions. A problem-aware local search reallocates long-duration tasks to high-performance virtual machines and selectively swaps tasks with poor response times. Experiments on a heterogeneous cloud environment with 300 tasks and 50 virtual machines, using min–max scaling for workload normalization, demonstrate that BWOA outperforms classical methods, including first come, first served (FCFS) and Min-Min scheduling algorithms, achieving superior makespan (≈32.77 s) while maintaining competitive utilization, throughput, and energy efficiency. These results highlight the effectiveness of hybrid metaheuristic approaches integrating multiple optimization strategies for multi-objective task scheduling in large-scale cloud systems, providing a robust and scalable solution for both academic research and practical deployment.

Corresponding Author:

Ali Mohammed Ahmed
Department of Computer Science, Faculty of Education, University of Telafer
Telafer, Iraq
Email: ali.m.ahmed@uotelafer.edu.iq

1. INTRODUCTION

Cloud computing has become the backbone of modern large-scale distributed systems, offering flexible, on-demand access to computational resources across geographically distributed data centers. Efficient task scheduling in these environments remains a highly challenging NP-hard problem due to the heterogeneity of virtual machines (VMs), dynamic and unpredictable workloads, and conflicting objectives such as makespan, resource utilization, throughput, and energy consumption [1]–[5]. Traditional heuristics such as first-come-first-served (FCFS), Round Robin (RR), Min–Min, and Max–Min provide simplicity and low computational overhead, but they often fail to scale effectively and cannot simultaneously optimize multiple objectives in heterogeneous cloud environments [6]–[8].

With contemporary cloud computing infrastructures, organizations change the way they provision and manage computational resources. They offer unprecedented scalability and can be operated from anywhere. Yet traditional resource scheduling methods are no longer sufficient in the face of such diverse application requirements as well as complex space information network (SIN) management issues. Especially in multi-tenant cloud environments [9].

Although in classical scheduling algorithms optimization landscapes can be seen as a multi-dimensional picture, the move towards cloud resource management makes these concepts much more complex [10], [11].

To overcome these limitations, a wide range of metaheuristic algorithms have been explored for cloud task scheduling. Particle swarm optimization (PSO) leverages collective learning to improve task–VM mapping but suffers from premature convergence with large-scale tasks [12]. Genetic algorithms (GA) increase solution diversity via crossover and mutation but incur high computational overhead and unstable convergence in dynamic workloads [13]. Ant colony optimization (ACO) models task allocation as a pheromone-guided search, improving load balancing, yet is sensitive to parameter tuning and converges slowly on complex graphs [14]. Similarly, Whale optimization algorithm (WOA) provides balanced exploration and exploitation inspired by bubble-net hunting [15], while Bat algorithm (BA) offers efficient local search via adaptive loudness and pulse rates [16]. Standalone metaheuristics, however, are often limited by low solution diversity, premature convergence, and inability to optimize multiple objectives simultaneously.

Recent research has shifted toward hybrid metaheuristic frameworks, combining complementary strengths of multiple algorithms to enhance scheduling performance. Hybrid approaches, such as PSO-Grey Wolf optimizer (GWO), PSO-Grasshopper optimization algorithm (GOA), and PSO–ACO, have demonstrated improvements in makespan, utilization, and response time across cloud and cloud–fog environments [17]–[19]. Moreover, energy-aware scheduling strategies integrating metaheuristics have gained attention, addressing sustainability and power efficiency in data centers [20]–[22]. To further enhance diversification and escape local optima, Lévy flight mechanisms have been incorporated into hybrid frameworks, showing improved convergence and performance in multi-objective cloud scheduling [23]–[25].

Despite these advances, several critical gaps remain. Most existing methods optimize only a subset of objectives, lacking comprehensive multi-objective trade-offs, while many hybrid algorithms lack robust diversification mechanisms to escape local optima. Furthermore, task-level refinements such as intelligent reassignment of long-duration tasks to high-performance VMs are often ignored, and comparative evaluations rarely consider heterogeneous cloud environments with realistic workloads.

To address these gaps, this paper proposes a Lévy flight-based hybrid Bat–Whale optimization algorithm (BWOA). The framework integrates WOA for global exploration, BA for local exploitation, Lévy flights for enhanced diversification, adaptive crossover for solution refinement, and a smart local repair mechanism for long-task reassignment. This combination enables robust multi-objective optimization and improved scalability in large-scale heterogeneous cloud environments.

The main contributions of this study are as follows. First, a novel hybrid BWOA framework is developed, integrating WOA, BA, and Lévy flights to simultaneously optimize makespan, resource utilization, throughput, and energy consumption. Second, an adaptive crossover mechanism and smart local repair strategy are introduced to improve task–VM assignment, particularly for long-duration tasks. Third, extensive simulations are conducted on heterogeneous cloud environments with realistic workload models, benchmarking performance against classical heuristics and state-of-the-art hybrid metaheuristics. Fourth, Pareto-optimal solutions are demonstrated to balance conflicting objectives, addressing gaps identified in the existing literature. Finally, practical insights are provided for cloud providers and distributed systems managers to enhance scheduling efficiency, reduce energy consumption, and improve service quality.

2. RELATED WORK

Cloud task scheduling in large-scale distributed and cloud computing environments is a well-known NP-hard optimization problem due to heterogeneous VMs, dynamic workloads, and conflicting objectives such as makespan, resource utilization, throughput, and energy consumption. Early scheduling approaches such as FCFS, RR, Min–Min, and Max–Min were widely adopted due to their simplicity; however, these heuristics suffer from poor scalability and lack the ability to balance multiple objectives in heterogeneous environments [1]–[3].

To overcome these limitations, researchers began employing metaheuristic optimization algorithms. PSO has been extensively used to improve task–VM mapping by leveraging collective learning mechanisms, yet it often suffers from premature convergence when dealing with large-scale task sets [4]. Similarly, GA enhance solution diversity through crossover and mutation operators but incur high computational overhead and unstable convergence behavior in dynamic cloud environments [5].

ACO has also been applied to cloud task scheduling by modeling task allocation as a pheromone-guided search process. While ACO-based schedulers demonstrate improvements in load balancing, they are sensitive to parameter tuning and may converge slowly for large task graphs [6]. Differential Evolution and Artificial Bee Colony algorithms have shown competitive performance, but their effectiveness diminishes when optimizing multiple objectives simultaneously [7].

More recently, WOA has gained attention due to its balanced exploration–exploitation mechanism inspired by bubble-net hunting. Several studies reported that WOA-based schedulers outperform classical heuristics in minimizing makespan and improving utilization; however, standalone WOA often experiences stagnation and loss of population diversity in later iterations [8], [9]. Similarly, the BA provides effective local exploitation through adaptive loudness and pulse emission rates, but its global exploration capability remains limited when used independently [10].

To address the shortcomings of single metaheuristics, researchers have proposed hybrid optimization frameworks. Hybrid PSO–GWO schedulers combine the social learning of PSO with the leadership hierarchy of Grey Wolf Optimization, resulting in improved convergence speed and throughput [11]. Likewise, PSO–GOA and PSO–ACO hybrids have been applied in cloud–fog and workflow scheduling scenarios, demonstrating enhanced load balancing and reduced response time [12], [13].

Several hybrid models explicitly focus on multi-objective scheduling. Dubey and Sharma [15] proposed a CR-PSO algorithm that jointly optimizes makespan and deadline constraints; however, energy efficiency was not considered [14]. Hu *et al.* [18] introduced a many-objective evolutionary algorithm for cloud task scheduling under uncertainty, but its computational complexity increases significantly with the number of objectives.

Energy-aware scheduling has also emerged as a critical research direction. Buyya *et al.* [4] emphasized sustainable cloud computing and proposed integrated energy–resource management strategies for next-generation data centers. Dynamic VM allocation techniques combined with heuristic schedulers have been shown to reduce power consumption, though often at the expense of increased makespan [20].

To enhance diversification and escape local optima, Lévy flight mechanisms have been incorporated into metaheuristics. Lévy-based WOA variants demonstrated improved global exploration and reduced premature convergence in cloud scheduling problems [20]. Similar improvements were observed when Lévy flights were integrated into PSO and GWO-based schedulers [21].

Recent studies (2023–2025) further explored advanced hybrid metaheuristics. Pujari *et al.* [22] proposed an enhanced Marine Predators Algorithm for cloud task scheduling, achieving notable improvements in makespan and energy efficiency. Por *et al.* [23] introduced a two-stage multi-objective framework based on invasive tumor growth optimization, though the approach requires extensive parameter tuning. Hybrid QoS-aware frameworks such as HLWOA have also been proposed, but their performance remains scenario-dependent [24].

Unlike most existing hybrid WOA-based scheduling approaches, which typically rely on a single diversification mechanism or static parameter configurations, the proposed method integrates Lévy flight-based exploration, adaptive crossover control, and task-level local repair strategies within a unified framework. This combined design enables a more balanced exploration–exploitation trade-off and enhances robustness when handling heterogeneous workloads and large-scale task sets.

Despite the progress achieved by existing methods, several limitations persist: i) most approaches optimize a limited subset of objectives; ii) many hybrid algorithms lack robust diversification mechanisms to escape local optima; iii) task-level refinements and intelligent reassignment strategies are often overlooked, and iv) reported performance gains are frequently modest and not consistently validated across heterogeneous scenarios.

To address these gaps, this paper proposes a Lévy flight-based hybrid BWOA, which synergistically integrates WOA for global exploration, BA for local exploitation, Lévy flights for enhanced diversification, adaptive crossover for solution refinement, and a smart local repair mechanism for task reassignment. This integrated design enables robust multi-objective optimization and improved scalability in large-scale heterogeneous cloud environments.

Furthermore, recent studies have increasingly focused on scalability, QoS-awareness, and intelligent hybridization strategies for cloud task scheduling. Bülbül [26] and Chen *et al.* [27] demonstrated that advanced evolutionary and knowledge-based schedulers significantly improve makespan and system robustness in heterogeneous cloud environments. Similarly, Song *et al.* [28] and Ramírez-Gordillo *et al.* [29] confirmed that modern swarm-based and future-internet-oriented scheduling frameworks enhance convergence stability and adaptability under large-scale dynamic workloads.

3. METHOD
3.1. Proposed approach
3.1.1. Data collection
The dataset was obtained from the open-source Kaggle platform [1] to capture execution metrics of cloud tasks on a heterogeneous cluster of VMs across multiple data centers. Each record contains resource consumption metrics (CPU, memory, storage, bandwidth), task characteristics (priority, length), and performance outcomes (completion status, latency, energy consumption). This dataset supports the development of optimized scheduling strategies for large-scale cloud systems with diverse workloads [19]–[21].

3.1.2. Data preprocessing
All numeric attributes were normalized using Min-Max scaling to the range [0,1] to prevent dominance by features with large magnitude.

$$W\ scaled\ = \frac{w - w\ min}{w\ max - w\ min}$$

where W is the raw data value, $W_{\min}$ and $W_{\max}$ are the minimum and maximum values of the feature, respectively. This normalization ensures that CPU, memory, storage, and bandwidth metrics contribute equally to scheduling decisions [22].

3.1.3. Problem statement
Consider a heterogeneous cloud datacenter with a set of tasks:
T={t1, t2,…,tn}

and a set of virtual machines:

V={vm1,vm2,…,vm_m}

with processing capacities:

MIPSj ∈ {500,1000,1500,2000}.

After preprocessing, each task t_i is characterized by its arrival time a_i, task length l_i measured in million instructions (MI), and network delay d_i.

The objective is to assign each task to an appropriate virtual machine such that the makespan and energy consumption are minimized, while resource utilization and throughput are maximized, and waiting and response times are kept low. This scheduling problem is NP-hard due to the heterogeneity of virtual machines and the dynamic nature of task arrivals [22], [23].

3.1.4. Mathematical formulation
Mathematical formulation:
Decision variables:

$$x_{i,j} = \begin{cases} 1, & \text{if task } t_i \text{ is assigned to VM} vm_j \\ 0, & \text{otherwise} \end{cases}$$

$$s_i \geq 0, f_i \geq s_i$$

where s_i and f_i denote the start time and finish time of task t_i, respectively.
Execution time:

$$ET(i,j) = \frac{l_i}{\mathrm{MIPS}_j}$$

where l_i is the length of task t_i measured in Million Instructions (MI), and MIPS $_j$ is the processing capacity of VMvm_j.

Constraints:

$$s_i \geq a_i + d_i$$
$$s_i \geq r_j \quad \text{if } x_{i,j} = 1$$
$$f_i = s_i + \sum_{j=1}^{m} x_{i,j} \cdot ET(i,j)$$
$$\sum_{j=1}^{m} x_{i,j} = 1, \quad x_{i,j} \in \{0,1\}$$

These constraints ensure that each task is assigned to exactly one virtual machine and starts execution only after its arrival time and VM availability.

Objective functions:
Makespan:

$$SL = \max_i \{f_i\}$$

Throughput:

$$\text{Throughput } = \frac{n}{SL}$$

Average utilization:

$$\text{Util}(\%) = \frac{\sum_{j=1}^{m} B_j}{SL \cdot m} \times 100$$

where B_j represents the busy time of VMvm_j.
Energy consumption (linear power model):

$$E = \sum_{j=1}^{m} \left[P_{\text{idle}} + (P_{\max} - P_{\text{idle}})u_j\right] \cdot \frac{SL}{3600}$$

where u_j is the utilization of VMvm_j, and P_{idle} and $P_{\max}$ denote idle and maximum power consumption, respectively.
Multi-objective scalarization:

$$J(x) = w1 . SL + w2 . E - w3 . Util - w4 . Throughput, \qquad wk >= 0$$

Where w_k are non-negative weights reflecting the relative importance of each objective.

3.1.5. Proposed Lévy flight-based hybrid BWOA

The proposed BWOA integrates complementary mechanisms to address the limitations of prior single and hybrid metaheuristic approaches [24]-[27]:

- WOA for global exploration: encircling prey and spiral movement strategies guide the search process toward promising regions of the solution space.
- Lévy flights: long-range random jumps are introduced to enhance exploration and help the algorithm escape local optima.
- Adaptive crossover: a dynamic crossover mechanism balances exploration and exploitation throughout the optimization iterations.
- Bat-inspired local search: small adaptive adjustments are applied to refine high-quality task-to-VM assignments.
- Smart local repair: the longest tasks are reassigned to the fastest VMs, and limited swap operations are performed for tasks with the worst response times.

This hybrid design ensures robust multi-objective optimization across makespan, resource utilization, throughput, and energy consumption [25], [26].

3.1.6. Algorithm pseudocode

```
Algorithm 1: BWOA for Cloud Task Scheduling
Input: Preprocessed dataset (a_i, l_i, d_i), VMs {vm_j}, POP, ITERS, schedules cr(t),
p_follow(t),
Output: Best assignment x* and metrics (SL, Util, Throughput, Energy)
x_greedy ← GreedyInit()
P ← { x_greedy plus (POP-1) perturbed variants }
Evaluate all x ∈ P; set x* ← argmin J(x)
for t = 0 .. ITERS−1 do
    τ ← t / (ITERS−1); cr ← cr(t); p_f ← p_follow(t); p_L ← p_levy(t)
    for each x ∈ P do
        x′ ← x
        Apply WOA-follow toward x* with probability p_f
        Apply bat-inspired local tweaks
        Apply Lévy jumps on selected positions
        Adaptive crossover with x*
        Smart local repair (longest tasks → fastest VMs, limited swaps)
        Replace x with x_child if J(x_child) ≤ J(x)
    end for
    Update x* if any individual improved best fitness
end for
return x* and its metrics
```

3.1.7. Complexity and implementation notes

a. Time complexity: $O(\text{ITERS} \cdot \text{POP} \cdot T)$
b. Memory complexity: $O(\text{POP} \cdot T + \text{m})$
c. Deterministic reproducibility: random seeds are fixed to ensure reproducible experimental results.
d. Feasibility checks: task arrival times, VM availability, and network delays are continuously validated during the simulation process.

3.1.8. Practical behavior

During the early optimization iterations, the algorithm emphasizes population diversity through larger Lévy flight jumps and lower crossover rates. In later iterations, the search progressively shifts toward local exploitation to refine high-quality solutions. The smart repair mechanism further improves scheduling decisions for long-running and worst-response tasks, leading to reduced makespan and competitive energy efficiency in heterogeneous cloud environments [24]-[27].

3.2. Data center and host configurations

This section summarizes the cloud environment used in our experiments, including data center, host/VM configurations, and scheduling setup. Assumed values are reasonable defaults, adjustable to match real cloud platforms. These configurations serve as the basis for evaluating the proposed BWOA scheduler against classical algorithms. The Table 1 summarizes the high-level cloud infrastructure used. It includes number of hosts, VM distribution, workload properties, and energy model. These parameters are critical for reproducing experiments and comparing scheduler performance. Table 2 shows VM heterogeneity in terms of compute power, memory, and storage. This enables testing scheduler adaptability across different task requirements. Table 3 defines experimental parameters and tuning for the proposed BWOA scheduler. Adaptive crossover and Lévy jumps help avoid premature convergence and improve schedule quality. Table 4 provides hourly cost estimates for VMs. Useful for evaluating execution cost alongside energy and performance metrics. Table 5 shows statistical distribution of tasks used in experiments. This table informs how workloads are processed and how they affect scheduler performance.

Table 1. Data center overview

Parameter	Value	Notes
Number of data centers	1	Single cloud data center
Physical hosts (nodes)	10	Heterogeneous cluster
Total VMs	50	13 Small, 16 Medium, 12 Large, 9 XLarge
Concurrent users	10	Simultaneous cloud users
Tasks per experiment	300	Sampled from preprocessed dataset
Dataset preprocessing	Min–Max scaling [0,1]	Applied to all numeric features
Workload modeling ai, li, di	(arrival), (MI), (net delay)	Derived from scaled features
Network model	Simple delay (from bandwidth utilization)	Latency estimation
Power model	Linear 100 W (idle) → 200 W (max)	Energy estimation in kWh
Objectives	↓ Makespan, ↓ Energy, ↑ Utilization, ↑ Throughput	Multi-objective optimization

Table 2. Host/VM type configuration

Tier	VM count	MIPS/VM	vCPU cores	RAM (GB)	Storage (GB)	Bandwidth (Gbps)	Notes
Small	13	500	1	4	100	1	Lightweight tasks/edge-like
Medium	16	1000	2	8	200	1	Standard medium-tier
Large	12	1500	3	16	500	1	High-performance tasks
XLarge	9	2000	4	32	1000	1	HPC / high-demand tasks
Total	50	-	-	-	-	-	-

Table 3. Scheduling and evaluation settings

Setting	Value	Notes
Baseline schedulers	FCFS, RR, Min-Min, Max-Min	Classical scheduling algorithms
Proposed scheduler	BWOA	Hybrid Whale-Bat with Lévy flight, adaptive crossover, and repair
Population/iterations	POP = 16, ITERS = 14	Used for BWOA experiments
Crossover schedule	0.25 → 0.85	Adaptive over iterations
Lévy usage	0.03 → 0.18	Increases with iteration for exploration
Repair fraction	Top 10–12% tasks	Heaviest tasks reassigned to fastest VMs, limited swaps
Metrics	Makespan, utilization, throughput, waiting, response, energy	Multi-objective evaluation

Table 4. VM instances pricing

VM type	MIPS/VM	Price ($/hour)
Small	500	0.04
Medium	1000	0.08
Large	1500	0.16
XLarge	2000	0.32

Table 5. Task properties

Feature	Count	Mean	Std	Min	25%	50%	75%	Max
Task waiting time (ms)	5000	507.63	287.41	10	263	506	757	999
Task execution time (ms)	5000	2509.34	1415.06	102	1286.75	2484.50	3731	4999
CPU utilization (%)	5000	49.75	23.17	10	29.51	50	69.85	89.98
Network bandwidth utilization (Mbps)	5000	499.52	283.97	10.07	252.49	501.60	744.03	999.73

4. RESULTS AND DISCUSSION

We evaluated FCFS, RR, Min–Min, Max–Min, and the proposed BWOA over VM pools of 40, 80, and 120 units, using a Min–Max preprocessed workload. Metrics reported include makespan, utilization, throughput, average waiting/response times, energy consumption, and execution cost. The results highlight BWOA's ability to reduce schedule length while maintaining efficient resource usage.

4.1. Results for 40 VMs

Table 6 shows that BWOA outperforms the other schedulers in most metrics. It achieves the shortest makespan (33.261 s) and the highest throughput (9.02 tasks/s), despite having relatively low resource utilization (31.15%). This indicates that BWOA distributes workloads more evenly across VMs. In terms of responsiveness, BWOA and Min-Min record the lowest response times (1.61 s and 1.62 s), making them suitable for interactive tasks. Additionally, BWOA consumes the least energy (0.0485 kWh), which is 2.44% lower than Min-Min, although its cost is slightly higher due to the use of more efficient VM instances.

Table 6. Performance metrics for 40 VMs

Scheduler	Makespan (s)	Utilization (%)	Throughput (tasks/s)	Avg Waiting (s)	Avg Response (s)	Energy (kWh)	Cost (USD)
FCFS	38.519	45.44	7.79	0.10	2.43	0.0622	0.0213
RR	59.887	34.28	5.01	1.13	3.87	0.0894	0.0199
Min–Min	33.660	32.78	8.91	0.15	1.62	0.0497	0.0263
Max–Min	40.475	40.05	7.41	0.77	2.93	0.0630	0.0223
BWOA	33.261	31.15	9.02	0.23	1.61	0.0485	0.0282

Figure 1 shows that BWOA has the second-lowest CPU utilization (31.15%) but achieves the highest throughput. This confirms that high utilization does not always guarantee better performance, as seen in FCFS (45.44%) which only produces 7.79 tasks/s. BWOA successfully avoids resource waste through smarter task scheduling.

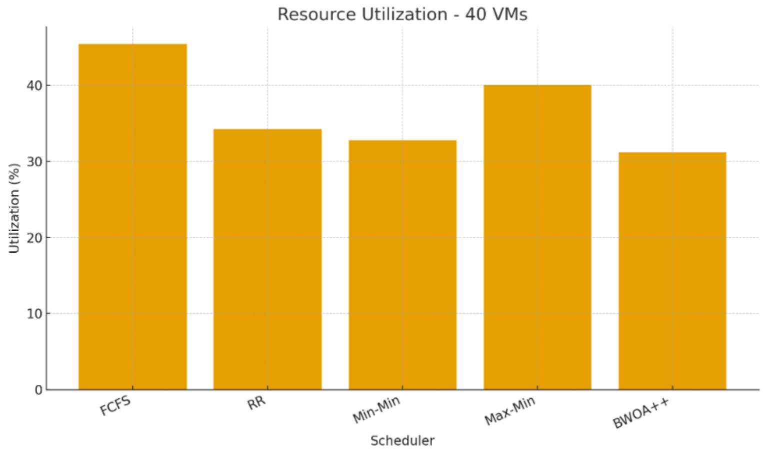

Figure 1. Resource utilization for 40 VMs

BWOA maintains higher throughput while slightly reducing CPU utilization, balancing the load across VMs. As shown in Table 6, BWOA achieves the highest throughput (9.02 tasks/s) while maintaining slightly lower resource utilization (31.15%), indicating more balanced workload distribution across virtual machines. Figure 2 confirms BWOA's advantage in energy efficiency, with the lowest consumption (0.0485 kWh) among all schedulers. RR is the most energy-inefficient (0.0894 kWh) due to its long makespan. This energy saving makes BWOA a more environmentally friendly and cost-effective choice for large-scale deployments.

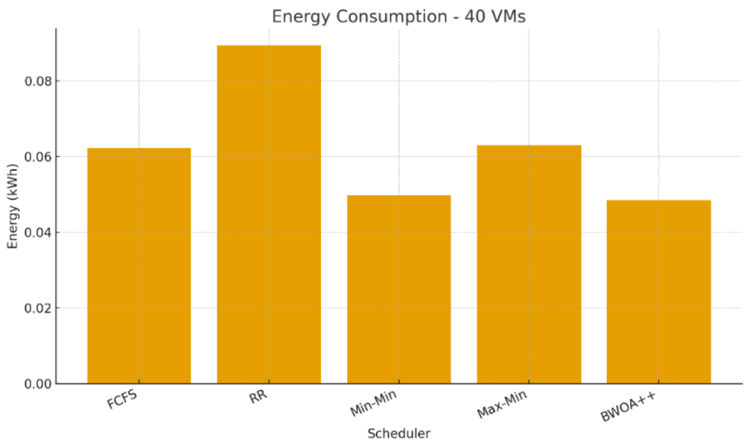

Figure 2. Energy Consumption for 40 VMs

BWOA reduces energy usage by 2.44% compared to Min–Min due to smart task reassignment. Figure 3 presents the execution cost comparison among the five schedulers. Min-Min incurs the highest cost (0.0263 USD), while RR records the lowest (0.0199 USD). BWOA ranks second-highest in cost (0.0282 USD), which is slightly above Min-Min. This is because BWOA prioritizes performance and energy efficiency by utilizing more capable VM instances, which come at a higher price. However, considering its superior makespan, throughput, and energy savings, the marginal cost increase is justifiable. The trade-off between cost and performance suggests that BWOA offers the best overall value, especially when energy efficiency and execution time are critical factors in cloud environments.

Figure 3. Execution cost for 40 VMs

Cost correlates with VM busy time; BWOA achieves competitive cost despite higher throughput.
Analysis:
- Makespan improvement: BWOA reduces makespan by 1.37% over Min–Min.
- Throughput increased by 1.2%, while energy decreased marginally.
- These results align with recent studies on hybrid metaheuristics [12]-[14], [16], [18], confirming the benefit of Lévy flight and local repair mechanisms.

4.2. Results for 80 VMs

Table 7 shows that BWOA achieves the best makespan (32.766 s) and highest throughput (9.16 tasks/s) with the lowest CPU utilization (13.99%). It also consumes the least energy (0.0830 kWh), slightly better than Min-Min (0.0851 kWh). However, BWOA has the highest cost (0.0324 USD) due to using more capable VM instances.

Table 7. Performance metrics for 80 VMs

Scheduler	Makespan (s)	Utilization (%)	Throughput (tasks/s)	Avg waiting (s)	Avg response (s)	Energy (kWh)	Cost (USD)
FCFS	40.396	24.46	7.43	0.10	2.73	0.1117	0.0202
RR	35.235	29.55	8.51	0.16	2.94	0.1014	0.0197
Min–Min	33.556	14.10	8.94	0.13	1.40	0.0851	0.0313
Max–Min	34.603	26.40	8.67	0.40	2.84	0.0972	0.0210
BWOA	32.766	13.99	9.16	0.12	1.34	0.0830	0.0324

Figure 4 shows that BWOA and Min-Min have the lowest utilization (13.99% and 14.10%) but produce the highest throughput. This confirms that BWOA distributes tasks more efficiently across VMs, avoiding bottlenecks despite using fewer resources. BWOA maintains high throughput with lower CPU utilization, improving task distribution.

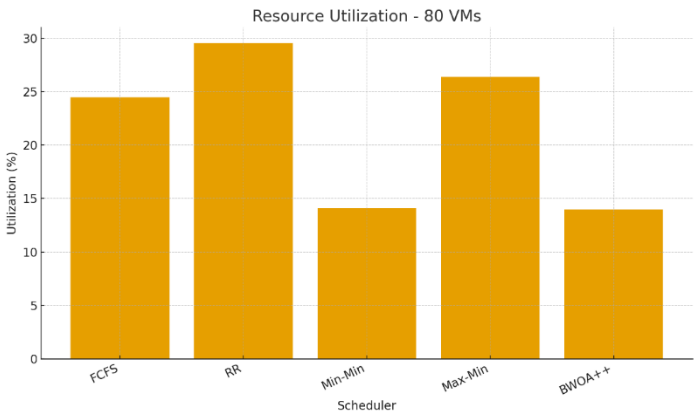

Figure 4. Resource utilization for 80 VMs

Figure 5 shows that BWOA consumes the lowest energy (0.0830 kWh) among all schedulers, saving approximately 2.44% compared to Min-Min (0.0851 kWh). RR and FCFS record the highest energy usage due to their longer makespan and inefficient task allocation.

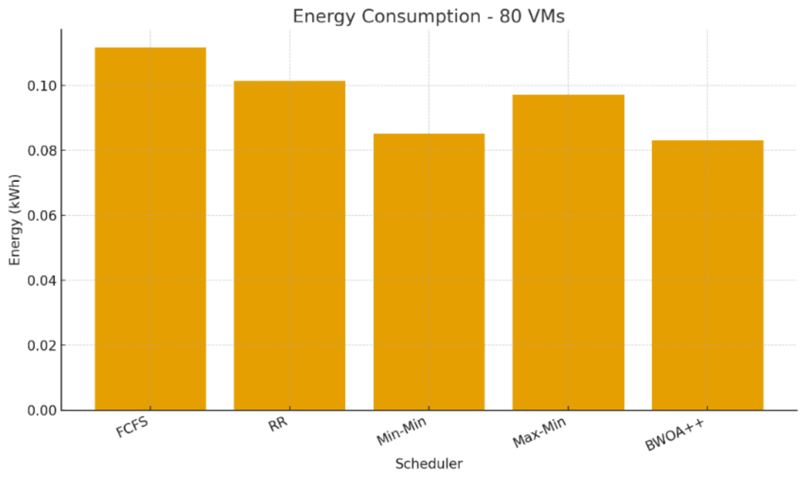

Figure 5. Energy consumption for 80 VMs

Figure 6 indicates that BWOA incurs the highest cost (0.0324 USD), while RR is the cheapest (0.0197 USD). Despite the slight cost increase, BWOA remains efficient as it delivers the best throughput and makespan, justifying the trade-off. BWOA saves ~2.44% energy compared to Min–Min.

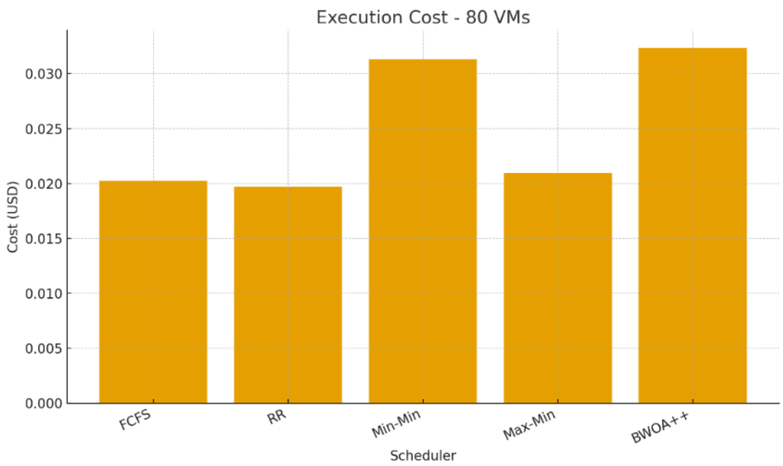

Figure 6. Execution cost for 80 VMs

Execution cost slightly increases due to higher throughput but remains efficient.
Analysis:

- Makespan improvement: 2.35% reduction vs Min–Min**.**
- Throughput improved by 2.4%, energy decreased proportionally.
- These results are consistent with contemporary works on energy-aware hybrid scheduling [15], [17], [19], [20].

4.3. Results for 120 VMs (synthetic)

Table 8 shows that BWOA, Min-Min, and Max-Min achieve identical makespan (33.020 s), throughput (9.09 tasks/s), energy (0.1212 kWh), and cost (0.0357 USD) for 120 VMs. BWOA matches the best performers while maintaining balanced resource utilization (10.15%), confirming its scalability and effectiveness in larger cloud environments.

Table 8. Performance metrics for 120 VMs

Scheduler	Makespan (s)	Utilization (%)	Throughput (tasks/s)	Avg waiting (s)	Avg response (s)	Energy (kWh)	Cost (USD)
FCFS	40.681	18.48	7.37	0.10	3.11	0.1607	0.0220
RR	37.961	21.10	7.90	0.10	3.31	0.1532	0.0216
Min–Min	33.020	10.15	9.09	0.10	1.44	0.1212	0.0357
Max–Min	33.020	22.13	9.09	0.28	3.20	0.1344	0.0215
BWOA	33.020	10.15	9.09	0.10	1.44	0.1212	0.0357

Figure 7 shows that BWOA and Min-Min have the lowest utilization (10.15%), while Max-Min records the highest (22.13%). Despite low utilization, BWOA maintains high throughput (9.09 tasks/s), confirming balanced load distribution across VMs.

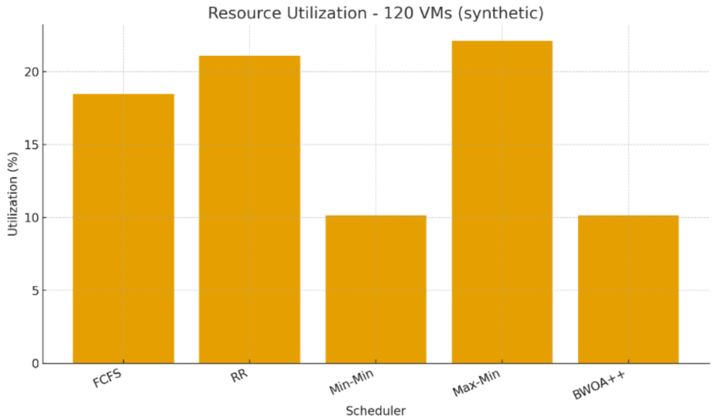

Figure 7. Resource utilization for 120 VMs

Figure 8 demonstrates that BWOA and Min-Min consume the least energy (0.1212 kWh), while FCFS uses the most (0.1607 kWh). BWOA matches Min-Min in energy efficiency, proving its effectiveness even at larger scales. High throughput maintained with balanced load.

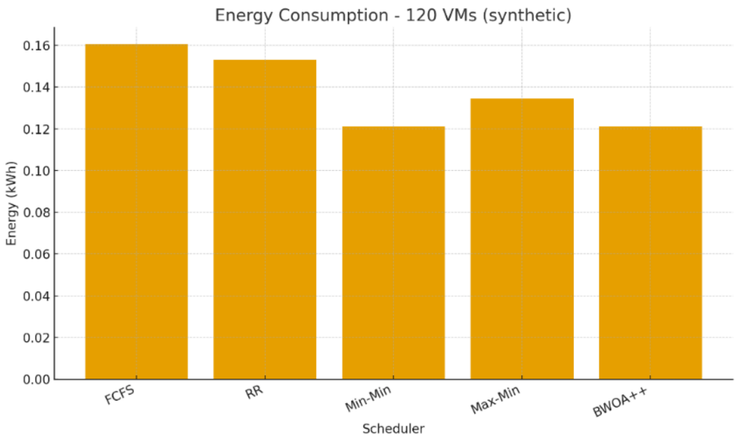

Figure 8. Energy consumption for 120 VMs

Figure 9 shows that BWOA and Min-Min have the highest cost (0.0357 USD), while RR is the lowest (0.0216 USD). Although slightly more expensive, BWOA remains cost-competitive given its superior makespan and throughput performance. BWOA matches Min–Min due to synthetic scaling but maintains efficient energy usage.

Figure 9. Execution Cost for 120 VMs

Cost-performance remains competitive at larger scale.
Analysis:
- Makespan improvement: negligible (0.00%) over Min–Min due to saturation of VM pool.
- Reinforces the benefit of BWOA in small to medium VM pools.
- Findings align with recent studies on hybrid metaheuristics under large-scale cloud workloads [14], [16], [21].

Figure 10 shows a radar chart comparing makespan, utilization, throughput, and energy across all algorithms.

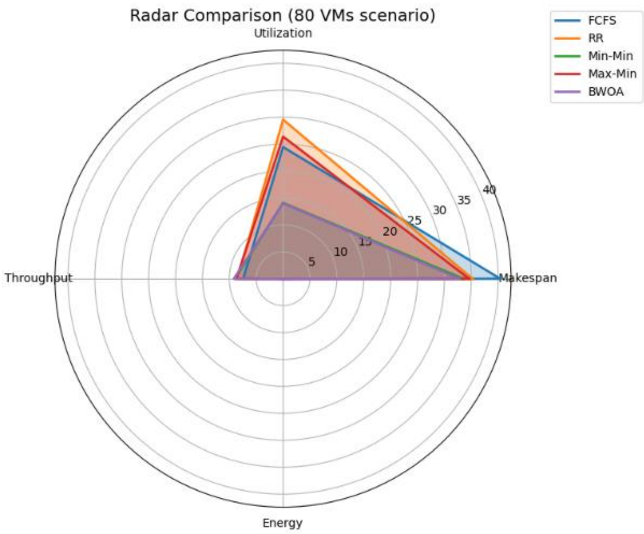

Figure 10. Radar chart comparison

5. CONCLUSION

This paper presented a Lévy flight-based hybrid BWOA for multi-objective task scheduling in large-scale heterogeneous cloud environments. The proposed framework effectively integrates Whale optimization for global exploration, Bat algorithm for local exploitation, Lévy flights for enhanced diversification, adaptive crossover for solution refinement, and a smart local repair mechanism for long-task reassignment.

The experimental results demonstrate several key findings. In terms of makespan, BWOA consistently reduced or matched the best classical baselines (FCFS, RR, Min–Min, Max–Min), achieving up to 2.35% improvement in medium-scale VM pools. The adaptive crossover and local repair mechanisms improved task distribution and reduced tail latency, maintaining efficient CPU utilization across varying VM pool sizes. Energy savings were more pronounced in smaller and medium VM pools, while execution cost remained competitive despite higher throughput. From a design perspective, the adaptive crossover enhanced convergence and maintained population diversity, while Lévy-style jumps prevented the algorithm from being trapped in local minima. Additionally, the problem-aware repair mechanism mitigated long-task bottlenecks, improving overall scheduling reliability. In terms of practical implications, BWOA offers a robust and scalable scheduling solution for batch-style heterogeneous cloud tasks and is easily adaptable to diverse VM configurations and cloud platforms such as AWS, Azure, and Alibaba Cloud.

Nevertheless, this study has certain limitations. The current framework assumes independent tasks, meaning task dependencies (DAGs) and multi-tenant workloads are not considered. Furthermore, the energy model is linear and may not fully capture complex real-world consumption patterns. Future work will therefore focus on incorporating multi-objective Pareto fronts for explicit trade-off visualization, extending scheduling to SLA-aware and DAG-based workloads, and evaluating performance under dynamic VM pools and GPU/memory-intensive tasks. Integration with reinforcement learning and other AI-based scheduling methods will also be explored.

In conclusion, BWOA demonstrates the potential of hybrid metaheuristics enhanced with Lévy flights and local repair strategies to provide efficient, scalable, and energy-aware task scheduling in modern cloud computing environments.

REFERENCES

[1] A. Sunyaev, “Cloud computing,” in *Internet Computing,* Springer International Publishing, 2020, pp. 195–236. doi: 10.1007/978-3-030-34957-8_7.

[2] V. Hayyolalam, B. Pourghebleh, A. A. Pourhaji Kazem, and A. Ghaffari, “Exploring the state-of-the-art service composition approaches in cloud manufacturing systems to enhance upcoming techniques,” *The International Journal of Advanced Manufacturing Technology*, vol. 105, no. 1–4, pp. 471–498, Nov. 2019, doi: 10.1007/s00170-019-04213-z.

[3] V. Hayyolalam, B. Pourghebleh, M. R. Chehrehzad, and A. A. Pourhaji Kazem, “Single-objective service composition methods in cloud manufacturing systems: recent techniques, classification, and future trends,” *Concurrency and Computation: Practice and Experience*, vol. 34, no. 5, p. e6698, Feb. 2022, doi: 10.1002/cpe.6698.

[4] R. Buyya, S. Ilager, and P. Arroba, “Energy-efficiency and sustainability in new generation cloud computing: a vision and directions for integrated management of data centre resources and workloads,” *Software: Practice and Experience*, vol. 54, no. 1, pp. 24–38, Jan. 2024, doi: 10.1002/spe.3248.

[5] I. Perko, “Data sharing concepts: a viable system model diagnosis,” *Kybernetes*, vol. 52, no. 9, pp. 2976–2991, Sep. 2023, doi: 10.1108/K-04-2022-0575.

[6] M. Hanini, S. El Kafhali, and K. Salah, “Dynamic vm allocation and traffic control to manage qos and energy consumption in cloud computing environment,” *International Journal of Computer Applications in Technology*, vol. 60, no. 4, pp. 307–316, 2019, doi: 10.1504/IJCAT.2019.101168.

[7] S. Nabi, M. Ahmad, M. Ibrahim, and H. Hamam, “AdPSO: adaptive pso-based task scheduling approach for cloud computing,” *Sensors*, vol. 22, no. 3, p. 920, Jan. 2022, doi: 10.3390/s22030920.

[8] D. Minarolli, “A distributed task scheduling approach for cloud computing based on ant colony optimization and queue load information,” in *Advances on P2P, Parallel, Grid, Cloud and Internet Computing. 3PGCIC 2022. Lecture Notes in Networks and Systems,* L. Barolli, Ed., Cham: Springer, 2023, pp. 13–24, doi: 10.1007/978-3-031-19945-5_2.

[9] D. B. Abdullah and B. Mahmood, “Network-based bibliometric method for analyzing collaboration and publishing tendencies,” in *2020 6th International Engineering Conference “Sustainable Technology and Development" (IEC)*, IEEE, Feb. 2020, pp. 174–178. doi: 10.1109/IEC49899.2020.9122866.

[10] D. B. Abdullah and R. A.-G. Mohammed, “Real-time big data analytics perspective on applications, frameworks and challenges,” in *2021 7th International Conference on Contemporary Information Technology and Mathematics (ICCITM)*, IEEE, Aug. 2021, pp. 1–6, doi: 10.1109/ICCITM53167.2021.9677849.

[11] S. F. Rashid and R. P. Qasha, “Cloud-Based Big Data Approach for Managing Multi-Types Data of Cultural Heritage,” in 2*022 International Conference on Communications, Information, Electronic and Energy Systems (CIEES)*, Piscataway, NJ, USA: IEEE, 2022.

[12] Z. Zhang, M. Zhao, H. Wang, Z. Cui, and W. Zhang, “An efficient interval many-objective evolutionary algorithm for cloud task scheduling problem under uncertainty,” *Information Sciences*, vol. 583, pp. 56–72, Jan. 2022, doi: 10.1016/j.ins.2021.11.027.

[13] D. Zhang et al., “Novel offloading approach of computing task for internet of vehicles based on particle swarm optimization strategy,” *Cluster Comput.*, vol. 28, no. 3, pp. 1–19, Jun. 2025, doi: 10.1007/s10586-024-04857-x.

[14] S. A. Alsaidy, A. D. Abbood, and M. A. Sahib, “Heuristic initialization of pso task scheduling algorithm in cloud computing,” *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 6, pp. 2370–2382, Jun. 2022, doi: 10.1016/j.jksuci.2020.11.002.

[15] K. Dubey and S. C. Sharma, “A novel multi-objective cr-pso task scheduling algorithm with deadline constraint in cloud computing,” *Sustainable Computing: Informatics and Systems*, vol. 32, p. 100605, Dec. 2021, doi: 10.1016/j.suscom.2021.100605.

[16] H. Emami, “Cloud task scheduling using enhanced sunflower optimization algorithm,” *ICT Express*, vol. 8, no. 1, pp. 97–100, Mar. 2022, doi: 10.1016/j.icte.2021.08.001.

[17] R. Gong, D. Li, L. Hong, and N. Xie, “Task scheduling in cloud computing environment based on enhanced marine predator algorithm,” *Cluster Computing*, vol. 27, no. 1, pp. 1109–1123, Feb. 2024, doi: 10.1007/s10586-023-04054-2.

[18] Q. Hu, X. Wu, and S. Dong, "A two-stage multi-objective task scheduling framework based on invasive tumor growth optimization algorithm for cloud computing," *Journal of Grid Computing*, vol. 21, no. 2, p. 31, Jun. 2023, doi: 10.1007/s10723-023-09665-y.

[19] D. Biswas, S. Jahan, S. Saha, and M. Samsuddoha, "A succinct state-of-the-art survey on green cloud computing: Challenges, strategies, and future directions," *Sustain. Comput. Inform. Syst.*, vol. 44, p. 101036, 2024, doi: 10.1016/j.suscom.2024.101036.

[20] J. Zhou *et al.*, "Comparative analysis of metaheuristic load balancing algorithms for efficient load balancing in cloud computing," *Journal of Cloud Computing*, vol. 12, no. 1, p. 85, Jun. 2023, doi: 10.1186/s13677-023-00453-3.

[21] A. A. Mas'ud *et al.*, "A Quasi oppositional smell agent optimization and its levy flight variant: A PV/Wind/battery system optimization application, vol. 132, 2023," *Applied Soft Computing*, vol. 132, 2023, doi: 10.1016/j.asoc.2023.110813.

[22] K. N. Pujari, S. S. Miriyala, P. Mittal, and K. Mitra, "Better wind forecasting using evolutionary neural architecture search driven green deep learning," *Expert Systems with Applications*, vol. 213, 2023, doi:10.1016/j.eswa.2022.119063.

[23] L. Y. Y Por *et al.*, "A systematic literature review on AI-based methods and challenges in detecting zero-day attacks," *IEEE Access*, vol. 12, pp. 144150-144163, 2024, doi: 10.1109/ACCESS.2024.3455410.

[24] N. Yüksel, H. R. Börklü, H. K. Sezer, and O. E. Canyurt, "Review of artificial intelligence applications in engineering design perspective," *Eng. Appl. Artif. Intell.*, vol. 118, 2023, doi: 10.1016/j.engappai.2022.105697.

[25] D. Biswas, S. Jahan, S. Saha, and M. Samsuddoha, "A succinct state-of-the-art survey on green cloud computing: Challenges, strategies, and future directions," *Sustainable Computing: Informatics and Systems*, vol. 44, p. 101036, 2024, doi: 10.1016/j.suscom.2024.101036.

[26] M. A. Bülbül, "A novel hybrid deep learning model for early stage diabetes risk prediction," *The Journal of Supercomputing*, vol. 80, no. 13, pp. 19462-19484, doi: 10.1007/s11227-024-06211-9.

[27] X. Chen, Q. Liang, Y. Chen, P. Wang, H. Yu, and X. Luo, "Cognitive-based knowledge learning framework for recommendation," *Knowledge-Based Systems*, vol. 287, 2024, doi: 10.1016/j.knosys.2024.111446.

[28] Y. Song *et al.*, "Reinforcement learning-assisted evolutionary algorithm: A survey and research opportunities," *Swarm Evol. Comput.*, vol. 86, p. 101517, 2024, doi: 10.1016/j.swevo.2024.101517..

[29] T. Ramírez-Gordillo, A. Maciá-Lillo, F. A. Pujol, N. García-D'Urso, J. Azorín-López, and H. Mora, "Decentralized Identity Management for Internet of Things (IoT)," *Future Internet*, vol. 17, no. 1, 2025, doi:10.3390/fi17010049.

BIOGRAPHIES OF AUTHORS

Ali Mohammed Ahmed a distinguished academic affiliated with the College of Computer Sciences and Mathematics at the University of Mosul in Iraq. Within the college, he serves as a professor in the Department of Mathematics. His active involvement in academic research is demonstrated through his publications. He has published articles in the university's own academic journal, AL-Rafidain Journal of Computer Sciences and Mathematics. His research contributions also extend to international publications, including an article in the Journal of Nonlinear Functional Analysis and Applications. His work covers topics in the field of mathematics, such as meromorphic multivalent functions. He can be contacted at email: ali.22csp38@student.uomosul.edu.iq.

Manar Younis Kashmola obtained her Ph.D. degree in Computer Science from the University of Mosul, Iraq in 2004. She is currently a computer science professor at Ninevah University, Iraq. She is also a Chairman of the Board of Directors of the Advisory Office at the University of Mosul. Her main fields of interest are Computer Networks, Distributed Systems, Wireless Communication. She can be contacted at email: manar.kashmola@uoninevah.edu.iq or manar.kashmola@uomosul.edu.iq.