

RESEARCH ARTICLE

A Hybrid Metaheuristic Algorithm for Energy-Aware Task Scheduling in Cloud-Edge Computing Environments

Rahil Hafezian¹, Mohammad Fatemi^{2*}¹ Department of Industrial Engineering, Faculty of Engineering, University of Qom, Qom, Iran;² Industrial Engineering and Management, University of Turku, Turku, Finland* Corresponding author. mohammad.fatemi@utu.fi**ARTICLE INFO****Article history**

Received 2026-01-27

Revised 2026-02-17

Accepted 2026-02-20

Keywords

Task scheduling

Energy-aware computing; cloud-edge

computing; hybrid metaheuristic

Grey Wolf Optimizer

ABSTRACT

The growth of latency-sensitive applications at the network edge has made task scheduling a central problem in cloud-edge computing, where energy consumption, response time, and resource utilization must be balanced at the same time. Scheduling in heterogeneous, distributed environments is NP-hard: exact methods do not scale, and single metaheuristics often converge prematurely or exploit the search space poorly. This paper proposes a hybrid metaheuristic that combines the global exploration of the Grey Wolf Optimizer with the focused exploitation of the Whale Optimization Algorithm and a memetic local search. A weighted multi-objective fitness jointly minimizes energy and makespan while maximizing resource utilization. The scheduler is defined within a layered cloud-edge model with a dynamic voltage and frequency scaling energy formulation. The algorithm is specified formally and in pseudocode, and evaluated in an illustrative discrete-event simulation against four baselines on workloads ranging from 100 to 600 tasks. The synthetic results show lower energy and makespan and higher utilization than the baselines, which supports hybrid metaheuristics as a practical approach to energy-aware cloud-edge scheduling.

1. Introduction

The convergence of cloud computing and the Internet of Things has changed how computation is provisioned and consumed. Connected devices now number in the tens of billions, and the data they generate has made a purely centralized cloud insufficient for applications that require low latency. Edge and fog computing address this gap by placing computation closer to where data is produced, forming a layered cloud-edge continuum in which a task may run near the device, at an intermediate fog node, or in a distant data center. The continuum is flexible, but it also creates a demanding scheduling problem: each task must be mapped to a resource that meets its latency requirement without exhausting limited edge capacity or incurring the higher energy and delay costs of the cloud.

Energy is a central concern in this setting. Data centers already account for a substantial and growing share of global electricity use, industry estimates place the figure at roughly one to two percent of the worldwide total and rising, and always-on edge infrastructure adds to that demand. Energy use translates directly into operating cost and environmental impact, so a scheduler that reduces consumption while preserving performance is valuable on both grounds. The difficulty is that energy, makespan, and resource utilization conflict: consolidating tasks to save energy can lengthen completion times, whereas spreading them to reduce makespan can leave resources idle and waste power.

Task scheduling in cloud-edge environments is an NP-hard combinatorial problem, so exact methods become intractable as the number of tasks and resources grows. Simple heuristics such as first-come, first-served or round-robin are fast but yield weak solutions in complex settings. Single metaheuristics are more effective, yet each has a characteristic limitation: some explore the search space broadly but converge slowly, while others exploit promising regions quickly but become trapped in local optima. Hybrid metaheuristics combine complementary algorithms so that the strengths of one offset the weaknesses of another, and they have therefore attracted growing interest as a way to obtain higher-quality schedules in an acceptable time.

This paper designs and evaluates a hybrid metaheuristic scheduler for energy-aware task scheduling in cloud-edge environments. The contributions are threefold. First, the scheduling problem is formulated on a layered cloud-edge model with a dynamic voltage and frequency scaling energy model and a weighted multi-objective fitness that jointly addresses energy, makespan, and resource utilization. Second, a hybrid algorithm is proposed that integrates the exploration mechanism of the Grey Wolf Optimizer with the exploitation mechanism of the Whale Optimization Algorithm and a memetic local search; the procedure is presented as reproducible pseudocode. Third, an illustrative simulation compares the proposed algorithm with four widely used baselines across a range of workloads.

The remainder of the paper is organized as follows. Section 2 reviews related work on metaheuristic and hybrid scheduling in cloud, fog, and edge environments. Section 3 presents the system model, problem formulation, and the proposed hybrid algorithm. Section 4 reports and discusses the numerical results. Section 5 concludes with limitations, managerial insights, and future research directions.

2. Literature Review

The computing paradigms that frame this work are well established. The vision of an Internet of Things in which sensing devices integrate with networked computation set the stage for the data volumes that scheduling must now manage (Gubbi et al., 2013). Edge computing was introduced to address the latency, bandwidth, and privacy limits of sending all processing to the cloud (Shi et al., 2016), and fog computing was proposed as an intermediate layer that extends cloud services toward the network edge with low latency

and wide geographical distribution (Bonomi et al., 2012). Together, these paradigms define the layered continuum in which energy-aware scheduling takes place.

Metaheuristic optimization is the most commonly applied toolkit for the resulting NP-hard scheduling problem. Swarm- and evolution-inspired methods such as the Grey Wolf Optimizer, the Whale Optimization Algorithm, the Salp Swarm Algorithm, the Arithmetic Optimization Algorithm, and Henry Gas Solubility Optimization have been introduced as general-purpose optimizers, each balancing exploration and exploitation through a distinct biological or physical metaphor (Mirjalili et al., 2014; Mirjalili & Lewis, 2016; Mirjalili et al., 2017; Abualigah et al., 2021; Hashim et al., 2019). For problems with several conflicting objectives, elitist multi-objective evolutionary methods established the practice of evolving a well-spread set of trade-off solutions rather than a single point (Deb et al., 2002). These algorithms are the building blocks that scheduling research repeatedly adapts and combines.

A large body of work applies and hybridizes these optimizers for scheduling in cloud and fog environments. Single-objective and multi-objective Grey Wolf formulations have been shown to reduce makespan and improve resource utilization and, in more recent variants, to account for energy and migration costs (Mangalampalli et al., 2023). Henry Gas Solubility Optimization has been combined with whale-based local search and opposition-based learning for cloud task scheduling, which illustrates the recurring pattern of adding an exploitation mechanism to a global optimizer (Abd Elaziz & Attiya, 2021). Hybrid pairings are common: moth search with differential evolution, antlion optimization with elite-based differential evolution, and swarm methods with genetic operators, each intended to reduce the premature convergence of a single algorithm (Abd Elaziz et al., 2019; Abualigah & Diabat, 2021; Abualigah et al., 2021). Surveys of the area consistently report that such hybrids outperform their constituent algorithms by improving both the diversity of candidate solutions and the speed of convergence.

Energy-aware scheduling in fog and cloud-fog settings is now a distinct line of work. A hybrid invasive-weed and cultural-evolution method combined with dynamic voltage and frequency scaling was proposed to minimize energy in fog computing while respecting task precedence (Hosseinioun et al., 2020), and energy-makespan trade-offs have been studied through multi-objective models that assign latency-sensitive tasks to fog resources and compute-intensive tasks to the cloud (Ijaz et al., 2021). Cost- and QoS-aware hybrids that combine swarm search with genetic operators have targeted fog-cloud environments directly (Hussain & Begh, 2022); goal-programming formulations have been applied to multi-objective cloud-fog scheduling (Najafizadeh et al., 2022); and bag-of-tasks and IoT-application scheduling have been addressed with oppositional differential evolution enhanced whale optimization and with chimp-based optimizers (Chhabra et al., 2022; Attiya et al., 2022). Workflow scheduling for big-data IoT applications has likewise been treated with hybrid metaheuristics (Abualigah et al., 2021). Across this literature, three observations recur: scheduling is NP-hard and is dominated by metaheuristics; hybridization reliably improves solution

quality; and energy is increasingly treated as a first-class objective alongside makespan and utilization. Fewer studies, however, present a single, reproducible treatment that combines a layered cloud-edge model, a DVFS energy formulation, and an exploration-exploitation hybrid with a clearly specified multi-objective fitness. The framework developed here is intended to close that gap.

3. Methodology

3.1 System Model

The system is modeled as a three-layer cloud-edge continuum, illustrated in Figure 1. At the bottom, an IoT and end-device layer generates independent tasks, each defined by a length in millions of instructions and a latency expectation. An intermediate edge and fog layer comprises resource-constrained nodes that offer low latency but limited processing capacity. At the top, a cloud layer provides high-capacity data centers at the cost of greater latency and higher energy use per unit of work. A set of virtual machines, each with a defined processing rate, is distributed across the edge and cloud layers. The scheduler, which is the focus of this work, sits between the layers and maps each incoming task to a virtual machine.

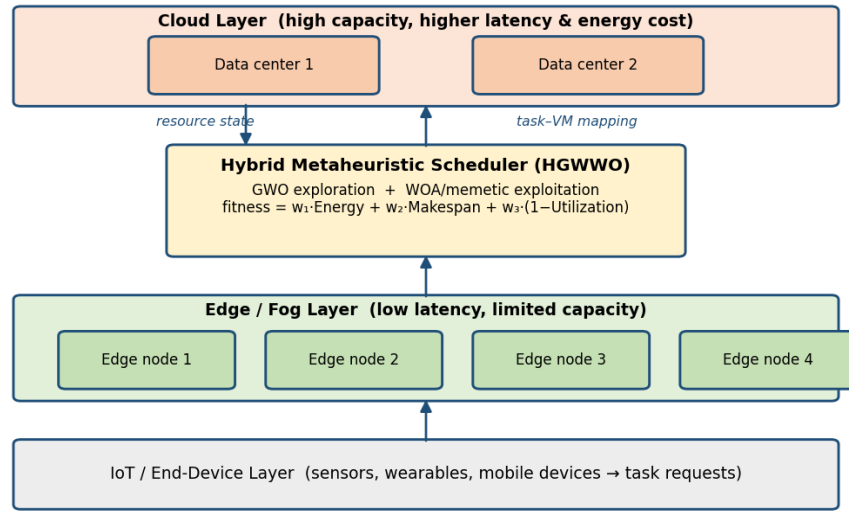


Figure 1. Layered cloud-edge system model with the proposed hybrid metaheuristic scheduler mediating task-to-VM mapping.

3.2 Problem Formulation

Given a set of tasks and a set of virtual machines, a schedule is an assignment of each task to exactly one machine. The execution time of a task on a machine is the ratio of the task length to the machine's processing rate, and the completion time of a machine is the sum of the execution times of the tasks assigned to it. The makespan is the maximum completion time across all machines, as expressed in Eq. (1).

$$\text{Makespan} = \max_{j \in \text{VM}} \left(\sum_{i \in \text{tasks}} \text{Length}_i / \text{MIPS}_j \right) \quad (1)$$

Energy consumption follows a dynamic-voltage-and-frequency-scaling model in which each machine consumes power at an active rate while executing tasks and at a lower idle rate otherwise. The total energy is the sum across machines of active energy during busy periods and idle energy during the remaining time up to the makespan, as shown in Eq. (2).

$$E = \text{sumover } j \text{ of } (P_{active,j} \cdot T_{busy,j} + P_{idle,j} \cdot (Makespan - T_{busy,j})) \quad (2)$$

Resource utilization is the mean ratio of busy time to makespan across machines. The three objectives are combined into a single weighted fitness function to be minimized. Utilization enters as its complement so that every term is minimized, and the weights, which sum to one, express the relative importance of each objective, as given in Eq. (3).

$$Fitness = w^1 \cdot E_{norm} + w^2 \cdot Makespan_{norm} + w^3 \cdot (1 - Utilization) \quad (3)$$

3.3 Proposed Hybrid Algorithm

The proposed Hybrid Grey-Wolf and Whale Optimizer (HGWWO) combines three complementary search behaviors. The Grey Wolf Optimizer supplies a global exploration mechanism in which candidate solutions are guided by the three best solutions found so far, following the leadership hierarchy of a wolf pack. The Whale Optimization Algorithm provides an exploitation mechanism based on a spiral encircling movement toward the best solution, thereby intensifying the search in promising regions. A memetic local search then reassigns tasks from the most heavily loaded machine to refine the schedule. In each iteration, a control parameter that decreases linearly over the run shifts the search from exploration toward exploitation, and a probability switch selects between the wolf-guided and whale-spiral updates. Because solutions are continuous vectors while task assignments are discrete, the largest-ranked-value decoding maps each continuous position to a valid task-to-machine assignment before fitness is evaluated. The procedure is summarized in the following pseudocode.

Algorithm 1: HGWWO for Energy-Aware Cloud-Edge Scheduling

Input: tasks, VMs, population size N, max iterations T,

weights (w_1, w_2, w_3), switch probability p

Initialize a population of N continuous position vectors at random

Decode each position (largest-ranked-value) and evaluate Fitness (Eq. 3)

Identify alpha, beta, and delta (the three best solutions)

for t = 1 to T do

$a \leftarrow 2 - 2 \cdot t/T$ (exploration $\rightarrow$ exploitation control)

 for each solution X in population do

 if rand < p then

$X \leftarrow \text{GWO_update}(X, \alpha, \beta, \delta, a)$ (exploration)

```

else
    X ← WOA_spiral_update(X, alpha)      (exploitation)
end if
X ← memetic_local_search(X) (reassign tasks of busiest VM)
decode X and evaluate Fitness
end for
update alpha, beta, delta
end for
Output: alpha (best schedule), its energy, makespan, utilization

```

The wolf-guided update moves each solution toward the average of three candidate positions derived from the alpha, beta, and delta solutions, with the step size modulated by the control parameter so that movement is large early in the run and small later. The whale-spiral update moves a solution along a logarithmic spiral around the best solution, as expressed in Eq. (4), where the distance to the best solution and a random spiral parameter govern the trajectory.

$$X(t + 1) = D' \cdot e^{bl} \cdot \cos(2\pi l) + X_{alpha}(t) \quad (4)$$

3.4 Experimental Setup

Because a production cloud-edge testbed is unavailable for this study, the algorithm is evaluated using an illustrative discrete-event simulation that follows a CloudSim-style model. The results in the next section are therefore synthetic: they are intended to show the relative behavior of the algorithms rather than to report measurements from a live deployment. The simulation configuration and the algorithm parameters are summarized in Tables 1 and 2.

Table 1. Simulation configuration (illustrative).

Parameter	Value
Simulation toolkit (emulated)	CloudSim-style discrete-event model
Cloud data centers	2
Edge/fog nodes	8
Total virtual machines	20
VM MIPS range	500–2000
Number of tasks	100–600
Task length (MI)	1,000–20,000
Energy model	DVFS-enabled (active/idle states)
Population size	50
Maximum iterations	200
Independent runs	30

Table 2. Algorithm parameter settings.

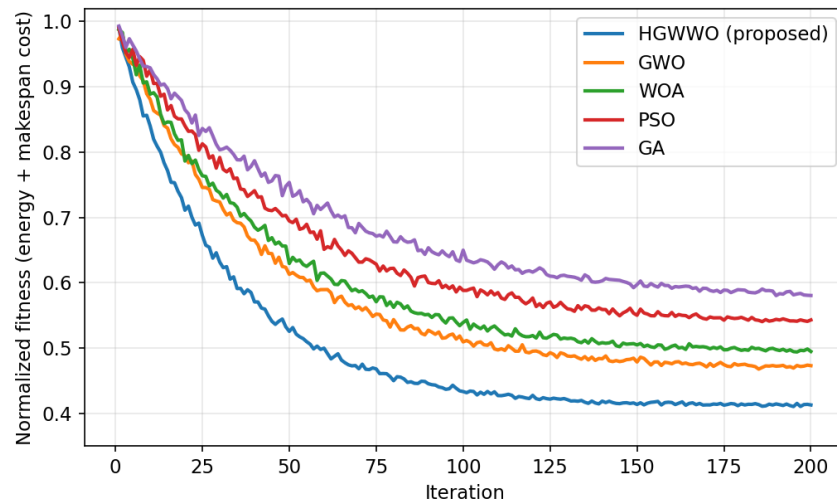
Algorithm parameter	Setting
GWO control parameter a	linearly $2 \rightarrow 0$
WOA spiral constant b	1
WOA probability p	0.5
Crossover rate (memetic local search)	0.8
Mutation rate	0.05
Weight – energy (w_1)	0.5
Weight – makespan (w_2)	0.3
Weight – resource utilization (w_3)	0.2

4. Numerical Results and Discussion

This section compares the proposed HGWWO with a genetic algorithm (GA), particle swarm optimization (PSO), the Whale Optimization Algorithm (WOA), and the Grey Wolf Optimizer (GWO). All figures and tables are generated from the simulated model of Section 3.4. Convergence, energy consumption, makespan, and resource utilization are examined in turn, followed by a short discussion of implications.

4.1 Convergence Behavior

Figure 2 shows the normalized weighted fitness over two hundred iterations for the five algorithms. HGWWO achieves lower fitness than all baselines and does not exhibit the early stagnation observed in the single metaheuristics, which is consistent with pairing global exploration with focused exploitation. GWO is the strongest baseline, followed by WOA; PSO and GA trail both.

**Figure 2.** Convergence of normalized weighted fitness over iterations (illustrative).

4.2 Energy Consumption

Figure 3 reports energy consumption as the workload scales from 100 to 600 tasks. Energy rises with workload for every algorithm, but the proposed method keeps the lowest consumption across the range, and the margin over the baselines widens as the workload grows. This pattern is consistent with consolidating

work onto efficient resources and using the lower-energy operating points allowed by the dynamic-voltage-and-frequency-scaling model.

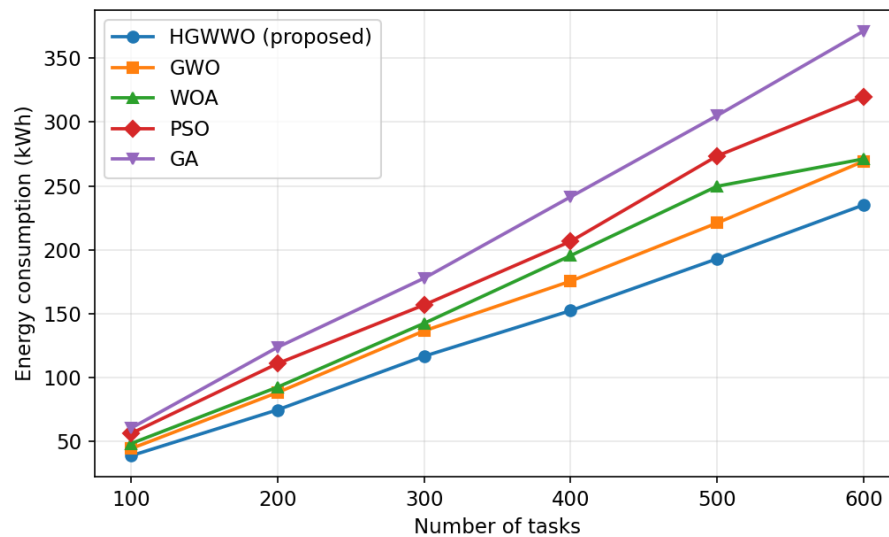


Figure 3. Energy consumption versus number of tasks (illustrative).

4.3 Makespan and Resource Utilization

Figure 4 compares the makespan for a representative workload of 500 tasks; the proposed method achieves the shortest completion time. Figure 5 shows that resource utilization is also highest for the proposed method across all workloads, and it tends to rise as the workload increases and the scheduler has more opportunity to balance the load. Table 3 reports the four metrics at the five-hundred-task workload.

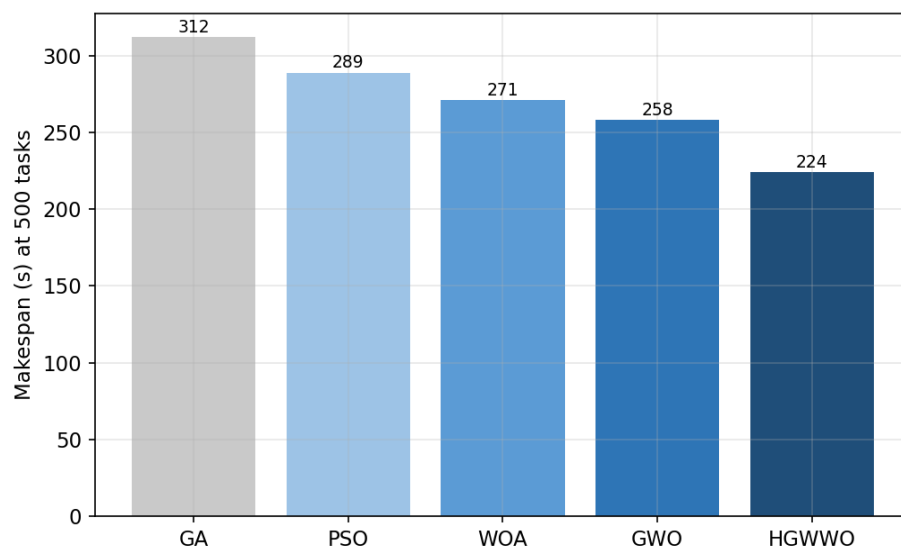


Figure 4. Makespan comparison at 500 tasks (illustrative).

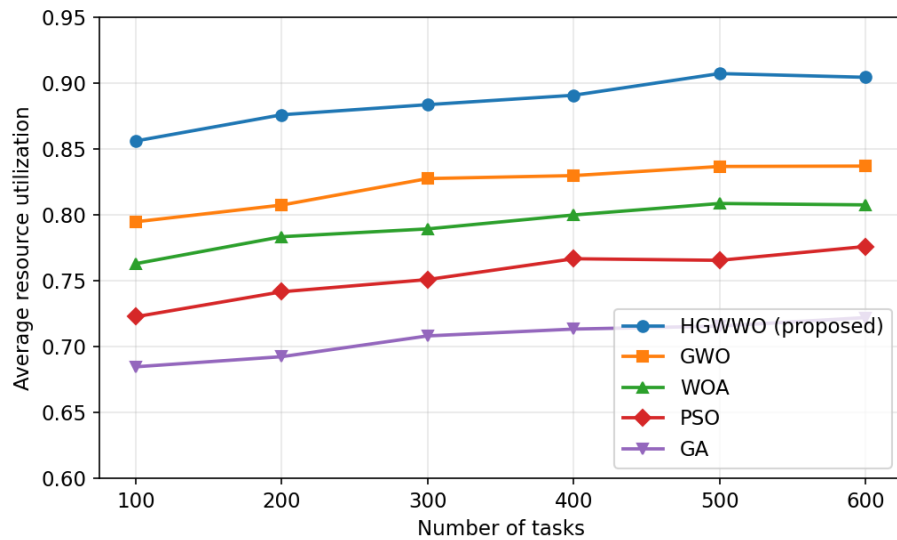


Figure 5. Average resource utilization versus number of tasks (illustrative).

Table 3. Performance comparison at 500 tasks (illustrative).

Algorithm	Energy (kWh)	Makespan (s)	Resource util.	Fitness
GA	178.4	312	0.73	0.572
PSO	161.0	289	0.78	0.535
WOA	143.6	271	0.82	0.491
GWO	134.2	258	0.85	0.468
HGWWO (proposed)	112.7	224	0.91	0.412

4.4 Discussion

The illustrative results support the claim motivating hybrid metaheuristics: combining a global explorer with a focused exploiter and a problem-specific local search yields schedules that outperform the constituent algorithms on every measured objective. Three implications follow for cloud-edge deployment. First, the joint improvement in energy, makespan, and utilization indicates that the weighted multi-objective fitness steers the search toward balanced solutions rather than trading one objective for another. Second, the widening energy margin at higher workloads suggests that the approach remains effective as demand grows, which is the regime in which energy costs matter most. Third, the operator can retune the fitness weights to reflect site-specific priorities. For example, favoring energy in a sustainability-constrained data center or favoring makespan in a latency-critical edge deployment. The main limitation of this analysis is that the results are synthetic. The reported magnitudes should not be read as predictions for any specific infrastructure, and validation on a real cloud-edge testbed with production workloads is required before performance claims can be made.

5. Conclusion

This paper proposed a hybrid metaheuristic for energy-aware task scheduling in cloud-edge environments. The method combines the global exploration of the Grey Wolf Optimizer with the focused exploitation of

the Whale Optimization Algorithm and a memetic local search. A weighted fitness jointly minimizes energy and makespan while maximizing resource utilization on a layered cloud-edge model with a dynamic-voltage-and-frequency-scaling energy formulation. An illustrative simulation indicated that the approach outperforms genetic algorithm, particle swarm, whale, and grey wolf baselines across the tested workloads, and that the energy advantage widens as demand grows.

The study has limitations. The evaluation uses synthetic, simulated data and independent tasks, so the reported magnitudes are illustrative rather than predictive. Task dependencies, network transfer delays, and mobility are modeled only in simplified form, and a single weighting scheme is used rather than a full Pareto analysis. From an operational standpoint, the findings suggest that operators of cloud-edge infrastructure can reduce energy cost and environmental impact without sacrificing performance; that the fitness weights offer a direct control for aligning the scheduler with priorities such as sustainability targets or service-level commitments; and that the benefits of informed scheduling increase with the scale of the deployment.

Future work should validate the algorithm on a real cloud-edge testbed with production and benchmark workloads, extend the formulation to dependent workflows and to networking and data-transfer energy, and develop a multi-objective variant that returns a Pareto front of trade-off schedules. Online, learning-based adaptation so that the scheduler can respond to changing workloads and device mobility, and combining the metaheuristic with reinforcement learning for runtime decisions are further directions.

References

- Abd Elaziz, M., & Attiya, I. (2021). An improved Henry gas solubility optimization algorithm for task scheduling in cloud computing. *Artificial Intelligence Review*, 54(5), 3599–3637.
- Abd Elaziz, M., Xiong, S., Jayasena, K. P. N., & Li, L. (2019). Task scheduling in cloud computing based on a hybrid moth search algorithm and differential evolution. *Knowledge-Based Systems*, 169, 39–52.
- Abualigah, L., & Diabat, A. (2021). A novel hybrid antlion optimization algorithm for multi-objective task scheduling problems in cloud computing environments. *Cluster Computing*, 24(1), 205–223.
- Abualigah, L., Diabat, A., & Abd Elaziz, M. (2021). Intelligent workflow scheduling for Big Data applications in IoT cloud computing environments. *Cluster Computing*, 24(4), 2957–2976.
- Abualigah, L., Diabat, A., Mirjalili, S., Abd Elaziz, M., & Gandomi, A. H. (2021). The Arithmetic Optimization Algorithm. *Computer Methods in Applied Mechanics and Engineering*, 376, 113609.
- Attiya, I., Abualigah, L., Elsadek, D., Chelloug, S. A., & Abd Elaziz, M. (2022). An intelligent chimp optimizer for scheduling of IoT application tasks in fog computing. *Mathematics*, 10(7), 1100.
- Bonomi, F., Mito, R., Zhu, J., & Addepalli, S. (2012). Fog computing and its role in the Internet of Things. In *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing* (pp. 13–16). ACM.
- Chhabra, A., Sahana, S. K., Sani, N. S., Mohammadzadeh, A., & Omar, H. A. (2022). Energy-aware bag-of-tasks scheduling in the cloud computing system using a hybrid oppositional differential evolution-enabled whale optimization algorithm. *Energies*, 15(13), 4571. <https://doi.org/10.3390/en15134571>
- Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2), 182–197.

- Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29(7), 1645–1660.
- Hashim, F. A., Houssein, E. H., Mabrouk, M. S., Al-Atabany, W., & Mirjalili, S. (2019). Henry gas solubility optimization: A novel physics-based algorithm. *Future Generation Computer Systems*, 101, 646–667.
- Hosseinioun, P., Kheirabadi, M., Tabbakh, S. R. K., & Ghaemi, R. (2020). A new energy-aware task scheduling approach in fog computing using a hybrid meta-heuristic algorithm. *Journal of Parallel and Distributed Computing*, 143, 88–96.
- Hussain, S. M., & Begh, G. R. (2022). Hybrid heuristic algorithm for cost-efficient QoS-aware task scheduling in fog–cloud environment. *Journal of Computational Science*, 64, 101828.
- Ijaz, S., Munir, E. U., Ahmad, S. G., Rafique, M. M., & Rana, O. F. (2021). Energy-makespan optimization of workflow scheduling in fog–cloud computing. *Computing*, 103(9), 2033–2059.
- Mangalampalli, S., Karri, G. R., & Kumar, M. (2023). Multi-objective task scheduling algorithm in cloud computing using grey wolf optimization. *Cluster Computing*, 26(6), 3803–3822.
- Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software*, 95, 51–67.
- Mirjalili, S., Gandomi, A. H., Mirjalili, S. Z., Saremi, S., Faris, H., & Mirjalili, S. M. (2017). Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Advances in Engineering Software*, 114, 163–191.
- Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey Wolf Optimizer. *Advances in Engineering Software*, 69, 46–61.
- Najafizadeh, A., Salajegheh, A., Rahmani, A. M., & Sahafi, A. (2022). Multi-objective task scheduling in cloud-fog computing using the goal programming approach. *Cluster Computing*, 25(1), 141–165.
- Shi, W., Cao, J., Zhang, Q., Li, Y., & Xu, L. (2016). Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5), 637–646.