Wireless Personal Communications
https://doi.org/10.1007/s11277-024-11421-8

# An Efficient Workflow Scheduling in Cloud–Fog Computing Environment Using a Hybrid Particle Whale Optimization Algorithm

Sumit Bansal[1] · Himanshu Aggarwal[1]

Accepted: 19 June 2024

**Abstract**
In the context of workflows, scheduling involves the systematic arrangement of tasks, allocation of resources, and coordination of dependencies to achieve desired outcomes. Effective scheduling is paramount in cloud–fog computing environments to harness the full potential of distributed resources while meeting application requirements. However, finding the most efficient solution often involves the quest for an optimal algorithm. An optimal algorithm is one that produces the best possible outcome in terms of a specified objective. While metaheuristic algorithms have been proposed by several researchers to address this issue, the solutions often converge at the local level, limiting their effectiveness. This study proposes the hybrid particle whale optimization algorithm as a solution for workflow scheduling problems. It is a combination of particle swarm optimization (PSO) and the whale optimization algorithm (WOA). By integrating the features of both algorithms, this algorithm tries to avoid becoming trapped in local optima. This algorithm aims to minimize the total execution cost, makespan, and energy consumption of tasks in a cloud–fog computing environment. The performance of the proposed algorithm is compared with that of the round robin, ant colony optimization, PSO, and WOA algorithms using five different scientific workflows, namely, the Inspiral, Montage, Epigenomics, Cybershake, and Sipht algorithms, which deploy various numbers of tasks and VMs. The proposed algorithm outperformed the other existing scheduling strategies. Moreover, the approach provides users with the flexibility to customize the scheduling strategy based on their preferences. Overall, this paper presents a novel approach for workflow scheduling in cloud–fog computing environments that improves the performance and efficiency of applications while also providing users with greater control over the scheduling process.

**Keywords** Optimization · Hybridization · Particle swarm optimization · WOA · Workflow

✉ Sumit Bansal
pup.sumit@gmail.com

[1] Department of Computer Science and Engineering, Punjabi University, Patiala, India

## 1 Introduction

Cloud–fog computing is an emerging paradigm that integrates the capabilities of cloud computing and fog computing. It allows for efficient resource utilization and provides ubiquitous services to end-users [1]. In cloud–fog computing environments, workflow scheduling plays a critical role in optimizing resource utilization and improving overall system performance [2]. A workflow is a sequence of tasks that are performed to achieve a specific goal. In a workflow, tasks are interdependent, and successful completion relies on the data input generated by the previous task(s) [3]. A directed acyclic graph (DAG) [4] is commonly used to represent the entire workflow, with nodes representing tasks and edges representing intertask dependencies. This structure ensures that the workflow is executed in the correct order, with no conflicting instructions or circular dependencies. Workflow scheduling is an important aspect of cloud–fog computing because it involves mapping dependent tasks to available resources while considering quality of service (QoS) parameters [5]. Workflow scheduling aims to optimize the efficiency and effectiveness of workflows by minimizing the idle time of resources, reducing the total completion time, and avoiding resource conflicts or bottlenecks. There are two types of workflow scheduling: static and dynamic. Static schedulers operate during compile time, deciding on scheduling algorithms based on prejob information without interference until the application's completion. In contrast, dynamic schedulers utilize updated information during job execution and observe system behavior to schedule subsequent jobs [6]. The task of scheduling is frequently addressed through the application of suboptimal approaches, including heuristics, metaheuristics, combinatorial optimization methods, and approximation algorithms [7]. One of the meta-heusritic algorithms, the genetic algorithm (GA), has been frequently utilized in workflow scheduling research to minimize makespan due to its robustness and ability to generate high-quality searches in polynomial time, although additional time may be needed to find a solution [8]. Pandey et al.[9] conducted a study that employed PSO [10] to schedule workflow applications in a cloud computing environment. PSO is a speedy optimization algorithm that may converge prematurely and become trapped in a locally optimal solution. On the other hand, the WOA [11] has been introduced as a metaheuristic algorithm that imitates the social behavior of humpback whales. [12] Although the WOA has a slow convergence speed, it can overcome the limitation of premature convergence, thereby reducing the likelihood of becoming trapped in local optima while maintaining high accuracy. The above study showed that a single scheduling algorithm is not sufficient for addressing these scheduling issues, such as local trapping, easy optimization solutions, low convergence speeds, and low time consumption. To overcome these issues, in the present study, we propose a new metaheuristic algorithm (HPWOA) that is constructed by the hybridization of two existing algorithms, PSO and the WOA. A hybrid algorithm combines two or more algorithms to achieve better performance and overcome the limitations of individual algorithms. The hybrid algorithm attempts to minimise each algorithm's drawbacks while maximising its advantages. The proposed algorithm combines PSO and the WOA to leverage the strengths of both algorithms. PSO can generate a set of feasible solutions, while the WOA can refine the solutions by searching for better solutions. Through its ability to avoid local optima and premature convergence, this method also enhances the quality of solutions. The proposed HPWOA algorithm is evaluated against existing algorithms, including RR, ACO, PSO, and the WOA, using three metrics: TEC, makespan and EC. The proposed algorithm is assessed on five distinct scientific workflows (Sipht, Inspiral, Cybershake, Montage and Epigenomics) with varying numbers of tasks. According to the simulation results, the proposed algorithm exhibited better performance than did the existing algorithms.

### 1.1 Existing Work

Numerous hybrid and metaheuristic techniques have been devised to address workflow-related problems. In this context, our primary focus is on discussing existing metaheuristic algorithms that are employed in cloud–fog computing [13] and are combined with either PSO or the WOA. In 2018, Meena et al. [14] proposed a metaheuristic workflow algorithm. The algorithm's primary goal is to optimize the mapping between the number of offloaded tasks using the MIN-MIN algorithm, thereby reducing costs in mobile and cloud computing environments. The use of workflow simulation can enhance workflow management effectiveness and scale to handle complex computational activities. Sahraei et al. [15] aimed to achieve efficient job management in cloud computing with reduced time and cost and increased utilization. They compared two scheduling methods, namely, MAX–MIN and MIN-MIN, with minimal cost scheduling, modified scenarios, and unmodified scenarios using workflow simulation.

In a study conducted by Tawfeek et al. [16], the ACO approach was used for task scheduling to minimize makespan. The study showed that ACO outperformed the FCFS and RR scheduling techniques. Similarly, Dasgupta et al. [17] proposed a genetic algorithm to address the task schedule ing problem, and the experimental results demonstrated that this algorithm outperformed RR, FCFS and a local search algorithm called stochastic hill climbing in terms of makespan. However, it was reported that the GA approach was time-consuming for finding optimal results [18]. On the other hand, ACO is a highly complex algorithm and requires a considerable amount of time to obtain optimal results [19]. Table 1 below illustrates a comparative analysis between existing studies and the proposed work, focusing on the algorithms employed, workflows utilized, objectives, strengths, limitations and open challenges.

### 1.2 Motivation

The emergence of fog computing as a new approach to edge computing has sparked a surge of interest in exploring workflow task scheduling within cloud–fog environments. Prior studies have focused primarily on evaluating algorithm performance based on cloud computing, using only a few scientific workflows and a limited number of tasks, and comparing outcomes with a small selection of existing algorithms. The motivation behind this research is to achieve a more comprehensive understanding of the most appropriate algorithm across all parameters. A new scheduling algorithm needs to be developed to conduct a more in-depth investigation. The new algorithms must be assessed against a wider range of workflows, with different task numbers, and compared with more similar algorithms to obtain more accurate and reliable results.

### 1.3 Research Contribution

The present study has made significant contributions to related research in the following ways:

**Table 1** A comparative table of existing studies with the proposed work

| References | Optimization Technique | Work-flow used | Objective function | Strength(s) | Limitations and open challenges |
|---|---|---|---|---|---|
| Mikram et al. [20] | Hybrid HEPGA algorithm | 0 | Makespan | Better in terms of job optimisation and fitness metrics | Focused exclusively on reducing makespan without integrating workflow or statistical factors |
| Ayoubi et al.[21] | Methodology used for Autonomous Placement of IoT Services | 0 | Workload | Better than MOPSO and NSGII algorithms | No workflow used and focus only on a single objective, i.e., workfload |
| Arora and Banyal [22] | Combination of PSO and GWO algorithm | 4 | Cost and Time | Better in TEC than in independent PSO and WOA | Didn't consider the time consuming |
| Li et al. [23] | PSO and Lion Optimization Algorithm | 3 | Makespan | Better than PSO, LOA, ACO, GA and NBWS | The distribution of workload distribution among resources is not considered |
| Baker et al. [24] | IoT critical infrastructure employing integrity, security, and privacy (SCADA) infrastructure | 0 | security | A secure and time saving platform designed for fog based systems | Only focus of the work on security and no workflow used |
| Unhelkar et al. [25] | RFID technology | 0 | Survey of RFID in supply chain | Discusses the role of RFID in SCM | Focused on time and cost. No workflows are used for comparison |
| Kaiwartya et al. [26] | A-NSGA Algorithm | 0 | Facult Tolerance and communication delay | Optimize fault tolerance in WSNs | Framework focused on fault tolerance for virtualization. No workflows are used for comparison |
| Ghobaei et al. [27] | Propose a moth-flame optimization (MFO) algorithm | 2 | Response time and cost | Better in high speed convergence | No workflow used for the comparison of results |
| Bothra et al. [28] | Cost-effective Hybrid Genetic Algorithm (CHGA) | 4 | Cost | Improve cost compared than ACO, PSO, CEGA, CLGA | Didn't consider TET and EC |
| Wada et al. [29] | A multiobjective GA and queuing theory is propsoed | 0 | Cost and Latency | Better in low latency and high convergence speed | The accuracy of the algorithm is low and no workflow used |

**Table 1** (continued)

| References | Optimization Technique | Workflow used | Objective function | Strength(s) | Limitations and open challenges |
|---|---|---|---|---|---|
| Dastjerdi et al. [30] | An Ontology based work | 0 | Cost and latency | Better in scalabilty and high in efficiency | No workflow is used for results comparison and fog nodes |
| Kaur et al. [31] | A deep-Qlearning-based heterogeneous earliest-finish-time (DQ-HEFT) algorithm | 4 | Cost and makespan | Better in makespan and speed metrics compared to CPOP, HEFT_U, QL-HEFT algorithms | Neglecting fog node considerations |
| Our approach | Hybrid particle whale optimization algorithm (HPWOA) | 5 | Makespan, cost and energy | Better in achievingoptimal solution and objective functions | Assessing results across various hybrid algorithms and high computational parameters |

- An analysis of workflow scheduling issues pertaining to cloud platforms was conducted.
- A new metaheuristic algorithm called the HPWOA was developed that addresses the challenges associated with workflow scheduling. This algorithm is a hybridization of two efficient algorithms, WOA and the PSO, leveraging their respective strengths to converge to an optimized solution.
- The proposed algorithm was evaluated using WorkflowSim, and the results were compared with those of four existing algorithms, namely, RR, ACO, PSO, and the WOA. The evaluation was conducted using scientific workflows at varying task numbers and VMs. The outcomes were measured in terms of total execution cost, makespan, and energy consumption and were found to demonstrate significant improvements over the existing algorithms. Moreover, statistical tests were applied to examine the results.

## 1.4 The Paper's Structure and Road Plan

This paper is structured as follows: Sect. 2 provides an overview of the application model and PSO algorithm, including its components such as pbest and gbest values, the WOA, particle velocity, the fitness function, makespan, TEC and EC. Section 3, titled "Components of the proposed HPWOA", describes the proposed workflow used to address the problem. In the "Simulation setup and dataset for performance evaluation" section, we present the simulation environment and datasets used for our experimentation and comparisons. Section 5, titled "Simulation results and outcomes", describes the results comparison and outcomes of the proposed algorithm with those of existing algorithms. Finally, we summarize our findings and provide recommendations for future research in the "Conclusion and future work" section. Figure 1 helps readers acquire a thorough understanding of the research paper. To fully grasp the novelty of the proposed work, readers are advised to focus on Sects. 1, 2, 3, 5, and 6. Section 3 provides a comprehensive guide to writing code for building a workflow application. The paper's significant discoveries and conclusions are discussed in Sects. 1, 2, 4 and 6. Finally, Sects. 4 and 6 present the results of the performance evaluation of the paper.

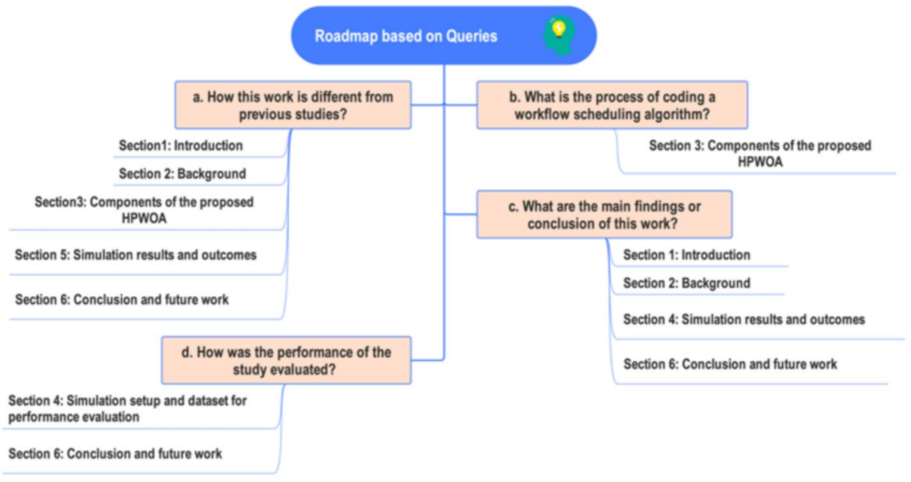

Roadmap based on Queries
a. How this work is different from previous studies?
Section1: Introduction
Section 2: Background
Section3: Components of the proposed HPWOA
Section 5: Simulation results and outcomes
Section 6: Conclusion and future work
b. What is the process of coding a workflow scheduling algorithm?
Section 3: Components of the proposed HPWOA
c. What are the main findings or conclusion of this work?
Section 1: Introduction
Section 2: Background
Section 4: Simulation results and outcomes
Section 6: Conclusion and future work
d. How was the performance of the study evaluated?
Section 4: Simulation setup and dataset for performance evaluation
Section 6: Conclusion and future work

**Fig. 1** Roadmap of the study

## 2 Background

In this section, the system model and the background of the well-known algorithms, PSO and the WOA, are explored. Furthermore, a comprehensive discussion is presented on the reason for combining the algorithms, fitness function, TEC, makespan, and EC associated with these algorithms.

### 2.1 System Model

The cloud–fog system model involves user devices ($T_1$, $T_2$, $T_3$…, $T_n$) generating tasks, each with a unique task number. After that, these tasks are sent to the user application, which subsequently dispatches them to a scheduler using sink nodes. The scheduler is an essential component of task scheduling since it sets user QoS policies and parameters. After this, tasks are routed to different workflows and then put on fog devices to be completed. The fog layer comprises fog nodes ($F_1$, $F_2$, $F_3$…, $F_n$) equipped with computers, mini-servers, and smart gateways, forming an intermediate fog layer. Every fog node performs distinct roles such as a smart device with limited computation, storage, and internet access. When a fog node ($F_g$) comes across data that can't be processed locally, it sends the information to a remote cloud so that it can be processed and analysed further. Fog cluster managers do this job by combining a cloud–fog manager and a task scheduler.

The top cloud layer is composed of high-performance servers with sufficient processing and storage capacity, as shown in Fig. 2. Every server has several virtual machines ($VM_1$, $VM_2$…, $VM_k$), each defined by specific memory and processing speed characteristics. Depending on the processing request, data can be transferred from the fog to the cloud or vice versa. The suggested task-scheduling technique is integrated into the cloud–fog manager to enable effective job execution. Better job distribution and execution are guaranteed by this approach for the system's fog and cloud nodes.

The cloud–fog manager makes decisions based on the task-scheduling strategy, determining whether tasks will be executed in the fog or cloud. After task completion, the results of execution are transmitted back to the cloud and fog manager. This manager strengthens all the results before forwarding them to the IoT system.

### 2.2 Particle Swarm Optimization (PSO)

PSO is a metaheuristic algorithm originally presented by Kennedy and Eberhart [32]. The popularity of this nature-inspired technique is increasing due to its adaptability and ease of execution. PSO has received immediate attention from researchers in all fields. The algorithm simulates the behaviour of a flock of birds flying together in multidimensional space in search of an optimal location, modifying their motions and distances for a better search. Particle swarms are generated at random, and the optimal method is searched for by updating generations. Each particle in PSO indicates a solution. In the creation process of each particle, the algorithm establishes the personal best particle as 'pbest' and the global best particle as 'gbest'. The fitness value of the fitness functions mentioned in Sect. 2.4 determines the selection of the pbest and gbest values. Based on the fitness value, the 'pbest' particle is the local best particle identified thus far. The gbest particle is the best particle found from the entire population based on the fitness value. In every iteration of the PSO algorithm, the position and velocity of each particle are updated. Equation (1) below is used to calculate the position and velocity updations.

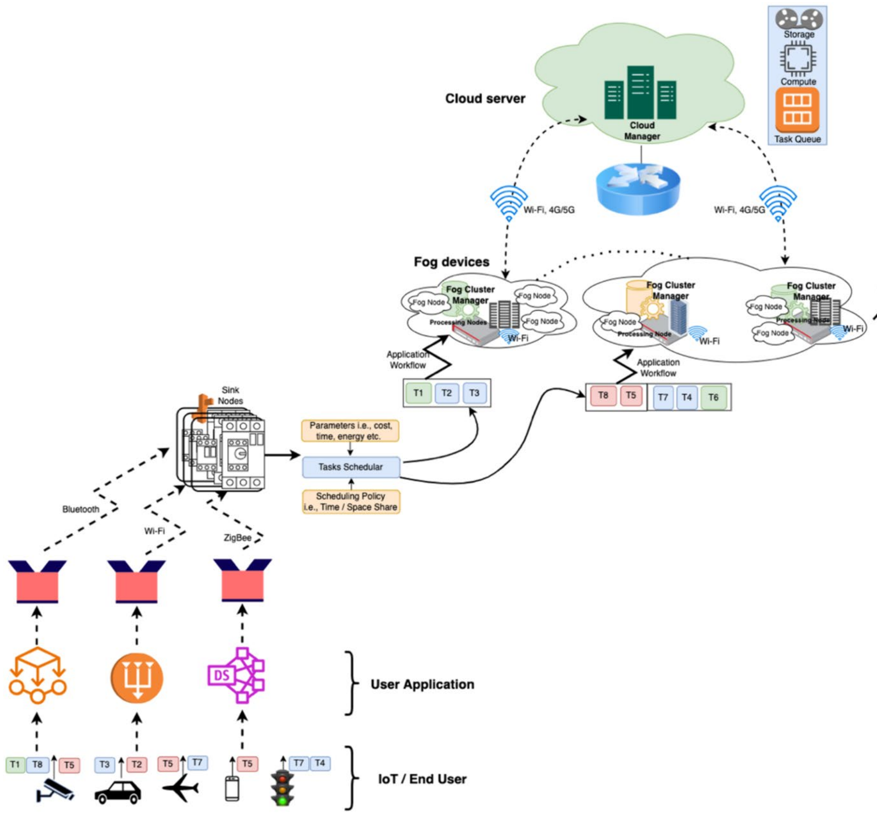


**Fig. 2** System model of the cloud–fog environment

$$V_i(t+1) = w * V_i(t) + C_1.r_1 * \left(pbest - x_i(t)\right) + C_2.r_2 * \left(gbest - x_i(t)\right) \tag{1}$$

The particles are shifted toward the optimal location in each iteration. Both the personal best (pbest) and global best (gbest) particles influence the velocity and position of the particle.

## 2.3 WOA

The whale optimization algorithm is a nature-inspired metaheuristic optimization algorithm developed by Mirjalili and Lewis [11]. It imitates the hunting behavior of humpback whales and is based on the bubble-net-looking strategy. Humpback whales are known to be among the most intelligent species of the whale family; they possess brain cells called spindle cells, which are similar to those of humans. The WOA leverages this behavior and the distinct bubble-net feeding strategy of humpback whales, which can be mathematically modelled and optimized. To begin the optimization process, a random population of 'n' whales is generated in the search space. Next, the WOA evaluates the fitness of the objective function for each potential solution in the population and identifies the upgraded solution. The algorithm comprises three distinct stages: (1) encircling the prey, (2) utilizing bubble-net hunting techniques to capture the prey during the exploitation phase, and (3) actively exploring the search space to locate prey during the exploration phase.

$$Q^{k+1} = Q^{k}_{best} - \left(2a * \rho_1 - a\right) * \left|2\rho_2 * Q^{k}_{best} - Q^{k}\right| \tag{2}$$

$$Q^{k+1} = Q^{k}_{rand} - A * \left|C * Q^{k}_{rand} - Q^{k}\right| \tag{3}$$

The position of the whale population is updated by using Eq. (2), and for individual whales, the position vector Eq. (3) is used. Here, $Q_{rand}$ is the random position vector chosen from the current population, $Q_{best}$ is the position vector of the best solution obtained thus far, and '$\rho$' is a random vector between [0,1].

### 2.4 Reasons Behind the Combination of Both Algorithms

The hybridization of PSO and the WOA stems from the need to overcome the inherent limitations present in each algorithm when used independently. Particle Swarm Optimization, known for its effectiveness in optimisation tasks, often lacks diversity in search and is prone to becoming stuck in global optima. Conversely, the WOA boasts exploration capabilities but suffers from reduced solution accuracy, slow convergence and a inclination to become trapped in local optima. The hybridization of Particle Swarm and the Whale Optimization Algorithm aims to harness the strengths of both algorithms while mitigating their weaknesses. By merging these approaches, we create a more efficient and robust optimization tool that benefits from the search capabilities of PSO and the exploratory nature of the WOA. This integration leads to improved implementation and efficacy in tackling difficult optimisation challenges across various domains, involving cloud–fog.

### 2.5 Fitness Function

The fitness function of the proposed algorithm outlines the goals that need to be optimized. There are two methods for creating multiobjective fitness functions, namely, priori and a posteriori. The a priori method involves assigning a weight to each objective based on its level of importance, which leads to the creation of a single-valued function. This function is commonly referred to as a fitness value. On the other hand, the a posteriori method identifies the set of nondominant options. For the proposed scheduling algorithm, we used the a priori method to create the fitness function, where makespan, TEC and EC are the three objectives that compose the fitness function [33]. Equation (4) can be used to mathematically define the fitness function under evaluation.

$$f(TET, TEC) = \alpha_1 \times TET + \alpha_2 \times TEC + \alpha_3 \times EC \tag{4}$$

The abbreviations TET, TEC and EC represent the total execution time (referring to the makespan time), total execution cost and energy consumption, respectively. The weights allocated to each objective are $\alpha_1$, $\alpha_2$, and $\alpha_3$. Here, we take into consideration the 0.5 weight that is equivalent to $\alpha_1, \alpha_2 and \alpha_3$. The following subsections provide a detailed explanation of makespan, TEC and EC.

### 2.5.1 Makespan Time

The workflow's maximum completion time for tasks is the total execution time (makespan). Put differently, makespan refers to the duration required to finish all the assigned tasks across different VMs [34]. Mathematically, Eq. (5) can be used to determine the workflow makespan.

$$TET_w = \max\{CT_i | i = 1,2,3,4\ldots..m\} \tag{5}$$

where 'CT$_i$' is the processing task T$_i$'s completion time. The time of all task execution is the duration of the completion time. In scenarios where tasks rely on one another, the duration for which predecessor tasks must wait is taken into account. The completion time for task 'i' is presented in Eq. (6) as 'CT$_i$'.

$$CT_i = \begin{cases} ET_i & \mathit{iff}\ pred(T_i) = \emptyset \\ WK_i + ET_i & \mathit{iff}\ pred(T_i) \neq \emptyset \end{cases} \tag{6}$$

According to Eq. (7), the wait time for task 'T$_i$' is equal to the total time required to complete all of the tasks that came before it in the process.

$$WK_i = \begin{cases} 0 & \mathit{iff}\ pred(T_i) = \emptyset \\ \max(CT_i) & \mathit{iff}\ pred(T_i) \neq \emptyset \end{cases} \tag{7}$$

$$ET_{i,j} = \frac{SZ_{Task}}{Num(PE_j) * PE_{Unit}} \tag{8}$$

The execution time of task 'T$_i$' on virtual machine 'VM$_j$' can be determined using Eq. (8), where 'SZ$_{Task}$' denotes the task size in million instructions (MI) and 'PE$_{Unit}$' represents the core size in MIPS. The variable Num (PEj) denotes the quantity of cores that have been assigned to the particular computing machine.

### 2.5.2 Total Execution Cost (TEC)

To minimize costs in cloud computing, the pay-as-you-go billing model is commonly used. This is because cloud service providers typically charge a fixed monthly fee based on the cloud services used, and costs are incurred for execution, communication, and storage. The overall cost of running a virtual machine (VM) is calculated based on the cost charged per unit interval for using the VM and the duration taken for the assigned tasks to be executed on the VM. Therefore, minimizing costs is a significant goal in cloud computing[35]. The TEC of workflow 'W' is represented mathematically in Eq. (9).

$$TEC_w = \sum_{i \in W, i=1}^{k} \frac{ET_{i,j}}{\tau} \times CO_j : j \in VM_j \tag{9}$$

where 'CO$_j$' is the price for a type-I Virtual Machine instance for a certain amount of time in the cloud data centre. The duration for which the user utilizes the resources is expressed. ET$_{i,j}$ is the task T$_i$'s type-j VM instance execution time.

### 2.5.3 Energy Consumption Model

In this section, our objective is to present a generic model for estimating the energy consumption of each previously mentioned cloud–fog-related infrastructure during a specified time period ‘T’ when the allocated VMs are active. It is important to note that we do not account for variations among these architectures in terms of quality of service (e.g., latency). Additionally, our estimation only covers the energy consumption of the infrastructure itself, encompassing the telecommunication network between data centers and users. To assess the consumption of energy of a resource, we compute the combined consumption of energy for both static and dynamic energy utilized by the processors. While static energy remains constant upon resource provisioning, dynamic energy is dependent on the processor frequency and voltage [36]. The main factor responsible for energy consumption in a cloud-data server is associated with the storage, memory, processors, and network components. The processor, in particular, accounts for 37% to 43% of the total energy usage [37]. However, the costs of storage, memory, and network components remain constant regardless of workflow execution. This paper focuses solely on the energy consumption of processors, which is strongly linked to workflow execution. In our analysis, we assume negligible overhead for frequency transitions, which occur over a very short period of time (for a period of microseconds) relative to other factors. Hence, the dynamic energy consumption is the primary factor responsible for the total EC of a processor.

This paper focuses on the dynamic EC, which is expressed in Eq. (10). The variables C, V and f represent the processor capacity constant, voltage, and frequency, respectively. Energy consumption can be divided into two categories: idle and busy. In the idle state, the processor operates at its minimum frequency and voltage levels. Equation (11) defines the overall energy consumption of a resource.

$$Energy_{dynamic} = C.V^2.f \tag{10}$$

$$Energy^{total} = (Energy_{r_j}^{idle} + Energy_{r_j}^{load}) + Energy_{r_j}^{send} + Energy_{r_j}^{rec} \tag{11}$$

If resource $r_j$ is occupied (i.e., busy in task performing), its energy usage is computed through Eq. (12), where $r_j$, $t_i$, and $f_{rj}$, $t_i$ denote the voltage and frequency of resource $r_j$ carrying out task $t_i$, respectively. When a resource is not in use, the processor operates with the least voltage and frequency settings.

$$Energy_{r_j}^{Total} = \sum_{t_i \in r_j} C.Energy_{r_{j,t_i}}^{load} + v^2.Energy_{r_{j,t_i}}^{send} + f.Energy_{r_{j,t_i}}^{rec} \tag{12}$$

## 3 Components of the Proposed HPWOA

This section presents the workflow, parameters, flowchart, pseudocode, fitness function, and performance metrics of the proposed HPWOA. The effectiveness and performance of the workflow scheduling scheme utilizing the HPWOA are evaluated and compared with those of other notable approaches documented in the literature.

### 3.1 The Proposed Algorithm Flow

A hybrid metaheuristic algorithm called the HPWOA is proposed that combines the PSO and the WOA algorithm. The effectiveness of the algorithm is evaluated and compared with that of other existing algorithms, such as RR, PSO, ACO and the WOA, using five scientific workflows, namely, Montage, Cybershake, Epigenomics, Sipht, and Inspiral, with varying task numbers and VMs. Montage is assessed as an example, with the evaluation spanning from 20 to 300 task numbers and 5–20 VMs. To evaluate its effectiveness, the algorithm's performance is assessed by minimizing the makespan, TEC, and EC during the workflow scheduling process. The performance of the HPWOA was evaluated by measuring its ability to minimize the makespan, TEC and EC, which are metrics for scheduling workflow tasks. To initialize the algorithm, certain parameters were set, and a population of particles, each representing a potential solution to the workflow issues, was generated randomly. The PSO algorithm was run for half of the total iteration count ($\frac{n}{2}$) using the randomly generated population, with the gbest outcome of these iterations used as a leader whale by the WOA algorithm. The remaining half of the iterations (from $\frac{n}{2} + 1$ to n) were conducted by employing the initialized leader whale through the Whale Optimization Algorithm. The outcomes generated from this process were subsequently used as the results of the prospective algorithm. The precision of the outcomes was ascertained based on the number of iterations, where each iteration brought the results closer to the optimal solution. Several task numbers were employed to perform evaluations on five scientific workflows, namely, Montage, Cybershake, Epigenomics, Sipht, and Inspiral.

After evaluating the results of the proposed algorithm, it was observed that its performance was better than that of existing algorithms. The main limitation of the standard PSO algorithm is that it can usually find only the best local solution when confronted with challenging and intricate NP-hard problems. In contrast, the HPWOA has shown superior performance compared to the RR, ACO, PSO and whale optimization techniques. To provide a visual representation of the algorithm, Fig. 3 shows the HPWOA flowchart, and Algorithm 1 provides a detailed description of the algorithm.

### 3.2 Reason for Using Fifteen Iterations in the Proposed Algorithm

To enhance the performance of the results, we executed the proposed approach for both fifteen and twenty-five iterations. However, after running for twenty-five iterations, only the initial fifteen iterations exhibited significant improvements. Subsequent rounds beyond fifteen showed minimal changes in values, rendering them negligible. The values for execution time, cost and energy obtained during the repeated runs nearly formed a constant graph [38]. This led us to conclude that the algorithm should be executed only fifteen times.

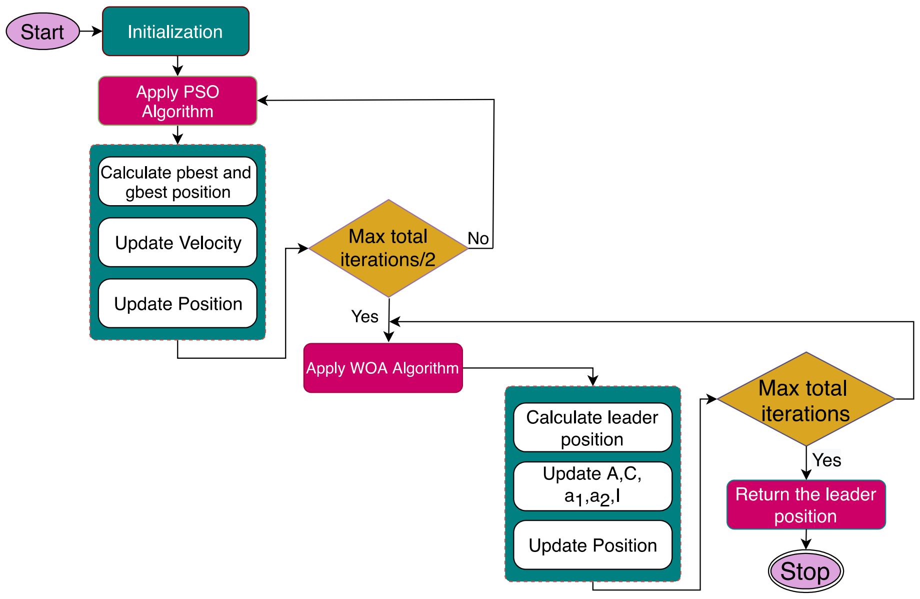


**Fig. 3** Proposed flowchart of the HPWOA

Algorithm 1. The Proposed HPWOA Algorithm

```
    Input: Workflow W(N,E) and set of resource(VM_1, VM_2, .... VM_j)
    Output: leaderpos
1   for i = 0 to populationsize do
2       | population ← randomize ()
3   end for
4   Set: i ← 0
5   while i < (maxiter/2) do
6       | calculate pbest and gbest position
7       | update velocity
8       | update position
9       | i ← i+1
10    end while
11  Set: leaderpos ← gbest
12    while I < maxiter do
13        | calculate leader position
14        | update A, C, a_1, a_2 and l
15        | update position
16        | i ← i+1
17    end while
```

The next subsection discusses the performance metrics and the initial parameters used for the proposed algorithm.

### 3.3 Performance Metrics

The application utilizes VMs to handle 30-2000 tasks for Cybershake and other workflows. Based on the experiments, the algorithm considers an equal number of particles and tasks, with an inertia weight of 0.1 and learning factors $c_1$ and $c_2$ set to 2. Table 2 lists the initial parameters employed for the proposed HPWOA.

## 4 Simulation Setup and Dataset for Performance Evaluation

This section covers the hardware components utilized in the simulation work, along with a discussion on the simulation setup proposed for testing. Furthermore, this review delves into the various scientific workflow applications that were employed in the Pegasus project.

### 4.1 Experimental Environment

The experimental evaluation was performed on a MacBook Air 2017 instrument equipped with MacOS Big Sur 11.5.1, which features 8 GB of RAM and a 1.8 GHz dual-core Intel Core i5 processor. The simulations were executed utilizing the WorkflowSim tool, which is an extended version of the CloudSim simulator. The workflow layer was simulated utilizing the WorkflowSim simulator, and different delay and failure levels were used to simulate real-world distributed environments. The task sequence was derived from the DAG task graph, and the final scheduling step utilized the selected scheduling algorithm. WorkflowSim can be augmented by integrating user algorithms to extend its capabilities. This experiment aimed to evaluate the effectiveness and efficiency of incorporating user algorithms into WorkflowSim for improved performance and productivity. Table 3 below illustrates the simulation parameters employed by the proposed algorithm.

### 4.2 Dataset

To conduct simulation experiments, we used five scientific workflows from various fields, each with different model structures and computational needs, as shown in Fig. 4. Montage is an astronomy application used to generate customized sky mosaics and has I/O-intensive tasks with low processing demands. LIGO [39] provides Inspiral workflows for detecting gravitational waves and has CPU-intensive tasks with large memory requirements. Epigenomics is a data processing pipeline for genome sequencing with CPU-intensive tasks. These workflows are crucial for researchers exploring diverse scientific domains. The Cybershake workflow [40] identifies ruptures, creates variations with different slip distributions and hypocenter locations, generates synthetic seismograms, extracts peak intensity measurements, and produces probabilistic seismic hazard curves. The epigenomic workflow [41] involves DNA extraction, sequencing, and mapping, followed by peak calling to identify regions of the genome with enriched signals and downstream analysis to investigate the biological relevance of the identified regions and their association with gene regulation and disease. The sipht workflow involves preprocessing raw imaging data, performing registration and segmentation to isolate individual cells, extracting morphological features to characterize cell phenotypes, and performing statistical analysis to identify and quantify phenotypic differences between experimental conditions or disease states. Juve

**Table 2** Parameters of the proposed HPWOA

| Name | Value | Explanation |
|---|---|---|
| Length of particle's | Number of tasks | Maximum number of Iterations for the solution exploration |
| Particle in number's | 25 | Size of the population of the search agent that is exploring for a solution by being updated by equations |
| Learning factor $c_1$ | 2 | Factor to control cognition of search agent |
| Inertia weight | 0.1 | Inertia makes the particle/whales move in the same direction and with the same velocity |
| Learning factor $c_2$ | 2 | Factor to control social Influence of population |
| Number of Iterations | 500 | Maximum Iteration Limit for Solution Exploration |
| Repeated experiment | 15 | Number of Iterations for Calculating Average Solution |

**Table 3** Assessment of the simulated values produced by the HPWOA algorithm

| Parameters | Value | |
|---|---|---|
| Number of tasks | 30–2000 | |
| Type of VMs | Cloud | Fog |
| VM (in numbers) | 5–20 | |
| MIPS | 1600 | 1300 |
| Cost | 0.96 | 0.48 |
| Random access Memory | 512 | |
| Processor used | 1 | |
| Policy used | Time shared/space shared | |

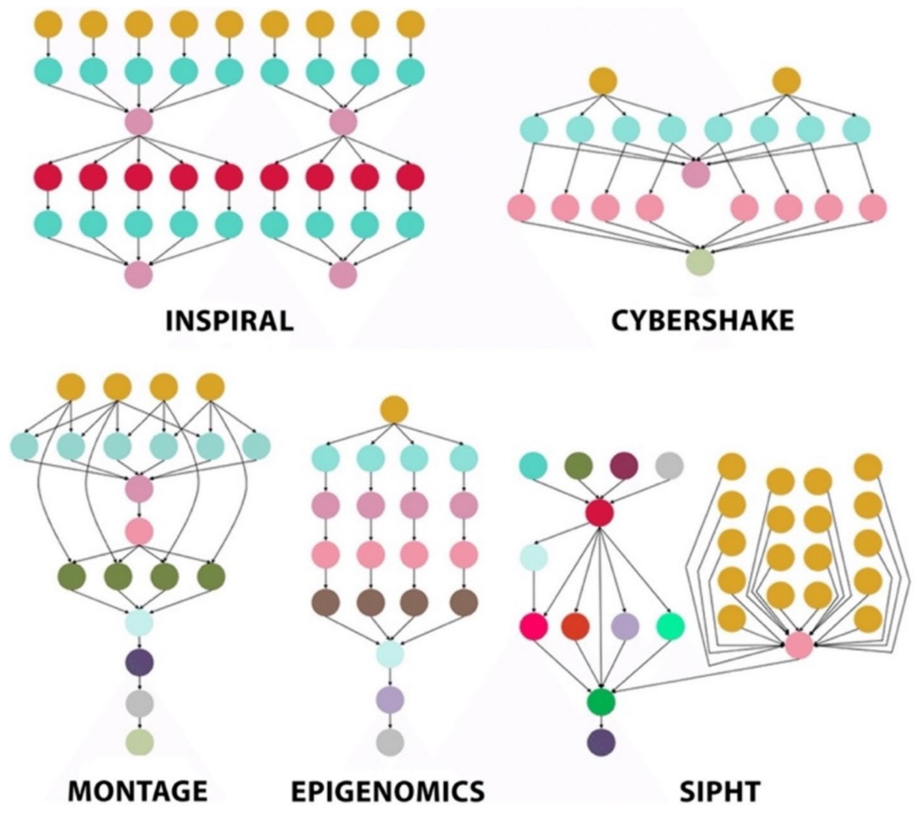


**Fig. 4** Five scientific workflow structures

et al. [42] provided a comprehensive description of these workflows. Four workflow scales of 30, 50, 100 and 1000 tasks were used in this paper for very small, small, medium, and large applications, respectively.

The next section contains the outcomes of the simulation and its results under different scientific workflows.

## 5 Simulation Results and Outcomes

To analyse the experimental outcomes of both the proposed and existing algorithms (RR, ACO, PSO, the WOA, and the HPWOA), simulation results were assessed on five different workflows, namely, Montage, Epigenomics, Cybershake, Inspiral and Sipht, with varying numbers of tasks and VMs. For instance, the Cybershake workflow was assessed for 30 to 1000 tasks, with a maximum limit of 500 iterations. For each scenario, 15 repeated experiments were performed, and the average value was calculated to determine the outcomes. To calculate the makespan, TEC and EC in this study, we used four algorithms: RR, ACO, PSO, and the WOA. These algorithms were compared with the performance of the prospective algorithm (HPWOA) with respect to time and cost objectives. However, due to a lack of support for the RR algorithm in Workflowsim 1.2.1, the results for energy consumption were compared only for the ACO, PSO, and WOA algorithms. The detailed outcomes of the workflow results for makespans, TECs and ECs at various tasks are shown in Tables 4, 5, and 6, respectively. The outstanding values in all the result tables are emphasized in bold.

### 5.1 Cybershake Workflow Performance Evaluation

Figure 5 shows the makespan simulation results for the RR, ACO, PSO, WOA and proposed algorithm (HPWOA). The x-axis and y-axis of the graph show the number of tasks and makespan, respectively.

**Table 4** Makespans of various scientific workflows

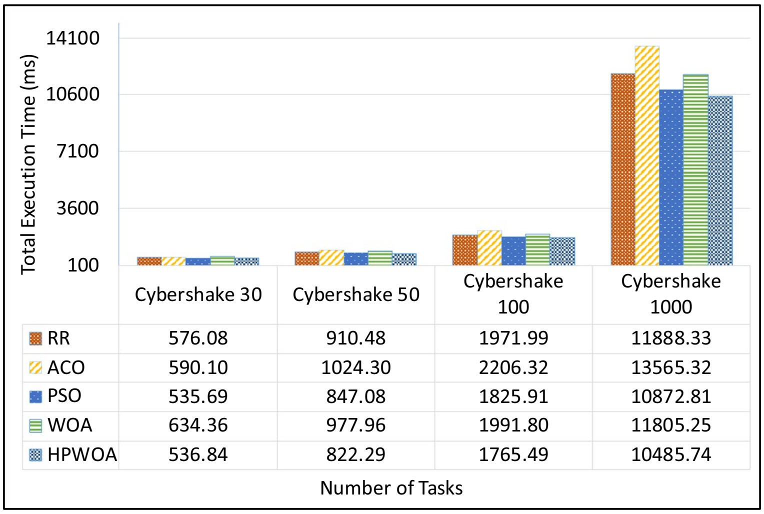
| Workflow | Tasks | VM | RR | ACO | PSO | WOA | HPWOA |
|---|---|---|---|---|---|---|---|
| Cybershake | 30 | 5 | 576.08 | 590.10 | **535.69** | 634.36 | 536.84 |
| | 50 | 10 | 910.48 | 1024.30 | 847.08 | 977.96 | **822.29** |
| | 100 | 15 | 1971.99 | 2206.32 | 1825.91 | 1991.80 | **1765.49** |
| | 1000 | 20 | 11,888.33 | 13,565.32 | 10,872.81 | 11,805.25 | **10,485.74** |
| Epigenomics | 24 | 5 | 1113.33 | 1079.58 | **1016.86** | 1236.17 | 1023.12 |
| | 47 | 10 | 1476.72 | 1413.11 | 1455.62 | 1597.14 | **1388.01** |
| | 100 | 15 | 2645.66 | 2613.53 | 2596.53 | 2814.75 | **2470.06** |
| | 997 | 20 | 17,219.07 | 17,235.01 | 16,638.39 | 17,790.34 | **15,775.93** |
| Inspiral | 30 | 5 | 2127.06 | 2019.65 | 1772.07 | 2129.13 | **1762.01** |
| | 50 | 10 | 3173.91 | 3342.68 | 2856.87 | 3366.73 | **2765.17** |
| | 100 | 15 | 5533.39 | 6128.65 | 4977.48 | 5329.84 | **4774.83** |
| | 1000 | 20 | 54,278.51 | 57,120.66 | 50,161.79 | 49,362.76 | **46,874.21** |
| Montage | 20 | 5 | 345.79 | **295.94** | 309.97 | 345.61 | 298.25 |
| | 60 | 10 | 398.20 | 383.36 | 391.74 | 426.74 | **369.01** |
| | 100 | 15 | 567.14 | 523.25 | 555.20 | 589.70 | **500.78** |
| | 300 | 20 | 3796.84 | 3468.36 | 3586.69 | 4099.34 | **3283.31** |
| Sipht | 29 | 5 | **4328.90** | 4956.34 | 4419.17 | 4706.40 | 4397.31 |
| | 58 | 10 | 5297.85 | 6182.60 | 5128.11 | 5422.33 | **4724.99** |
| | 97 | 15 | 7597.85 | 8052.32 | 7019.67 | 8085.84 | **6306.90** |
| | 968 | 20 | 62,327.05 | 70,273.66 | 59,654.49 | 72,569.22 | **52,899.26** |

**Table 5** TECs of various scientific workflows

| Workflow | Tasks | VM | RR | ACO | PSO | WOA | HPWOA |
|---|---|---|---|---|---|---|---|
| Cybershake | 30 | 5 | 816.67 | 814.52 | 652.96 | 837.79 | **649.28** |
| | 50 | 10 | 2320.53 | 2620.30 | 2179.45 | 2419.93 | **1951.91** |
| | 100 | 15 | 7341.46 | 6940.63 | 6792.92 | 7215.98 | **6667.15** |
| | 1000 | 20 | 55,217.39 | 57,497.58 | 53,920.77 | 54,772.91 | **50,992.50** |
| Epigenomics | 24 | 5 | 2008.61 | 2285.86 | 1902.43 | 2444.58 | **1782.44** |
| | 47 | 10 | 2831.66 | 2926.98 | 2613.97 | 3247.61 | **2368.81** |
| | 100 | 15 | 4734.92 | 4823.69 | 4024.29 | 4884.93 | **3746.53** |
| | 997 | 20 | 26,299.21 | 29,846.44 | 23,297.02 | 29,784.10 | **21,928.51** |
| Inspiral | 30 | 5 | 5794.77 | 5809.83 | **4910.75** | 5632.15 | 4984.50 |
| | 50 | 10 | 11,287.73 | 11,462.55 | 10,753.68 | 10,900.68 | **10,621.71** |
| | 100 | 15 | 21,956.52 | 23,105.28 | **21,596.61** | 23,764.72 | 21,811.10 |
| | 1000 | 20 | 213,499.93 | 215,095.96 | 206,350.92 | 217,434.62 | **199,377.12** |
| Montage | 20 | 5 | 251.98 | 271.45 | 218.30 | 263.59 | **202.55** |
| | 60 | 10 | 606.30 | 629.42 | **523.16** | 635.46 | 527.47 |
| | 100 | 15 | 1239.61 | 1273.93 | 1199.58 | 1268.10 | **1189.03** |
| | 300 | 20 | 12,522.88 | 12,442.48 | 12,550.89 | 12,596.38 | **12,211.07** |
| Sipht | 29 | 5 | 6945.80 | 8440.98 | 6276.62 | 9151.73 | **6134.93** |
| | 58 | 10 | 17,646.83 | 17,615.98 | 17,144.48 | 22,165.97 | **16,875.70** |
| | 97 | 15 | **25,209.25** | 29,142.36 | 26,401.31 | 35,329.01 | 26,871.82 |
| | 968 | 20 | 241,324.62 | 238,280.87 | 227,460.23 | 243,416.87 | **226,718.51** |

**Table 6** ECs of various scientific workflows

| Scenario | Tasks | VM | ACO | PSO | WOA | HPWOA |
|---|---|---|---|---|---|---|
| Cybershake | 30 | 5 | 322.69 | **311.72** | 327.85 | 329.86 |
| | 50 | 10 | 384.72 | 380.29 | 380.31 | **377.30** |
| | 100 | 15 | **415.67** | 438.39 | 440.79 | 417.52 |
| | 1000 | 20 | 709.83 | 751.69 | 741.51 | **695.70** |
| Epigenomics | 24 | 5 | 283.31 | **260.89** | 299.71 | 265.54 |
| | 47 | 10 | 386.71 | 390.78 | 407.56 | **370.64** |
| | 100 | 15 | 478.39 | 477.70 | 505.56 | **464.15** |
| | 997 | 20 | 998.07 | 965.39 | 1021.83 | **928.31** |
| Inspiral | 30 | 5 | 57.75 | **48.90** | 65.70 | 50.63 |
| | 50 | 10 | 98.25 | 92.45 | 102.35 | **90.34** |
| | 100 | 15 | 173.85 | 165.37 | 190.63 | **162.63** |
| | 1000 | 20 | 752.96 | 733.27 | 743.82 | **713.61** |
| Montage | 20 | 5 | 113.63 | 110.54 | 117.94 | **100.86** |
| | 60 | 10 | 169.64 | 165.52 | 174.72 | **156.86** |
| | 100 | 15 | 284.23 | 294.84 | 278.03 | **259.36** |
| | 300 | 20 | 578.95 | 597.63 | 551.06 | **505.19** |
| Sipht | 29 | 5 | 103.35 | **92.76** | 104.46 | 107.22 |
| | 58 | 10 | 109.84 | **103.69** | 107.89 | 108.56 |
| | 97 | 15 | 128.63 | 132.41 | 144.39 | **119.26** |
| | 968 | 20 | 507.73 | 527.85 | 578.74 | **458.26** |



| | Cybershake 30 | Cybershake 50 | Cybershake 100 | Cybershake 1000 |
|---|---|---|---|---|
| ▨ RR | 576.08 | 910.48 | 1971.99 | 11888.33 |
| ▨ ACO | 590.10 | 1024.30 | 2206.32 | 13565.32 |
| ▨ PSO | 535.69 | 847.08 | 1825.91 | 10872.81 |
| ▤ WOA | 634.36 | 977.96 | 1991.80 | 11805.25 |
| ▩ HPWOA | 536.84 | 822.29 | 1765.49 | 10485.74 |

**Fig. 5** Makespans under the CyberShake workflow for various algorithms

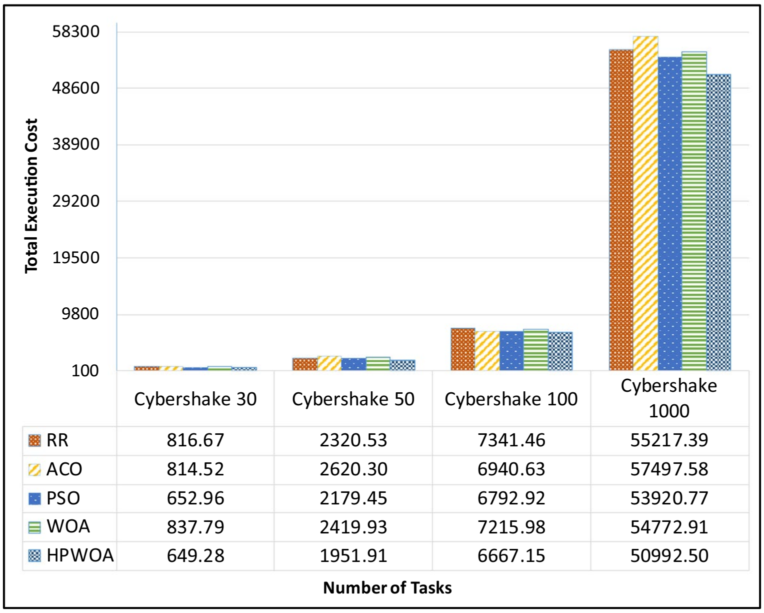
When the proposed algorithm was compared for 30, 50, 100 and 1000 tasks for the Cybershake workflow, the makespan decreased by 7.31%, 9.92%, 18.17%, 10.72%, 24.57%, 3.02%, 18.93%, 11.70%, 24.97%, 3.42%, and 12.82% and 13.38%, 29.37%, 3.69%, and 12.58% for the RR, ACO, and WOA algorithms, respectively. Similarly, when comparing the proposed algorithm to the PSO algorithm, it was discovered that the makespan increased by only 0.21% for 30 tasks.

The simulation results for the RR, ACO, PSO, WOA, and proposed HPWOA are shown in Fig. 6 for the TEC measurements. The HPWOA showed a reduction in TEC for all task sizes compared to the RR, ACO, and WOA algorithms. For 30 tasks, the HPWOA reduced the TEC by 25.78%, 25.45%, 0.57% and 29.03% compared to the RR, ACO, PSO and WOA algorithms, respectively. For 50 tasks, the HPWOA reduced the TEC by 18.89%, 34.24%, 11.66% and 23.98% compared to those of the RR, ACO, PSO and WOA algorithms, respectively. Finally, for 100 and 1000 tasks, the HPWOA reduced the TEC by varying percentages for the RR, ACO, and WOA algorithms.

In Fig. 7, the results of the EC for the ACO, PSO, WOA and HPWOA are presented. When the HPWOA was applied for 30 tasks, the EC increased by 2.17%, 5.50% and 0.61% for the ACO, PSO and WOA, respectively. For 50 tasks, the EC decreased by 1.97%, 0.79% and 0.80% compared to that of the ACO algorithm, PSO algorithm and WOA, respectively. For 100 and 1000 tasks, the EC for the HPWOA decreased by 5.0%, 5.57% and 2.03%, 8.05%, and 6.59% for the PSO algorithm and WOA, respectively. Only the ACO algorithm showed an increase in EC for 100 tasks of 0.44% compared to the proposed algorithm.

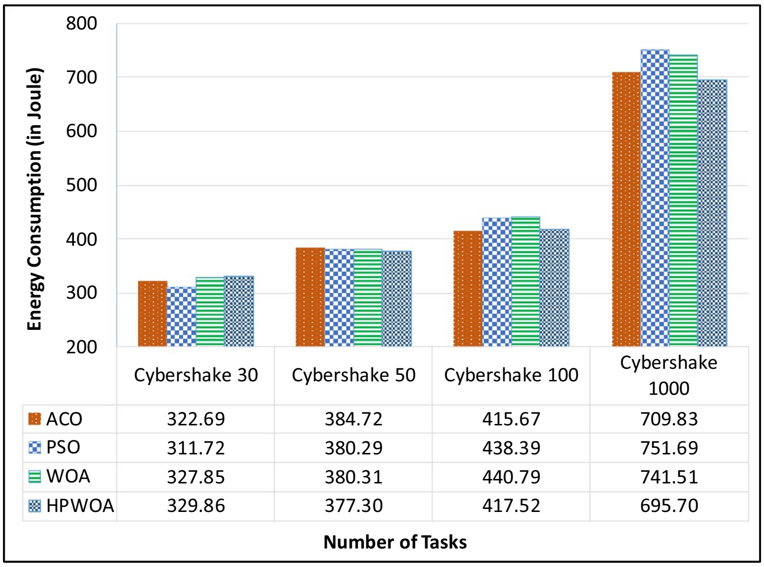
### 5.2 Epigenomic Workflow Performance Evaluation

The simulation results of the RR, ACO, PSO, WOA, and HPWOA under the epigenomic workflow are depicted in Fig. 8. The proposed algorithm outperformed all the other algorithms in all the other cases. For 24 tasks, applying the HPWOA reduced makespan by 8.82%, 5.52%, and 20.82% for the RR, ACO, and WOA algorithms,



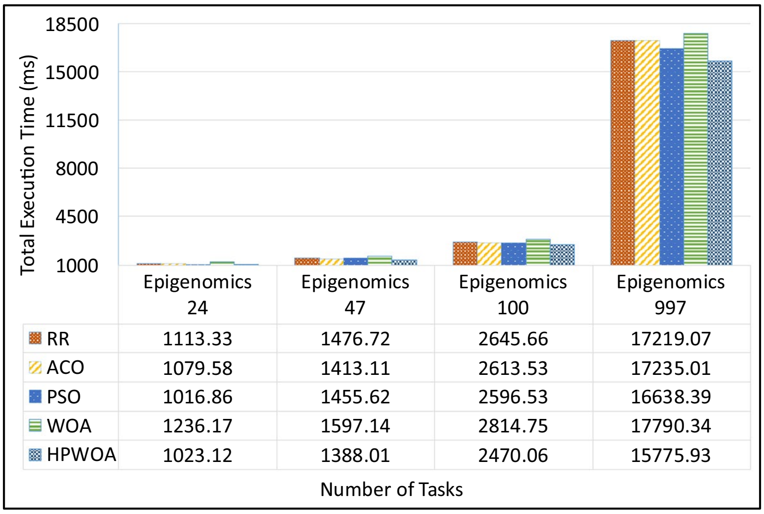
| | Cybershake 30 | Cybershake 50 | Cybershake 100 | Cybershake 1000 |
|---|---|---|---|---|
| ▨ RR | 816.67 | 2320.53 | 7341.46 | 55217.39 |
| ▨ ACO | 814.52 | 2620.30 | 6940.63 | 57497.58 |
| ▨ PSO | 652.96 | 2179.45 | 6792.92 | 53920.77 |
| ▤ WOA | 837.79 | 2419.93 | 7215.98 | 54772.91 |
| ▩ HPWOA | 649.28 | 1951.91 | 6667.15 | 50992.50 |

**Fig. 6** TEC under the CyberShake workflow for various algorithms



| | Cybershake 30 | Cybershake 50 | Cybershake 100 | Cybershake 1000 |
|---|---|---|---|---|
| ▨ ACO | 322.69 | 384.72 | 415.67 | 709.83 |
| ▨ PSO | 311.72 | 380.29 | 438.39 | 751.69 |
| ▤ WOA | 327.85 | 380.31 | 440.79 | 741.51 |
| ▩ HPWOA | 329.86 | 377.30 | 417.52 | 695.70 |

**Fig. 7** ECs under the Cybershake workflow for various algorithms

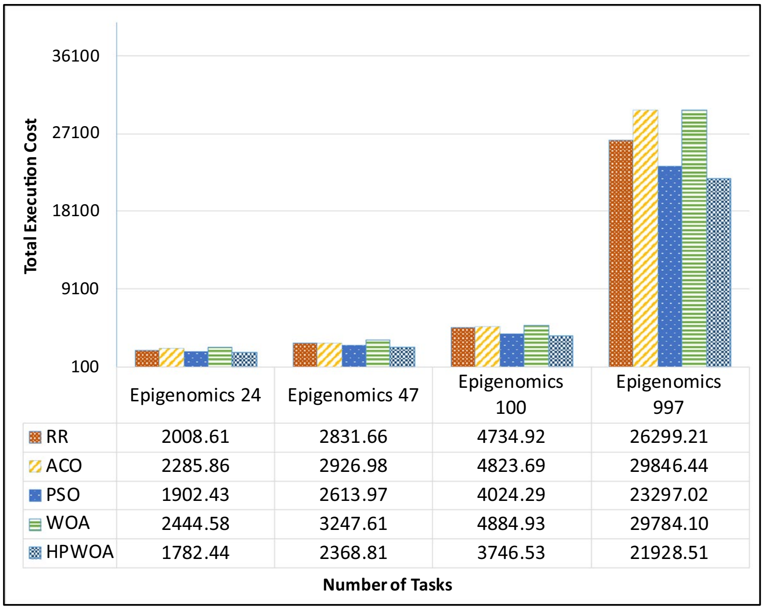


| | Epigenomics 24 | Epigenomics 47 | Epigenomics 100 | Epigenomics 997 |
|---|---|---|---|---|
| ▨ RR | 1113.33 | 1476.72 | 2645.66 | 17219.07 |
| ▨ ACO | 1079.58 | 1413.11 | 2613.53 | 17235.01 |
| ■ PSO | 1016.86 | 1455.62 | 2596.53 | 16638.39 |
| ▤ WOA | 1236.17 | 1597.14 | 2814.75 | 17790.34 |
| ▩ HPWOA | 1023.12 | 1388.01 | 2470.06 | 15775.93 |

**Fig. 8** Makespan under the epigenomic workflow for various algorithms

respectively. Similarly, for 47 tasks, makespan was 6.39%, 1.81%, 4.87%, and 15.07% lower than those of RR, ACO, PSO, and the WOA, respectively.

For 100 and 1000 tasks, the HPWOA algorithm reduced makespan by 7.11%, 5.81%, 5.12%, 13.95%, and 9.15%, 9.25%, 5.47%, and 12.77%, respectively, for the RR, ACO, PSO, and WOA algorithms.



| | Epigenomics 24 | Epigenomics 47 | Epigenomics 100 | Epigenomics 997 |
|---|---|---|---|---|
| ▨ RR | 2008.61 | 2831.66 | 4734.92 | 26299.21 |
| ▨ ACO | 2285.86 | 2926.98 | 4823.69 | 29846.44 |
| ■ PSO | 1902.43 | 2613.97 | 4024.29 | 23297.02 |
| ▤ WOA | 2444.58 | 3247.61 | 4884.93 | 29784.10 |
| ▩ HPWOA | 1782.44 | 2368.81 | 3746.53 | 21928.51 |

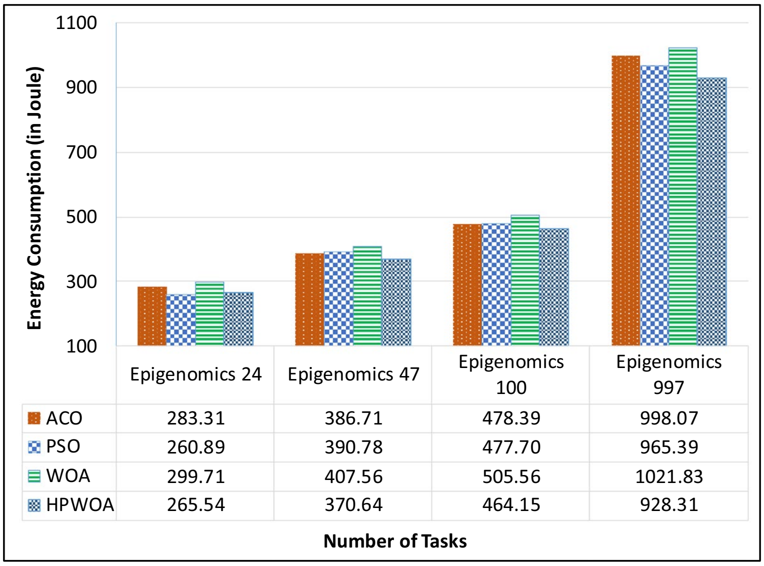
**Fig. 9** TEC under the epigenomic workflow for various algorithms

The simulation results of the TEC for the RR, ACO, PSO, WOA and HPWOA under the epigenomic workflow are presented in Fig. 9. The graph shows the reduction in TEC achieved by the proposed algorithm when compared to the other algorithms for different numbers of tasks, namely, 24, 47, 100 and 997. For example, for 24 tasks, the TEC decreased by 12.69%, 28.24%, 6.73% and 37.15% for the RR, ACO, PSO and WOA algorithms, respectively, while for 997 tasks, it decreased by 19.93%, 36.11%, 6.24% and 35.82%, respectively, for the same algorithms.

The simulation results of the ACO, PSO, WOA, and HPWOA for EC measurement under the epigenomic workflow are presented in Fig. 10. For 24 tasks, applying the HPWOA reduced the EC by 6.69% and 12.87% and increased it by 1.75% for the ACO, PSO, and WOA algorithms, respectively. Similarly, for 47 tasks, the EC decreased by 4.34%, 5.43%, and 9.96% compared to that of the ACO algorithm, PSO algorithm, and WOA, respectively. Moreover, for 100 and 997 tasks, the EC was reduced by 3.07%, 2.92%, 8.92%, and 7.52% for the HPWOA compared to the ACO, PSO, and WOA, respectively.

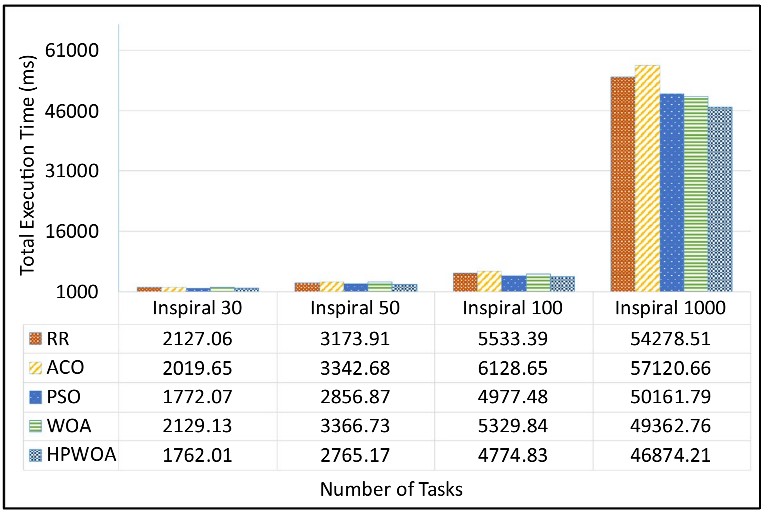
## 5.3 Inspiral Workflow Performance Evaluation

In Fig. 11, the results of makespan in the simulation of the RR, ACO, PSO, WOA and HPWOA under the Inspiral workflow are presented. The HPWOA algorithm outperformed all the other algorithms. For 30 tasks, the makespan measured with the HPWOA decreased by 20.72%, 14.62%, 0.57% and 20.84% for the RR, ACO, WOA and PSO algorithms, respectively. Similarly, for 50 tasks, makespan was reduced by 14.78%, 20.88%, 3.32% and 21.75% compared to RR, ACO, PSO and the WOA, respectively. For 100 and 1000 tasks, the makespan with the HPWOA decreased by 15.89%, 28.35%, 4.24%, and 11.62% and 15.80%, 21.86%, 7.01%, and 5.31% for the RR, ACO, PSO and WOA algorithms, respectively.



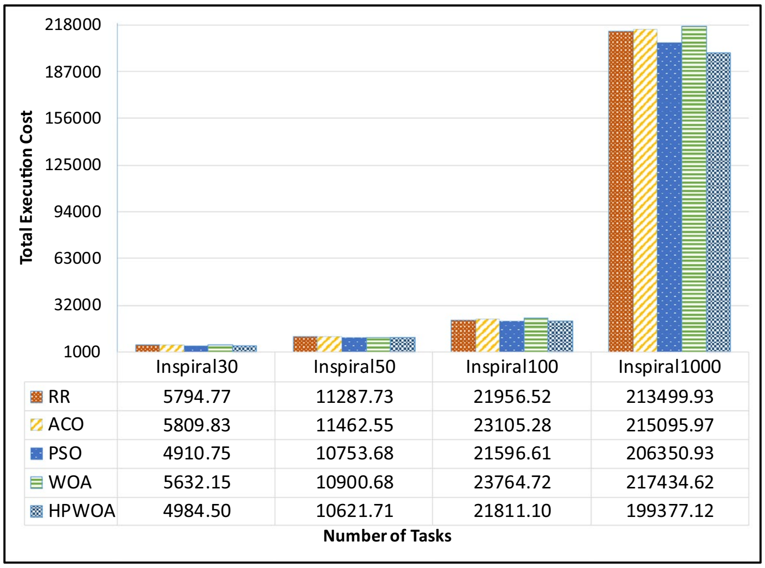
| | Epigenomics 24 | Epigenomics 47 | Epigenomics 100 | Epigenomics 997 |
|---|---|---|---|---|
| ▨ ACO | 283.31 | 386.71 | 478.39 | 998.07 |
| ▨ PSO | 260.89 | 390.78 | 477.70 | 965.39 |
| ▤ WOA | 299.71 | 407.56 | 505.56 | 1021.83 |
| ▩ HPWOA | 265.54 | 370.64 | 464.15 | 928.31 |

**Fig. 10** ECs calculated via the epigenomeomic workflow for various algorithms



| | Inspiral 30 | Inspiral 50 | Inspiral 100 | Inspiral 1000 |
|---|---|---|---|---|
| RR | 2127.06 | 3173.91 | 5533.39 | 54278.51 |
| ACO | 2019.65 | 3342.68 | 6128.65 | 57120.66 |
| PSO | 1772.07 | 2856.87 | 4977.48 | 50161.79 |
| WOA | 2129.13 | 3366.73 | 5329.84 | 49362.76 |
| HPWOA | 1762.01 | 2765.17 | 4774.83 | 46874.21 |

**Fig. 11** Makespan under the spiral workflow for various algorithms



| | Inspiral30 | Inspiral50 | Inspiral100 | Inspiral1000 |
|---|---|---|---|---|
| RR | 5794.77 | 11287.73 | 21956.52 | 213499.93 |
| ACO | 5809.83 | 11462.55 | 23105.28 | 215095.97 |
| PSO | 4910.75 | 10753.68 | 21596.61 | 206350.93 |
| WOA | 5632.15 | 10900.68 | 23764.72 | 217434.62 |
| HPWOA | 4984.50 | 10621.71 | 21811.10 | 199377.12 |

**Fig. 12** TEC under the spiral workflow for various algorithms

The Total Execution Cost results for the simulations of the RR, ACO, PSO, WOA and HPWOA under the spiral workflow are presented in Fig. 12. The proposed algorithm outperformed all the other methods. For 30 tasks, the TEC decreased by 16.26%, 16.56%, and 12.99% for the RR, ACO, and WOA algorithms, respectively, an increase of 1.48% compared to that of PSO. Similarly, for 50 tasks, the TEC was reduced by 6.27%, 7.92%, 1.24%, and 2.63% compared to that of RR, ACO, PSO, and the WOA, respectively. For 100 and 1000 tasks, the TEC for the HPWOA decreased by 0.67%, 5.93%, and 8.96% and by 7.08%, 7.88%, 3.50%, and 9.06%, respectively, for the RR, ACO, PSO, and WOA algorithms. Only the PSO algorithm showed an increase in TEC for 100 tasks of 0.98% compared with the proposed algorithm.

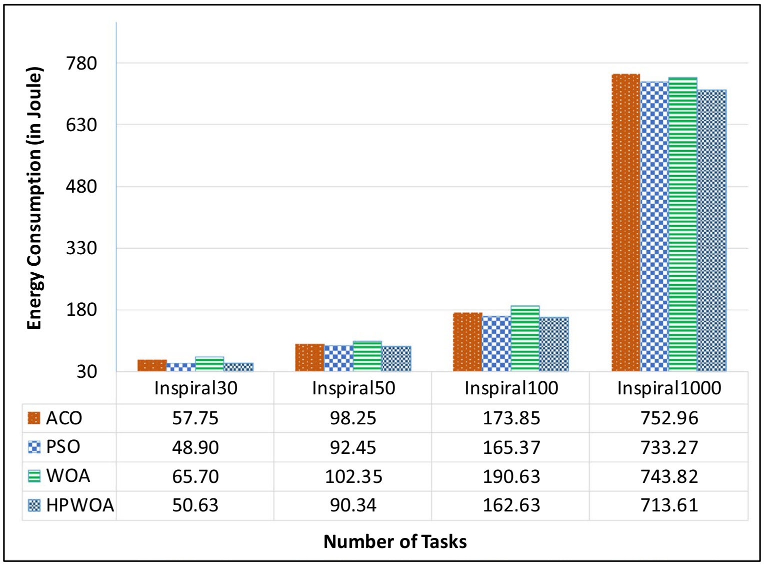
The results of the EC in the simulations of the PSO, ACO, WOA, and HPWOA are presented in Fig. 13. It was observed that applying the HPWOA resulted in a reduction in the EC in all the cases. Specifically, for 30 tasks, the HPWOA decreased the EC by 14.07% and 29.78% for the ACO algorithm and WOA, respectively, and increased the EC by 3.42% for the PSO algorithm. Similarly, for 50 tasks, the HPWOA algorithm decreased the EC by 8.76%, 2.33%, and 13.29% compared to those of ACO, PSO, and the WOA, respectively. For 100 and 1000 tasks, the EC was reduced for the HPWOA by 6.90%, 1.68%, and 17.21% for the ACO, PSO, and WOA algorithms, respectively.

## 5.4 Montage Workflow Performance Evaluation

When the proposed algorithm was assessed on 20, 60, 100, and 300 tasks for the Montage workflow, Fig. 14 shows that the makespan was reduced for all tasks. Specifically, the makespan reductions were 15.94%, 3.93%, 15.88%, 7.91%, 3.89%, 6.16%, 15.65%, 13.25%, 4.49%, 10.87%, 17.76%, 15.64%, 5.64%, 9.24%, and 24.85% for the RR, ACO, PSO, and WOA algorithms, respectively. Notably, compared with the proposed algorithm, only the ACO algorithm showed a slight increase in makespan for 20 tasks (0.78%).

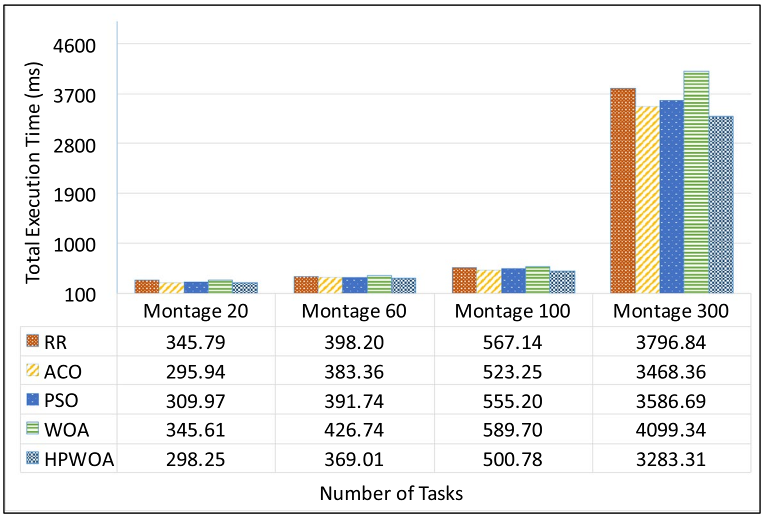
Figure 15 illustrates the TEC performance for 20, 60, 100 and 300 tasks under the montage workflow.

The results of the ACO, PSO, WOA, and HPWOA are presented in Fig. 16. The EC showed decreases of 12.66%, 9.60%, and 16.94% for the ACO, PSO, and WOA, respectively, when the HPWOA was applied to 20 tasks. Similarly, for 60 tasks, the EC was 8.15%, 5.52%, and 11.39% lower than that of the ACO, PSO, and WOA algorithms,



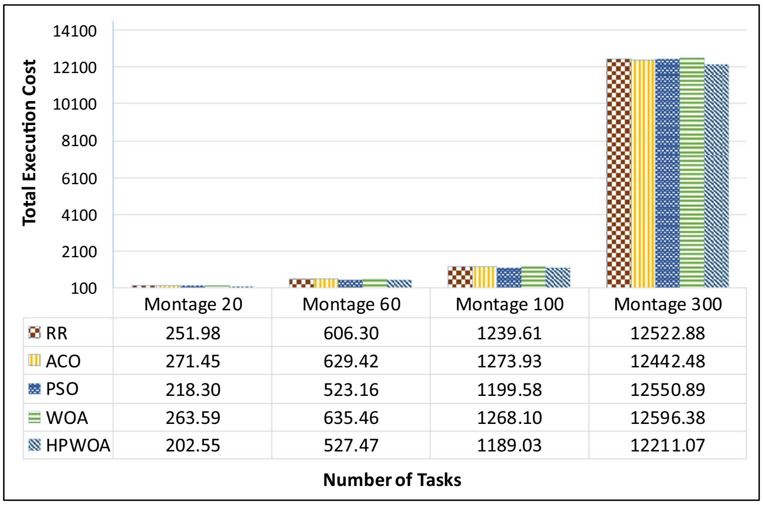
| | Inspiral30 | Inspiral50 | Inspiral100 | Inspiral1000 |
|---|---|---|---|---|
| ▨ ACO | 57.75 | 98.25 | 173.85 | 752.96 |
| ▨ PSO | 48.90 | 92.45 | 165.37 | 733.27 |
| ▤ WOA | 65.70 | 102.35 | 190.63 | 743.82 |
| ▩ HPWOA | 50.63 | 90.34 | 162.63 | 713.61 |

**Fig. 13** ECs under the spiral workflow for various algorithms



| | Montage 20 | Montage 60 | Montage 100 | Montage 300 |
|---|---|---|---|---|
| RR | 345.79 | 398.20 | 567.14 | 3796.84 |
| ACO | 295.94 | 383.36 | 523.25 | 3468.36 |
| PSO | 309.97 | 391.74 | 555.20 | 3586.69 |
| WOA | 345.61 | 426.74 | 589.70 | 4099.34 |
| HPWOA | 298.25 | 369.01 | 500.78 | 3283.31 |

**Fig. 14** Makespan under the Montage workflow for various algorithms



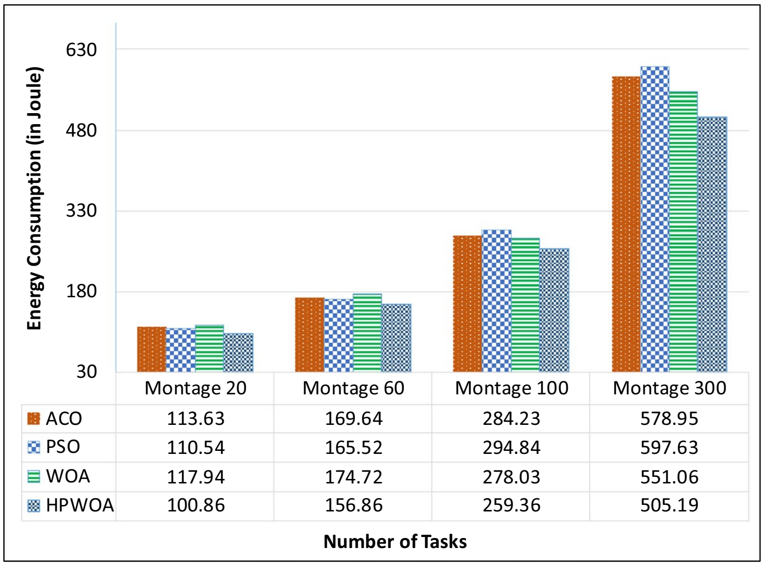
| | Montage 20 | Montage 60 | Montage 100 | Montage 300 |
|---|---|---|---|---|
| RR | 251.98 | 606.30 | 1239.61 | 12522.88 |
| ACO | 271.45 | 629.42 | 1273.93 | 12442.48 |
| PSO | 218.30 | 523.16 | 1199.58 | 12550.89 |
| WOA | 263.59 | 635.46 | 1268.10 | 12596.38 |
| HPWOA | 202.55 | 527.47 | 1189.03 | 12211.07 |

**Fig. 15** TEC under Montage Workflow for various algorithms

respectively. For 100 and 300 tasks, the EC for the HPWOA decreased by 9.59%, 13.68%, and 7.20%, and 14.60%, 18.30%, and 9.08%, respectively, for the ACO, PSO, and WOA algorithms.

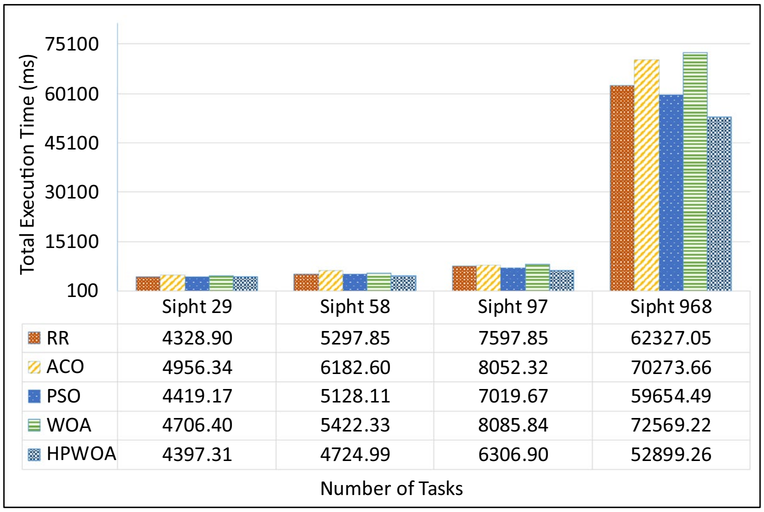
## 5.5 Sipht Workflow Performance Evaluation

Figure 17 displays the simulation results of the makespan for the RR, ACO, PSO, WOA, and HPWOA under the Sipht workflow. The proposed algorithm outperformed all the others. Compared with those of the proposed algorithm, the makespan were reduced for 29,



| | Montage 20 | Montage 60 | Montage 100 | Montage 300 |
|---|---|---|---|---|
| ACO | 113.63 | 169.64 | 284.23 | 578.95 |
| PSO | 110.54 | 165.52 | 294.84 | 597.63 |
| WOA | 117.94 | 174.72 | 278.03 | 551.06 |
| HPWOA | 100.86 | 156.86 | 259.36 | 505.19 |

**Fig. 16** ECs under the Montage workflow for various algorithms

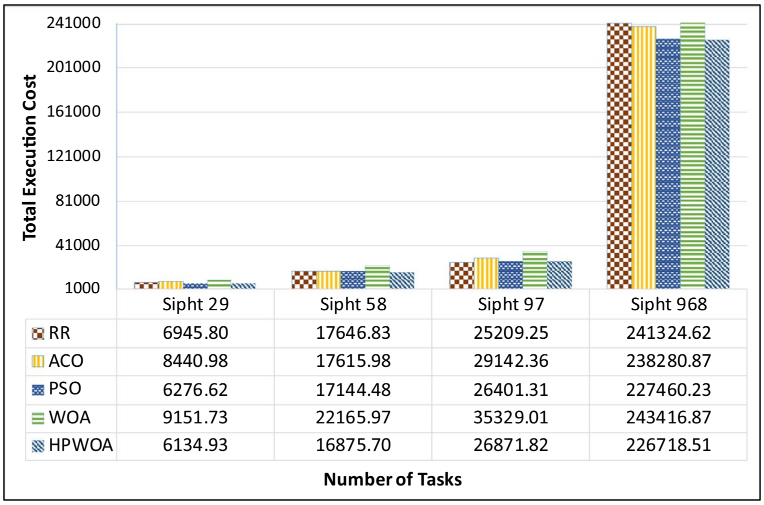


| | Sipht 29 | Sipht 58 | Sipht 97 | Sipht 968 |
|---|---|---|---|---|
| RR | 4328.90 | 5297.85 | 7597.85 | 62327.05 |
| ACO | 4956.34 | 6182.60 | 8052.32 | 70273.66 |
| PSO | 4419.17 | 5128.11 | 7019.67 | 59654.49 |
| WOA | 4706.40 | 5422.33 | 8085.84 | 72569.22 |
| HPWOA | 4397.31 | 4724.99 | 6306.90 | 52899.26 |

**Fig. 17** Makespan under the Sipht workflow for various algorithms

58, 97, and 968 tasks by 12.71%, 0.50%, 7.03%, 12.12%, 30.85%, 8.53%, 14.76%, 20.47%, 27.67%, 11.30%, and 28.21% and by 17.82%, 32.84%, 12.77%, and 37.18%, respectively, for the ACO, PSO, and WOA algorithms. Similarly, compared to that of the RR algorithm, the makespan increased by 1.56% for 29 tasks.

Figure 18 shows the TEC simulation results for the RR, ACO, PSO, WOA and proposed algorithm (HPWOA). The x-axis and y-axis of the graph show the number of tasks and the TEC, respectively. When the proposed algorithm was compared for 29, 58, 97 and

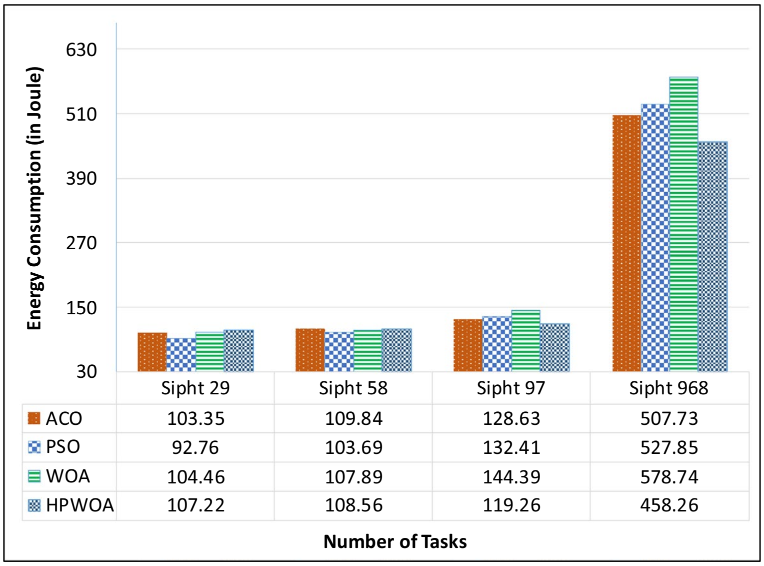


| | Sipht 29 | Sipht 58 | Sipht 97 | Sipht 968 |
|---|---|---|---|---|
| RR | 6945.80 | 17646.83 | 25209.25 | 241324.62 |
| ACO | 8440.98 | 17615.98 | 29142.36 | 238280.87 |
| PSO | 6276.62 | 17144.48 | 26401.31 | 227460.23 |
| WOA | 9151.73 | 22165.97 | 35329.01 | 243416.87 |
| HPWOA | 6134.93 | 16875.70 | 26871.82 | 226718.51 |

**Fig. 18** TEC under the Sipht workflow for various algorithms

968 tasks for the Sipht workflow, the TEC decreased by 13.22%, 37.59%, 2.31%, 49.17%, 4.57%, 4.39%, 1.59%, 31.35%, 8.45%, and 31.47% and by 6.44%, 5.10%, 0.33%, and 7.37% for the RR, ACO, and WOA algorithms, respectively. Similarly, when the proposed algorithm was compared to the RR and PSO algorithms, it was found that the TEC increased for 97 tasks by 6.19% and 1.75%, respectively.

The results of the EC in the simulations of the ACO, PSO, WOA, and HPWOA are presented in Fig. 19. For 29 tasks, the application of the HPWOA resulted in increases of



| | Sipht 29 | Sipht 58 | Sipht 97 | Sipht 968 |
|---|---|---|---|---|
| ACO | 103.35 | 109.84 | 128.63 | 507.73 |
| PSO | 92.76 | 103.69 | 132.41 | 527.85 |
| WOA | 104.46 | 107.89 | 144.39 | 578.74 |
| HPWOA | 107.22 | 108.56 | 119.26 | 458.26 |

**Fig. 19** ECs under the Sipht workflow for various algorithms

3.60%, 13.49%, and 2.57% in the EC measures for the ACO, PSO, and WOA algorithms, respectively. In comparison, for 58 tasks, the EC decreased by 1.18% and increased by 4.48% and 0.61% for the ACO, PSO, and WOA algorithms, respectively. Moreover, for 97 and 968 tasks, the ECs for the HPWOA decreased by 7.86%, 11.03%, and 21.08% and by 10.80%, 15.19%, and 26.29%, respectively, for the ACO, PSO, and WOA algorithms.

## 5.6 Ablation Studies

In this section, we will be contrasting the outcomes of the innovative hybrid HPWOA with those achieved by the current PSO and WOA algorithms. This comparison highlights the individual effects of the WOA and PSO algorithms in contrast to the hybrid approach. The ablation study for TEC, makespan and EC is presented below.

### 5.6.1 Ablation Studies for Makespan

The ablation study of the proposed HPWOA highlights its superiority over the individual PSO and WOA algorithms in terms of makespan. Its exceptional performance is particularly notable in the sipht workflow, where it effectively handles 58 and 97 tasks across 25 iterations, respectively. As demonstrated in Table 7, the algorithm's resilience and flexibility further establish its reputation as a best choice for complex computational assignments. In the table below, indexes 1 and 2 are considered, which show the simulation results for the PSO algorithm and WOA, respectively. The makespan values for the WOA and PSO algorithm were 5422.33 and 5128.11, respectively. In comparison, index 3, representing the HPWOA, shows a significant improvement, with a TET value of 4724.99. Similarly, the other indices in the table show enhancements consistently across all the ablation studies.

### 5.6.2 Ablation Studies for TEC

The ablation analysis of the HPWOA demonstrates its superiority over the standalone PSO and WOA algorithms in terms of total execution cost (TEC). Its remarkable efficiency excels in the epigenomics workflow, effectively managing 47 and 100 tasks over 25 iterations. This is illustrated in Table 8, which also showcases the algorithm's resilience and flexibility, solidifying its status as a top choice for challenging computational assignments. In the table, indexes 1 and 2 are considered, which show the simulation results for the PSO algorithm and WOA, respectively. The TEC values for PSO and the WOA were 2613.97 and 3247.61, respectively. In comparison, index 3, representing the HPWOA, shows a

**Table 7** Results of Ablation Study of the TET for sipht workflow

| Index | WOA | PSO | Tasks (in numbers) | Iterations | TET |
|---|---|---|---|---|---|
| 1 | ✗ | ✓ | 58 | 25 | 5128.11 |
| 2 | ✓ | ✗ | 58 | 25 | 5422.33 |
| 3 | ✓ | ✓ | 58 | 25 | **4724.99** |
| 4 | ✗ | ✓ | 97 | 25 | 7019.67 |
| 5 | ✓ | ✗ | 97 | 25 | 8085.84 |
| 6 | ✓ | ✓ | 97 | 25 | **6306.90** |

**Table 8** Results of the ablation investigation for TEC as per the epigenomic workflow

| Index | WOA | PSO | Tasks (in numbers) | Iterations | TEC |
|---|---|---|---|---|---|
| 1 | ✗ | ✓ | 47 | 25 | 2613.97 |
| 2 | ✓ | ✗ | 47 | 25 | 3247.61 |
| 3 | ✓ | ✓ | 47 | 25 | **2368.81** |
| 4 | ✗ | ✓ | 100 | 25 | 4024.29 |
| 5 | ✓ | ✗ | 100 | 25 | 4884.93 |
| 6 | ✓ | ✓ | 100 | 25 | **3746.53** |

significant improvement, with a TEC value of 2368.81. Similarly, the other indices in the table show enhancements consistently across all the ablation studies.

### 5.6.3 Ablation Studies for EC

Research on the HPWOA underscores its clear advantages over the individual PSO and WOA algorithms in terms of energy consumption (EC). Its outstanding performance is particularly evident in the montage workflow, where it effectively manages 60 and 100 tasks across 25 iterations. These findings are detailed in Table 9, which also demonstrates the algorithm's resilience and adaptability, establishing it as premier option for challenging computational tasks.

Specifically, when comparing indexes 1 and 2, which represent the simulation results for the PSO and WOA algorithms, their EC values were 165.52 and 174.72, respectively. In contrast, index 3, representing the HPWOA, shows a significant improvement, with an EC value of 156.86. The other indexes in the table similarly reveal enhancements across all the ablation studies.

## 5.7 Statistical Results

In assessing the effectiveness of the proposed algorithm, we employed nonparametric statistical analyses like the Friedman and Wilcoxon tests. Furthermore, this subsection presents the metrics associated with the speedup and SLR to conduct a thorough assessment.

**Table 9** Results of the ablation research on ECs using a montage workflow

| Index | WOA | PSO | Tasks (in numbers) | Iterations | EC |
|---|---|---|---|---|---|
| 1 | ✗ | ✓ | 60 | 25 | 165.52 |
| 2 | ✓ | ✗ | 60 | 25 | 174.72 |
| 3 | ✓ | ✓ | 60 | 25 | **156.86** |
| 4 | ✗ | ✓ | 100 | 25 | 294.84 |
| 5 | ✓ | ✗ | 100 | 25 | 278.03 |
| 6 | ✓ | ✓ | 100 | 25 | **259.36** |

**Table 10** The Wilcoxon test indicates the importance of algorithm performance in TET at the Sipht workflow

| Work flow | Iteration (in numbers) | WOA | PSO | HPWOA | Wilcoxon test for PSO vs HPWOA | | Wilcoxon test for WOA vs HPWOA | |
|---|---|---|---|---|---|---|---|---|
| Sipht | 29 | 4706.396 | 4419.169 | 4397.307 | $n_1$ | 4 | $n_1$ | 4 |
| | 58 | 5422.325 | 5128.114 | 4724.993 | $n_2$ | 4 | $n_2$ | 4 |
| | 97 | 8085.835 | 7019.672 | 6306.897 | sum | 28 | Sum | 25 |
| | 968 | 72,569.21 | 59,654.49 | 52,899.25 | expectation | 18 | expectation | 18 |
| | | | | | std. error | 3.7947331 | std. error | 3.7947331 |
| | | | | | stat | 2.6352313 | stat | 1.8446619 |
| | | | | | *p*-value | 0.004204 | *p*-value | 0.0325433 |

**Table 11** The Wilcoxon test indicates the importance of algorithm performance in TEC at the Epigenomics workflow

| Work flow | Iteration (in numbers) | WOA | PSO | HPWOA | Wilcoxon test for PSO vs HPWOA | | Wilcoxon test for WOA vs HPWOA | |
|---|---|---|---|---|---|---|---|---|
| Epigenomics | 24 | 5173.094 | 1902.43 | 3702.43 | $n_1$ | 4 | $n_1$ | 4 |
| | 47 | 5081.544 | 2613.97 | 3683.11 | $n_2$ | 4 | $n_2$ | 4 |
| | 100 | 5198.637 | 4024.29 | 3666.95 | Sum | 26 | Sum | 29 |
| | 997 | 5127.479 | 23,297.02 | 3777.35 | expectation | 18 | expectation | 18 |
| | | | | | std. error | 3.7947331 | std. error | 3.7947331 |
| | | | | | Stat | 2.1081851 | Stat | 2.8987545 |
| | | | | | *p*-value | 0.0175074 | *p*-value | 0.0018732 |

### 5.7.1 Wilcoxon Test

The Wilcoxon signed-rank test is a nonparametric statistical technique used to compare paired samples or repeated measurements within a single group. This approach becomes especially useful when the data fail to meet the normal distribution assumptions necessary for parametric tests. The test assesses whether there is a notable difference between paired observations by ranking the absolute differences in their values. Its resistance to outliers establishes it as a reliable option for analyzing data in cases where parametric assumptions cannot be met. Commonly utilized in diverse fields like experimental psychology, medicine, and engineering, the Wilcoxon test is employed to evaluate the importance of alterations or variances in paired observations. Typically, a standard significance level of 1% or 5% is adopted, with 5% being the commonly established threshold. At a significance level of 5%, when the p-value from the Wilcoxon test is below 0.05, it suggests significant differences between the two sets of values; otherwise, they are considered to have similar characteristics. In Tables 10, 11 and 12, the outcomes of the Wilcoxon test conducted on the sipht, epigenomic and montage datasets for time, cost and energy parameters are presented, considering various task numbers.

**Table 12** Wilcoxon test indicating the importance of algorithm performance in ECs in the Montage workflow

| Work flow | Iteration (in numbers) | WOA | PSO | HPWOA | Wilcoxon test for PSO vs HPWOA | | Wilcoxon test for WOA vs HPWOA | |
|---|---|---|---|---|---|---|---|---|
| Montage | 20 | 56.46 | 46.76 | 59.22 | $n_1$ | 4 | $n_1$ | 4 |
| | 60 | 59.89 | 58.69 | 60.56 | $n_2$ | 4 | $n_2$ | 4 |
| | 100 | 95.39 | 86.41 | 71.26 | Sum | 25 | Sum | 26 |
| | 300 | 530.74 | 475.85 | 410.26 | expectation | 18 | expectation | 18 |
| | | | | | std. error | 3.7947331 | std. error | 3.7947331 |
| | | | | | Stat | 1.8446619 | Stat | 2.1081851 |
| | | | | | *p*-value | 0.0325433 | *p*-value | 0.0175074 |

The *p*-values obtained from the Wilcoxon test on the algorithms are presented at a significance level of 0.05. Importantly, all *p*-values are below 0.05, indicating a significant difference between the proposed HPWOA and the other comparative methods.

### 5.7.2 Friedman Test

The Friedman test, coined after its creator Milton Friedman, is a nonparametric statistical technique employed for analysing distinctions among multiple correlated groups. Particularly useful in scenarios involving repeated measures designs or ranked data, this test assesses the presence of significant variations in the central tendencies of multiple datasets, making it suitable for scenarios where the normal distribution assumption may not be valid. The test assigns ranks to the observations in each group and then investigates if there are significant differences in the ranked means among the groups. The Friedman test is valuable for comparing the performance of multiple treatments or interventions and is commonly employed across various fields, including experimental studies, psychology, and research. In this research, the Friedman test was used to compare the efficacy of the proposed HPWOA with traditional methods. The resulting P value determines if the HPWOA shows a significant difference. The ranking methodology employed indicates that a lower rank signifies better performance. Table 13 presents the mean ranks of all algorithms over the tested dataset; results indicate the one in which the HPWOA has the best mean rank, followed by the WOA algorithm and the PSO.

**Table 13** Analyzing algorithmic performance using friedman test results

| Metrics | Tasks (in numbers) | WOA | PSO | HPWOA | *p*-value |
|---|---|---|---|---|---|
| ET | 100 | 3 | 2 | 1 | $< 0.001$ |
| | 1000 | 2 | 3 | 1 | |
| CCC | 100 | 2 | 3 | 1 | |
| | 1000 | 3 | 2 | 1 | |
| Fitness | 100 | 3 | 2 | 1 | |
| | 1000 | 2 | 3 | 1 | |

The assessment was carried out considering three crucial parameters: the cumulative communication cost, execution time, and fitness function. Based on the findings, it is clear that the proposed HPWOA surpasses other algorithms in terms of total execution time, cost, and energy. This is evident from its higher mean rank. In summary, the HPWOA generates superior solutions and demonstrates overall superiority, highlighting the effectiveness of integrating PSO with the WOA to enhance efficiency and solution quality.

## 6 Conclusion and Future Work

A novel metaheuristic approach called the HPWOA, proposed in this paper, is a combination of the PSO algorithm and the WOA. Five scientific workflows—montage, epigenomics, cybershake, sipht and inspiral—were used to evaluate the performance of the HPWOA with existing algorithms for different numbers of tasks. The proposed algorithm's goal is to minimize the overall execution time, cost and energy consumption. In all the cases, the proposed HPWOA outperforms the other methods in all the terms, as also supported by the statistical analysis. The average percentage decreases in makespan for the montage, epigenomic, cybershake, sipht and inspiral workflows were 10.64%, 8.21%, 12.77%, 17.08% and 14.22%, respectively. For TECs, the average percentage decreases in the montage, epigenomic, cybershake, sipht and inspiral workflows were 11.22%, 22.90%, 14.26%, 12.21% and 6.53%, respectively. For ECs, the average percentage decreases for montage, epigenomic, cybershake, sipht and inspiral workflows were 11.39%, 6.17%, 1.84%, 5.72% and 8.59%, respectively. When considering QoS parameters as the base, the sipht workflow outperforms other workflows in terms of makespan results. All the algorithms achieved an average time savings of 17.08%. Similarly, the epigenomic workflow performed best for determining TEC results, offering an average cost savings of 22.90% compared to that of other workflows. Finally, the montage workflow yields the best EC results, achieving an average energy savings of 11.39% compared to those of other workflows. From these results, the proposed approach user or service provider can choose the most suitable scientific workflow according to his or her requirements or preferences.

In future research, it may be worthwhile to explore additional metaheuristic algorithms to create a new hybrid algorithm. The resulting algorithm can then be evaluated based on the same parameters.

**Funding** This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

**Data Availability** No data were used for the research described in the article.

## Declarations

**Competing interests** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References

1. Khaleel, M. I. (2024). Region-aware dynamic job scheduling and resource efficiency for load balancing based on adaptive chaotic sparrow search optimization and coalitional game in cloud computing

environments. *Journal of Network and Computer Applications, 221*(November 2023), 103788. https://doi.org/10.1016/j.jnca.2023.103788
2. Behera, I., & Sobhanayak, S. (2024). Task scheduling optimization in heterogeneous cloud computing environments: A hybrid GA-GWO approach. *J. Parallel Distrib. Comput., 183*, 104766. https://doi.org/10.1016/j.jpdc.2023.104766
3. Khaledian, N., Khamforoosh, K., Akraminejad, R., Abualigah, L., & Javaheri, D. (2023). An energy-efficient and deadline-aware workflow scheduling algorithm in the fog and cloud environment. *Computing, 106*(1), 109–137. https://doi.org/10.1007/s00607-023-01215-4
4. Digitale, J. C., Martin, J. N., & Glymour, M. M. (2022). Tutorial on directed acyclic graphs. *Journal of Clinical Epidemiology, 142*, 264–267. https://doi.org/10.1016/j.jclinepi.2021.08.001
5. Menaka, M., & Sendhil Kumar, K. S. S. (2022). Workflow scheduling in cloud environment—challenges, tools, limitations and methodologies: A review. *Measurement Sensors, 24*(September), 100436. https://doi.org/10.1016/j.measen.2022.100436
6. Bansal, S., Aggarwal, H., & Aggarwal, M. (2022). A systematic review of task scheduling approaches in fog computing. *Transactions on Emerging Telecommunications Technologies, 33*(9), 4523. https://doi.org/10.1002/ett.4523
7. Bittencourt, L. F., Goldman, A., Madeira, E. R. M., Da Fonseca, N. L. S., & Sakellariou, R. (2018). Scheduling in distributed systems: A cloud computing perspective. *Computer Science Review, 30*, 31–54. https://doi.org/10.1016/j.cosrev.2018.08.002
8. Shao, K., Song, Y., & Wang, B. (2023). PGA: A new hybrid PSO and GA method for task scheduling with deadline constraints in distributed computing. *Mathematics, 11*(6), 1–16. https://doi.org/10.3390/math11061548
9. Pandey, S., Wu, L., Guru, S. M., Buyya, R. (2014). A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments. In *Proceedings of international conference on advanced information networking and applications, AINA*, no. January, pp. 400–407. https://doi.org/10.1109/AINA.2010.31.
10. Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization, pp. 1942–1948. https://doi.org/10.1002/9780470612163.
11. Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software, 95*, 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008
12. Thennarasu, S. R., Selvam, M., & Srihari, K. (2021). A new whale optimizer for workflow scheduling in cloud computing environment. *Journal of Ambient Intelligence and Humanized Computing, 12*(3), 3807–3814. https://doi.org/10.1007/s12652-020-01678-9
13. Bansal, S., Aggarwal, M., Aggarwal, H. (2019). Advancements and applications in fog computing. In *Security designs for the cloud, IoT, and social networking*, pp. 207–240. https://doi.org/10.1002/9781119593171.ch14.
14. Meena, V., Arvind, V., Vijayalakshmi, P., Kalpana, V., Senthil Kumar, J. (2018). Optimized task clustering for mobile cloud computing using Workflowsim. In *Proceedings of the 2nd international conference on inventive systems and control, ICISC 2018*, 2018, pp. 1000–1005. https://doi.org/10.1109/ICISC.2018.8398952.
15. Sahraei, S. H., Kashani, M. M. R., Rezazadeh, J., & Farahbakhsh, R. (2019). Efficient job scheduling in cloud computing based on genetic algorithm. *International Journal of Communication Networks and Distributed Systems, 22*(4), 447–467. https://doi.org/10.1504/IJCNDS.2019.099968
16. Tawfeek, M., El-Sisi, A., Keshk, A., & Torkey, F. (2021). Cloud task scheduling based on ant colony optimization. *International Arab Journal of Information Technology, 12*(2), 129–137.
17. Dasgupta, K., Mandal, B., Dutta, P., Mandal, J. K., & Dam, S. (2013). A Genetic algorithm (GA) based load balancing strategy for cloud computing. *Procedia Technology, 10*, 340–347. https://doi.org/10.1016/j.protcy.2013.12.369
18. Ge, Y., & Wei, G. (2010). GA-based task scheduler for the cloud computing systems. In *Proceedings of 2010 international conference web information systems and mining, WISM 2010*, vol. 2, pp. 181–186. https://doi.org/10.1109/WISM.2010.87.
19. Fakhfakh, F., Kacem, H. H., & Kacem, A. H. (2014). "orkflow scheduling in cloud computing: A survey. In *Proceedings of IEEE international enterprise distributed object computing workshops EDOCW*, pp. 372–378. https://doi.org/10.1109/EDOCW.2014.61.
20. Mikram, H., El Kafhali, S., & Saadi, Y. (2024). HEPGA: A new effective hybrid algorithm for scientific workflow scheduling in cloud computing environment. *Simulation Modelling Practice and Theory, 130*(October 2023), 102864. https://doi.org/10.1016/j.simpat.2023.102864
21. Ayoubi, M., Ramezanpour, M., & Khorsand, R. (2021). An autonomous IoT service placement methodology in fog computing. *Software: - Practice and Experience, 51*(5), 1097–1120. https://doi.org/10.1002/spe.2939

22. Arora, N., & Banyal, R. K. (2021). Workflow scheduling using particle swarm optimization and gray wolf optimization algorithm in cloud computing. *Concurrency and Computation: Practice and Experience, 33*(16), 1–16. https://doi.org/10.1002/cpe.6281
23. Li, H., Wang, D., Cañizares Abreu, J. R., Zhao, Q., & Bonilla Pineda, O. (2021). PSO+LOA: Hybrid constrained optimization for scheduling scientific workflows in the cloud. *The Journal of Supercomputing, 77*(11), 13139–13165. https://doi.org/10.1007/s11227-021-03755-y
24. Baker, T., et al. (2020). A secure fog-based platform for SCADA-based IoT critical infrastructure. *Software: - Practice and Experience, 50*(5), 503–518. https://doi.org/10.1002/spe.2688
25. Unhelkar, B., Joshi, S., Sharma, M., Prakash, S., Mani, A. K., & Prasad, M. (2022). Enhancing supply chain performance using RFID technology and decision support systems in the industry 4.0–A systematic literature review. *International Journal of Information Management Data Insights, 2*(2), 100084. https://doi.org/10.1016/j.jjimei.2022.100084
26. Kaiwartya, O., et al. (2018). Virtualization in wireless sensor networks: Fault tolerant embedding for internet of things. *IEEE Internet of Things Journal, 5*(2), 571–580. https://doi.org/10.1109/JIOT.2017.2717704
27. Ghobaei-Arani, M., Rahmanian, A. A., Souri, A., & Rahmani, A. M. (2018). A moth-flame optimization algorithm for web service composition in cloud computing: Simulation and verification. *Software: - Practice and Experience, 48*(10), 1865–1892. https://doi.org/10.1002/spe.2598
28. Kaur, G., Kalra, M., Bothra, S. K., Singhal, S., & Goyal, H. (2022). Cost effective hybrid genetic algorithm for workflow scheduling in cloud. *System Research and Information Technologies, 2022*(3), 121–138. https://doi.org/10.20535/SRIT.2308-8893.2022.3.08
29. Wada, H., Suzuki, J., Yamano, Y., & Oba, K. (2011). Evolutionary deployment optimization for service-oriented clouds. *Software: - Practice and Experience, 41*(5), 469–493. https://doi.org/10.1002/spe.1032
30. Dastjerdi, A. V., Garg, S. K., Rana, O. F., & Buyya, R. (2015). CloudPick: A framework for QoS-aware and ontology-based service deployment across clouds. *Software: - Practice and Experience, 45*(2), 197–231. https://doi.org/10.1002/spe.2288
31. Kaur, A., Singh, P., Singh-Batth, R., & Peng-Lim, C. (2022). Deep-Q learning-based heterogeneous earliest finish time scheduling algorithm for scientific workflows in cloud. *Software: - Practice and Experience, 52*(3), 689–709. https://doi.org/10.1002/spe.2802
32. Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *IEEE International Conference on Neural Networks, 4*, 1942–1948. https://doi.org/10.1002/9780470612163
33. Jain, S., & Meena, J. (2019). *Workflow scheduling algorithms in cloud computing: An analysis, analogy, and provocations*, vol. 74. Springer, Singapore. https://doi.org/10.1007/978-981-13-7082-3_57.
34. Konjaang, J. K., & Xu, L. (2021). Multi-objective workflow optimization strategy (MOWOS) for cloud computing. *Journal of Cloud Computing, 10*(1), 1–19. https://doi.org/10.1186/s13677-020-00219-1
35. Pasdar, A., Lee, Y. C., & Almi'ani, K. (2020). Hybrid scheduling for scientific workflows on hybrid clouds. *Computer Networks, 181*(July), 107438. https://doi.org/10.1016/j.comnet.2020.107438
36. Taghinezhad-Niar, A., Pashazadeh, S., & Taheri, J. (2022). Energy-efficient workflow scheduling with budget-deadline constraints for cloud. *Computing, 104*(3), 601–625. https://doi.org/10.1007/s00607-021-01030-9
37. Garg, N., Singh, D., Goraya, M. S. (2021). *Energy and resource efficient workflow scheduling in a virtualized cloud environment*, vol. 24, no. 2. Springer US. https://doi.org/10.1007/s10586-020-03149-4.
38. Trivedi, V., Prakash, S., & Ramteke, M. (2017). Optimized on-line control of MMA polymerization using fast multi-objective DE. *Materials and Manufacturing Processes, 32*(10), 1144–1151. https://doi.org/10.1080/10426914.2016.1257802
39. Abbott, B. P., et al. (2009). LIGO: The laser interferometer gravitational-wave observatory. *Reports on Progress in Physics*. https://doi.org/10.1088/0034-4885/72/7/076901
40. Graves, R., et al. (2011). CyberShake: A physics-based seismic hazard model for Southern California. *Pure and Applied Geophysics, 168*(3–4), 367–381. https://doi.org/10.1007/s00024-010-0161-6
41. Florean, C., Schnekenburger, M., Grandjenette, C., Dicato, M., & Diederich, M. (2011). Epigenomics of leukemia: From mechanisms to therapeutic applications. *Epigenomics, 3*(5), 581–609. https://doi.org/10.2217/epi.11.73
42. Deelman, E., et al. (2015). Pegasus, a workflow management system for science automation. *Future Generation Computer Systems, 46*, 17–35. https://doi.org/10.1016/j.future.2014.10.008

**Er. Sumit Bansal** is an active researcher who is always keen to learn new technologies. He is pursuing Ph.D. in Computer Science and Engineering at Punjabi University, Patiala. He has research interests in Fog computing, Cloud computing, AWS. He is working as assistant professor in Gurukula Kangri (DU), Haridwar.

**Dr. Himanshu Aggarwal** is currently serving as Professor in Department of Computer Science and Engineering at Punjabi University, Patiala. He has more than 30 years of teaching experience. He is an active researcher who has supervised more than 40 M.Tech. Dissertations and contributed 100 articles in various research journals. He is guiding Ph.D.° to 8 scholars and 12 have already completed their Ph.D. He is on the Editorial and Review Boards of several journals of repute. His areas of interest are Software Engineering, Computer Networks, Information Systems, ERP and Parallel Computing.