Journal of Engineering
and Applied Science

RESEARCH Open Access

Enhanced Whale Optimization Algorithm for task scheduling in cloud computing environments

Yanfeng Zhang[1*] and Jiawei Wang[1]

*Correspondence:
chinazhyf@163.com

[1] College of Artificial Intelligence, Jiaozuo University, Jiaozuo 454000, Henan, China

Abstract

The escalation of cloud services, driven by their accessibility, improved performance, and cost-effectiveness, has led cloud service providers to consistently seek methods to expedite job completion, thereby boosting profits and reducing energy consumption expenses. Despite developing numerous scheduling algorithms, many of these techniques address only a specific objective within the scheduling process. To efficiently achieve better optimization results for the cloud task scheduling problem, a novel approach, the Enhanced Whale Optimization Algorithm (EWOA), is proposed. EWOA integrates the WOA with the Lévy flight. The incorporation of Lévy flight is tailored to broaden the search space of WOA, expediting convergence with adaptive crossover. The EWOA model is simulated using the Cloudsim tool and evaluated under diverse test conditions. The effectiveness of EWOA is assessed by employing various parameters and comparing them with existing algorithms. The results demonstrate that EWOA outperforms other algorithms in resource utilization, energy consumption, and execution cost, establishing its superiority in addressing the complexities of multi-objective cloud task scheduling.

Keywords: Cloud computing, Scheduling, Lévy flight, Adaptive crossover, Resource allocation

Introduction

Cloud computing is a revolutionary computational framework that caters to a wide spectrum of users, from individuals to large enterprises [1]. The backbone of cloud computing consists of sophisticated and cost-intensive data centers (DCs) that significantly impact the financial sustainability of service providers [2]. Cloud service providers offer three primary categories of services, namely Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS), utilizing web service technology [3]. Cloud providers must balance offering their computational resources to meet users' Quality of Service (QoS) needs while efficiently managing their Total Cost of Ownership (TCO) in the increasingly competitive cloud market [4]. Virtualization technology plays a significant role in data centers to optimize resource allocation and decrease overall power consumption, an essential factor in TCO.

Additionally, power management strategies align with sustainability objectives. In a virtualized environment, a hypervisor allocates the resources of Physical Machines (PMs) among Virtual Machines (VMs). Inadequate resource allocation adversely impacts both resource utilization and the overall power consumption of datacenters [5].

Resource scheduling is a significant challenge in cloud computing as it involves allocating users' tasks across multiple VMs to meet specific objectives [6]. Given the limited number of VMs with varying capacities, an effective scheduling approach is required to allocate incoming tasks to suitable VMs efficiently. The scheduling problem is renowned for its complexity, falling within the category of NP-hard problems due to the constraints of resources and diverse user demands [7, 8]. Consequently, numerous heuristic and meta-heuristic strategies have been proposed to tackle this issue, each aiming at different objectives. Initially, heuristic methods were introduced to address the scheduling problem.

Many scheduling algorithms in cloud computing are designed to address specific objectives such as minimizing task completion time, maximizing resource utilization, or reducing energy consumption. For example, some algorithms focus solely on task scheduling without considering resource utilization optimization, while others prioritize resource allocation efficiency over task completion time. Additionally, there are scheduling approaches tailored specifically for certain application domains, such as scientific computing or multimedia processing, which may have unique requirements and constraints. Therefore, while there are indeed numerous scheduling algorithms available, each tends to target a specific aspect of the resource scheduling problem rather than providing a comprehensive solution that addresses all objectives simultaneously.

Existing scheduling algorithms often face limitations in effectively optimizing multiple objectives simultaneously and efficiently handling complex scheduling dilemmas. Many traditional heuristic methods rely heavily on predefined procedures, which may not adapt well to large-scale applications, leading to suboptimal solutions. Similarly, while meta-heuristic techniques have shown promise in addressing complex optimization problems, they can suffer from issues such as entrapment in local optima, reduced convergence, and high memory consumption if control parameters are not properly tuned. The EWOA introduces a novel approach to addressing the challenges inherent in cloud computing resource scheduling, targeting applications across diverse domains and task types. The EWOA aims to overcome these limitations by offering a hybrid approach that leverages the strengths of both heuristic and meta-heuristic methods, incorporating innovative strategies like Lévy flight to broaden the search space and enhance convergence, thereby enabling more effective and efficient scheduling in cloud computing environments.

However, most heuristic methods heavily rely on predefined procedures to enhance solution quality, particularly in scenarios involving large-scale applications. On the other hand, meta-heuristic techniques have demonstrated superior effectiveness in solving various complex optimization problems encountered in practical settings. Nevertheless, these algorithms have inherent weaknesses. For instance, improper tuning of control parameters can lead to entrapment in local optima, reduced convergence during iterative conditions, and high memory consumption. To address these limitations, a promising area of research involves hybridizing one or more heuristic and meta-heuristic methods. This approach capitalizes on the strengths of these techniques while mitigating their weaknesses. EWOA, an extension of the WOA, incorporates Lévy flight to tackle the multi-objective scheduling dilemmas efficiently. By integrating Lévy flight, the search space of WOA is broadened, accelerating convergence in conjunction with the adaptive crossover strategy.

The EWOA is specifically designed to tackle the multi-objective scheduling dilemmas efficiently by incorporating a Lévy flight mechanism. This integration of Lévy flight expands the search space of the WOA, allowing it to explore a wider range of solutions. By doing so, EWOA can simultaneously optimize multiple objectives, such as minimizing task completion time, maximizing resource utilization, and reducing energy consumption. Moreover, EWOA employs an adaptive crossover strategy, which further enhances its ability to converge towards optimal solutions for the complex scheduling problem. Therefore, EWOA stands out as a promising approach for addressing the diverse and interconnected objectives inherent in resource scheduling in cloud computing environments.

Related work

Zhang, et al. [9] effectively addressed uncertainty by transforming ambiguous variables into precisely defined interval parameters. The intricate scheduling process involved considering crucial factors such as makespan, task completion rate, load balancing, and scheduling cost. This comprehensive methodology led to the formulation of a novel algorithm termed Interval Multi-objective Cloud Task Scheduling Optimization (I-MCTSO). Deliberately designed, this algorithm accurately mirrors the intricacies of real-world cloud computing task scheduling scenarios. To implement this approach, a new Interval Multi-objective Evolutionary approach (InMaOEA) was devised. An inventive strategy to enhance convergence performance was integrated through a distinct interval credibility approach.

Additionally, augmenting population diversity was achieved by incorporating overlap and hyper-volume assessments alongside the interval congestion distance method. Empirical simulations provided compelling evidence supporting the high effectiveness and superiority of the InMaOEA algorithm in comparison to other existing algorithms. These proposed methodologies furnish a framework that equips decision-makers with a robust guideline for allocating cloud job scheduling, enabling well-informed decisions. These advancements signify a significant progression in cloud computing resource management, potentially elevating operational efficiency and effectiveness.

Alsaidy, et al. [10] introduced an innovative enhancement to the initialization process of the Particle Swarm Optimization (PSO) algorithm by integrating heuristic techniques. The strategic incorporation of the Minimum Completion Time (MCT) and Longest Job to Fastest Processor (LJFP) algorithms is implemented in the initialization phase of the PSO algorithm. This unique approach primarily aims to augment the overall efficiency of the PSO algorithm. A comprehensive evaluation of the formulated MCT-PSO and LJFP-PSO algorithms encompasses several crucial metrics. These metrics encompass the minimization of makespan, reduction in overall energy consumption, mitigation of imbalance, and decrease in total execution time. These metrics serve as pivotal benchmarks to assess the effectiveness of the proposed algorithms in the realm of task scheduling. Through extensive simulations, evidence is presented demonstrating the notable superiority and efficacy of the suggested MCT-PSO and LJFP-PSO approaches when compared to traditional PSO methods and other contemporary task scheduling algorithms. These findings underscore the potential of these enhancements to significantly enhance the optimization capabilities of task scheduling methods based on the PSO algorithm, thereby contributing significantly to the advancement of efficient and effective management of cloud computing resources.

Dubey and Sharma [11] introduced the Chemical Reaction PSO, a distinctive task scheduling method offering a hybrid approach for efficiently allocating multiple independent jobs among a collection of VMs. The method outlined in this study amalgamates the advantageous features of traditional chemical reaction optimization and particle swarm optimization, resulting in a unique synergy that leads to an optimal sequence for scheduling. This sequence facilitates job processing by considering both demand and deadline considerations, thereby augmenting the quality of outcomes across various parameters such as cost, energy consumption, and makespan. The methodology under scrutiny underwent comprehensive examination through extensive simulation experiments conducted using the CloudSim toolbox. The experimental outcomes underscore the effectiveness of the proposed algorithm. A rigorous assessment of the average execution time was conducted by contrasting studies involving varying quantities of VMs and jobs. Notably, the results manifest a substantial enhancement in the execution duration, ranging from 1 to 6%, with specific instances displaying even more substantial improvements exceeding 10%.

Furthermore, the makespan results exhibit noteworthy gains ranging from 5 to 12%, while the overall cost factor demonstrates enhancements of 2% to 10%. Additionally, there is a significant increase in the rate of energy consumption, ranging from 1 to 9%. Collectively, these results emphasize the effectiveness of the Chemical Reaction PSO algorithm in delivering measurable enhancements in critical performance measures.

Emami [12] introduced the Enhanced Sunflower Optimization (ESFO) algorithm, representing an innovative methodology aimed at bolstering the effectiveness of prevailing job scheduling techniques. The methodology delineated in this study highlights its capability to achieve optimal scheduling within polynomial time complexity. The proposed ESFO approach undergoes comprehensive scrutiny, subjected to a battery of task scheduling benchmarks to ascertain its strengths and limitations. The outcomes from simulation studies illustrate the superior performance of the ESFO algorithm in comparison to existing algorithms. Evidently, the algorithm demonstrates significant proficiency in optimizing task scheduling outcomes, notably evident in its robust performance across critical parameters such as energy usage and makespan. This technological advancement advances job scheduling methodologies, offering the potential for improved resource allocation and heightened system efficiency.

Gong, et al. [13] introduced the Enhanced Marine Predator Algorithm (EMPA) as a means to advance scheduling efficiency. The proposed methodology comprises several pivotal stages. It involves formulating a task scheduling model that considers both makespan and resource utilization. Each entity within the algorithm represents a potential outcome for task scheduling, with the primary aim of determining the most favorable scheduling solution. To bolster its performance, the EMPA integrates various components drawn from the Whale Optimization Algorithm (WOA), encompassing operator functions, nonlinear inertia weight coefficients, and the golden sine function. In simulation trials, the Evolutionary Multi-Objective Particle Algorithm (EMPA) undergoes an extensive comparative assessment against other established optimization algorithms, such as WOA, PSO, SCA, and GWO. These comparisons take place across diverse settings, considering distinct workloads in the GoCJ and synthetic datasets. The empirical evaluation highlights the advantages of the EMPA algorithm, showcasing notable strengths in resource utilization, degree of imbalance, and makespan. The findings presented in this study furnish empirical evidence supporting the efficacy of the Enhanced Marine Predator Algorithm in optimizing task scheduling outcomes. Consequently, these results make a valuable contribution to the field of scheduling approaches and have the potential to enhance resource management in various applications.

Hu, et al. [14] introduced a multi-objective scheduling algorithm termed MSITGO, which targets the optimization of three conflicting objectives: idle resource costs, energy consumption, and batch task completion time. Inspired by principles from Invasive Tumor Growth Optimization (ITGO), the MSITGO algorithm harnesses the inherent properties of tumor cell growth modeling by integrating Pareto optimum and packing problem models. This integration fosters a comprehensive and efficient exploration of potential solutions, broadening the spectrum of ideas and expediting the consensus-building process. Furthermore, the MSITGO framework meticulously encompasses the entirety of the task processing operation by bifurcating it into two distinct stages: machine assignment and timeslot allocation. This refined framework enhances job scheduling efficiency and mitigates improper allocations. In practical application, MSITGO undergoes empirical validation utilizing real cluster data obtained from Alibaba. The experimental results illustrate the superiority of MSITGO over existing techniques in addressing the multi-objective task scheduling problem. The framework demonstrates its ability to furnish more efficient solutions, signifying its potential to significantly contribute to optimizing task scheduling across diverse applications.

Problem statement and formulation

The allocation of users' tasks to available VMs within a cloud data center poses a significant challenge in cloud computing. This environment comprises a pool of homogeneous and heterogeneous resources where individual servers may concurrently operate multiple VMs. Through virtualization, clients can access scalable computing resources provided by these virtual environments. The scheduling mechanism is directed by the data center broker, who manages and oversees user task allocation. The schematic representation of the scheduling process can be observed in fig. 1. Initially, tasks are submitted by cloud users and stored within the task manager module. This module tracks incoming tasks and communicates relevant status updates to the respective users. Subsequently, the task manager forwards these task requests to the cloud scheduler. The cloud scheduler employs the proposed EWOA scheduling algorithm to assign incoming tasks to available VMs. This allocation is determined based on VM information and the task requisites obtained from the cloud information system.

The public cloud system model has multiple data centers to accommodate resource requirements. Assume a set of data centers labeled as $(dc_1, dc_2, \ldots, dc_p)$. Each data

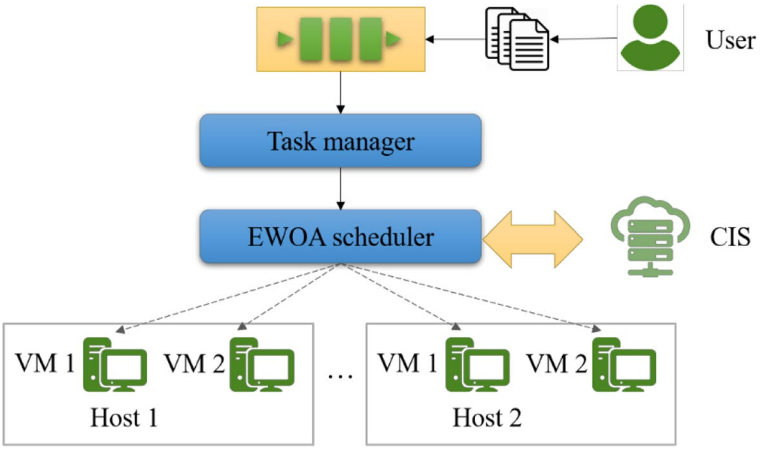

Fig. 1 Scheduling process

center consists of several Physical Hosts (PHs). For instance, data center dc_r has k PHs labeled as $(PH_{r1}, PH_{r2}, \ldots, PH_{rk})$. Each PH has specific characteristics, such as the number of cores, which determines its computing power measured in Million Instructions Per Second (MIPS). Each PH also has bandwidth, memory, storage capacity, and a VM Manager (VMM) installed. The VMM installed on PHs plays a crucial role in controlling and monitoring all VMs hosted on that particular physical host. It ensures efficient allocation and utilization of resources for the VMs running on the host. Each PH within a data center can accommodate a set of m VMs, denoted as $(VM_1, VM_2, \ldots, VM_m)$. Each VM has its specific configurations as follows:

The public cloud system model comprises multiple data centers designed to meet diverse resource demands. These data centers are denoted as $(dc_1, dc_2, \ldots, dc_p)$, house several Physical Hosts (PHs). For instance, data center dc_r comprises k PHs identified as $(PH_{r1}, PH_{r2}, \ldots, PH_{rk})$. Each PH is characterized by specific attributes, such as the number of cores, determining its computational capacity measured in Million Instructions Per Second (MIPS). Additionally, every PH possesses bandwidth, memory, and storage capacity and is equipped with a VMM responsible for overseeing and managing all VMs hosted on that specific physical host. The VMM ensures the effective allocation and utilization of resources for the VMs operating on the host. Within each data center, every PH is capable of accommodating a set of m VMs, identified as $(VM_1, VM_2, \ldots, VM_m)$. Each VM is configured with specific attributes, including:

- Main memory: Allocated to store data and execute applications within the VM.
- Storage: Capacity assigned to store VM-specific data and files.
- Processing power: Represented in MIPS, indicating the computing capability for executing instructions and performing tasks.
- A number of cores: Determines the VM's ability to handle simultaneous tasks and parallel processing by specifying the number of cores assigned to the VM.

Moreover, each VM comes with an associated price per hour, indicating the cost of utilizing that specific VM for every hour of usage. This pricing model enables users to pay for the resources they utilize based on the chosen VM configuration and the duration of their usage. In the realm of cloud computing, cloud consumers can submit their independent tasks to the service provider for processing without necessitating a deep understanding of the underlying system infrastructure. These tasks present varying requirements, particularly in terms of task duration and essential resources. In essence, users submit a total of n tasks denoted as $(t_1, t_2, \ldots, t_n)$ for processing. Each task is characterized by a unique length (l_i) measured in millions of instructions (MI). The scheduling process initiates by calculating the execution time of the i^{th} task on the j^{th} VM, as described in Eq. 1.

$$ET\left(l_i, vm_j\right) = \frac{l_i}{total_{MIPS}(vm_j)} \tag{1}$$

The scheduler then proceeds to assess the energy consumption associated with the VM while it carries out the assigned tasks, along with the cost related to task processing on that particular machine. The primary goal is to minimize the overall expense of task execution by pinpointing the VM that provides an optimal execution cost while fulfilling the specific requirements of each task. Given that each task is eligible for execution on any VM, and these machines possess varying processing capabilities, the execution time and costs for running the i^{th} task on different VMs are not uniform. Consequently, a multi-objective scheduling problem emerges, aiming to minimize execution time, energy consumption, and costs, all while optimizing resource utilization. Therefore, this research undertakes the challenge of addressing this multi-objective scheduling problem by implementing the proposed EWOA algorithm.

In cloud computing, two primary entities operate: service providers and cloud consumers. Service providers offer resources, while consumers rely on these resources to execute their tasks. Consumers prioritize application performance, seeking efficient and swift processing. In contrast, service providers focus on maximizing resource usage to optimize profitability. The objectives can be divided into consumer preferences (execution cost and schedule length) and provider preferences (resource utilization and energy consumption). Execution cost refers to the total price of running a user's application in the cloud. It is a quantifiable metric that provides a concrete measure of expenses. However, it is essential to express the cost concerning the utilized resources. Users aim to minimize both cost and schedule length. The computation of the execution cost can be outlined as follows:

$$EC = \sum_{j=1}^{m} ET_j \times price_j \tag{2}$$

In Eq. 2, ET_j represents the duration of task execution assigned to the j^{th} VM after the final task's execution. The schedule length is vital in evaluating scheduling quality and is defined as the maximum completion time among all submitted tasks or the completion time of the last processing VM. It is a crucial metric for assessing the efficiency of the scheduler. A lower schedule length signifies a more efficient scheduling strategy where tasks are allocated effectively to suitable resources. Conversely, a higher schedule length indicates a less effective scheduling strategy. Equation 3 can calculate the schedule length value.

$$SL = \max\left(\sum_{i=1}^{n} ET\left(l_i, vm_j\right)x_ij\right)\forall VM_j \tag{3}$$

Equation 3 captures the allocation decision variable, x_{ij}, representing whether the task l_i is assigned to the j^{th} VM. It is a binary variable, taking a value of 1 if l_i is allocated to VM j and 0 otherwise. Optimizing resource utilization is of paramount importance for cloud service providers. Their primary objective is to achieve maximum resource occupancy to improve profitability. With limited resources available, providers aim to maximize their usage. Equation 4 outlines the determination of average resource utilization.

$$averageRU = \frac{\sum_{i=1}^{m} ET_i}{o_1} \tag{4}$$

In Eq. 4, O_1 represents the minimum schedule length, reflecting the preferred service quality. In this scenario, effective resource utilization suggests the efficient use of available VMs in handling the submitted tasks. Energy consumption in data centers encompasses various elements, such as the CPU, network interfaces, and storage devices. Among these resources, the CPU is typically the most energy-consuming. When assessing the energy consumption of a VM, it is bifurcated into two states: energy consumption during idle periods and active states. Both the idle and active states of the VM are considered when calculating the total energy consumption. Energy used during idle periods is estimated to be roughly 60% of the energy consumed by a fully operational VM. The remaining 40% is the energy expended by the VM during active computations, which is contingent on the VM's computational speed, as depicted in Eq. 5.

$$EC = 10^{-8} \times \left(vm_{i_{mips}}\right)^2 \frac{J}{MI} andIE_i = 0.6 \times EC_i \tag{5}$$

Equation 5 features EC_i signifying the energy expended in the active state of the VM and IE_i indicating the energy utilized during the idle state. This allows the definition of the model's energy usage as denoted in Eq. 6.

$$TE_i = EC_i + IE_i \tag{6}$$

Proposed algorithm

The integration of Lévy flight and adaptive crossover in EWOA brings significant benefits to multi-objective optimization. Lévy flight, a stochastic search strategy inspired by the flight patterns of foraging animals, enables EWOA to explore a wider range of solutions by allowing for large, infrequent steps in the search space. This exploration mechanism helps prevent EWOA from getting stuck in local optima and facilitates the discovery of diverse and potentially superior solutions. Additionally, the adaptive crossover strategy enhances EWOA's ability to balance exploration and exploitation during the optimization process. By dynamically adjusting the crossover rate based on the progress of the optimization, EWOA can effectively adapt its search strategy to the changing characteristics of the problem landscape, leading to improved convergence and better overall performance in tackling multi-objective optimization challenges. Overall, the integration of Lévy flight and adaptive crossover empowers EWOA to effectively navigate the complex trade-offs inherent in multi-objective optimization problems, ultimately leading to more robust and efficient solutions.

The WOA is inspired by the foraging behavior of whales, particularly the hunting strategies of humpback whales [15]. This algorithm involves three foraging behaviors that imitate the actions of humpback whales: surrounding prey, bubble net attack, and randomly hunting prey. These behaviors are mathematically modeled to represent the hunting approach of the whales. Humpback whales exhibit the capability to locate nearby prey and position themselves considering the identified prey location as the optimal position among the group. As they close in on the prey, they continuously adjust their positions. In the context of WOA, the algorithm views the generated viable solutions as 'whales' and identifies the current best solution or local optimum as the best position for prey encircling. The algorithm employs an operator that simulates the act of encircling prey, represented in Eq. 7.

$$\overrightarrow{X}(t+1) = \overrightarrow{X}_{best}(t) - \overrightarrow{A}.|\overrightarrow{C}.\overrightarrow{X}_{best}(t) - \overrightarrow{X}(t)| \tag{7}$$

In Eq. 7, $\cdot$ indicates an element-wise multiplication, $\overrightarrow{X}_{best}(t)$ represents the best position of a whale in the current iteration t, X stands for the selected search whale, and $\left|\overrightarrow{C}.\overrightarrow{X}_{best}(t) - \overrightarrow{X}(t)\right|$ refers to the distance between $\overrightarrow{C}.\overrightarrow{X}_{best}(t)$ and $\overrightarrow{X}(t)$. The coefficient vectors $\overrightarrow{A}$ and $\overrightarrow{C}$ are dynamic variables, and their updates are determined by Eq. 8 and Eq. 9, respectively.

$$\overrightarrow{A} = 2 \times \overrightarrow{a} \times \overrightarrow{r} - \overrightarrow{a} \tag{8}$$

$$\overrightarrow{C} = 2 \times \overrightarrow{r} \tag{9}$$

The vector $\overrightarrow{a}$ will gradually decrease from 2 to 0, adhering to the formula max $2 - 2t/t_{max}$ with t_{max} as the maximum iteration count. The symbol $\overrightarrow{r}$ represents a random vector ranging between 0 and 1. This introduced random vector $\overrightarrow{r}$ constrains the $\overrightarrow{A}$ within the interval $[-\overrightarrow{a}, \overrightarrow{a}]$. It is important to note that the random vectors $\overrightarrow{A}$ and $\overrightarrow{C}$ assist the whales in updating their positions to achieve the optimal solution. Humpback whales utilize bubble nets to corral and capture prey near the water's surface. The mathematical representation of the spiral bubble net attack process is as follows:

$$\overrightarrow{X}(t+1) = \left|\overrightarrow{X}_{best}(t) - \overrightarrow{X}(t)\right|.e^{bl}\cos(2\pi l) + \overrightarrow{X}_{best}(t) \tag{10}$$

The parameter b is employed to describe the logarithmic spiral shape, while l denotes a randomly selected number within the range of [0, 1]. Humpback whales explore new target prey by randomly selecting a whale position and swimming towards it. The formula used in the WOA simulates this process for global search.

$$\overrightarrow{X}(t+1) = \overrightarrow{X}_{rand}(t) - \overrightarrow{A}.|\overrightarrow{C}.\overrightarrow{X}_{rand}(t) - \overrightarrow{X}(t)| \tag{11}$$

The choice of the three operators is governed by a random switch control parameter, p, within the range of [0,1]. The vector $\overrightarrow{A}$ determines the hunting method of the whale. Assuming that the whale has a 50% probability of selecting the bubble-net attacking method for updating their position during solution exploitation, the probability for choosing the operator to search for prey or encircle prey is additionally influenced by the adaptive variation of the vector $\overrightarrow{A}$. The mathematical model for the operator selection can be defined as follows:

$$\overrightarrow{X}(t{+}1) = \left\{ \begin{matrix} (\overrightarrow{X}_{best}(t) - \overrightarrow{A}.|\overrightarrow{C}.\overrightarrow{X}_{best}(t) - \overrightarrow{X}(t)|, \mathit{if}\ p < 0.5\ \mathit{and}\ |\overrightarrow{A}| < 1 \\ \overrightarrow{X}(t+1) = \overrightarrow{X}_{rand}(t) - \overrightarrow{A}.|\overrightarrow{C}.\overrightarrow{X}_{rand}(t) - \overrightarrow{X}(t)|, \mathit{if}\ p < 0.5\ \mathit{and}\ |\overrightarrow{A}| \geq 1 \\ |\overrightarrow{X}_{best}(t) - \overrightarrow{X}(t)|.e^{bl}\cos(2\pi l) + \overrightarrow{X}_{best}(t), \qquad \mathit{if}\ p\ \geq\ 0.5) \end{matrix} \right\} \tag{12}$$

WOA employs $\overrightarrow{A}$ as a switch governing the transition between exploration ($\left|\overrightarrow{A}\ \geq 1\right|$) and exploitation ($\left|\overrightarrow{A}\ < 1\right|$). Yet, the probability of exploration gradually diminishes with an increasing number of iterations since $\left|\overrightarrow{A}\right|$ diminishes overall as per its definition in Eq. 8. This could result in entrapment within local optima. In the standard WOA, Eq. 11 governs the updating of each individual's position based on another randomly selected individual within a confined range during the exploration phase, thereby restricting the exploration space. To broaden the search scope and reinforce global search capability, incorporating the random walk mechanism of Lévy flight into Eq. 11 results in the occasional inclusion of long-distance leaps to update positions. The global search improvement facilitated by Lévy flight, used to update the positions of humpback whales, can be expressed as follows:

$$\overrightarrow{X}(t+1) = \overrightarrow{X}_{rand}(t) + \alpha_0.\left|\overrightarrow{X}_{rand}(t) - \overrightarrow{X}(t)\right|.sign\left[rand - \frac{1}{2}\right] \oplus Levy(s) \tag{13}$$

The function $sing[rand - \frac{1}{2}]$ is a symbolic function that has three potential outputs: -1, 0, or 1. The parameter a_0 represents a step for the distance $\left|\overrightarrow{X}_{rand}(t) - \overrightarrow{X}(t)\right|$, and in this study, it is assigned a value of 0.05. *Levy(s)* refers to the Lévy distribution, which characterizes the non-Gaussian random process, and its distribution can be mathematically represented as follows:

$$L\acute{e}vy(s) \sim |s|^{-1-\beta}, \qquad 0 < \beta \leq 2 \tag{14}$$

The variable s represents the length of the Lévy flight's step, while β denotes the index. The value of s is determined by employing Manteca's algorithm, incorporating two normal distributions in accordance with the formula given below:

$$s = \frac{\mu}{|\nu|^{\frac{1}{\beta}}}, \mu \sim N\left(0, \sigma_\mu^2\right), \nu \sim N(0, \sigma_\nu^2) \tag{15}$$

In this study, the parameter β is fixed at 1.5, σ_υ is set to 1, and the calculation for σ_u is determined by Eq. 16.

$$\sigma_\mu = \left\{ \frac{\tau(1+\beta).\sin(\frac{\pi\beta}{2})}{(\beta.\tau[(1+\beta)/2^{(\beta-1)/2}]} \right\}^{\frac{1}{\beta}} \tag{16}$$

Table 1 Data center and host configurations

Cloud entity	Characteristic	Value
Datacenter	Number of users	1
	Number of hosts	2
	Number of data centers	2
Host	Bandwidth	10 GB
	RAM	2 GB
	Storage	1 TB

Table 2 VM configurations

Characteristic	Value
Size	100 MB
VMM	Xen
Bandwidth	0.5 Gb/S
MIPS	500–1500
Number of VMs	40–120

Results and discussion

Scheduling outcomes for a range of task quantities distributed across multiple VMs are outlined in this segment. The modified EWOA proposed in this study is contrasted against traditional WOA, Ant Colony Optimization (ACO), Harris Hawks Optimizer (HHO), and Genetic Algorithm (GA), using diverse performance metrics such as resource utilization, energy consumption, and execution cost. The simulations were conducted on a Windows 8 64-bit laptop equipped with 8 GB RAM. CloudSim 3.0.3, a widely used cloud environment simulation tool, was employed for these simulations. Specifications for hosts and data centers used in the simulation are presented in Table 1. Details regarding the properties of the VMs utilized in the simulation are outlined in Table 2.

Additionally, Table 3 illustrates the pricing structure for various Azure Bs-series, which is pertinent for cost considerations in the simulation. Table 4 provides a summary of the synthetic workload employed to assess the efficiency of the proposed EWOA algorithm. This synthetic workload is generated using a uniform distribution, ensuring an even distribution of tasks of varying sizes. The evaluation is centered on the High-Performance Computing Centre (HPC2N) workload, a recognized standard for evaluating distributed system performance. The table furnishes details of a realistic scenario, including task sizes and other critical parameters for evaluating the effectiveness of the WOA algorithm.

Figures 2 and 3 exhibit the results of resource utilization for various algorithms, EWOA, WOA, ACO, HHO, and GA, utilizing HPC2N workloads. The experiments were conducted using 40 or 80 VMs, with the task quantity ranging from 250 to 2000. The findings show that EWOA surpasses other algorithms in terms of resource utilization. This superiority in resource utilization can be attributed to EWOA's consideration of resource usage during task scheduling for appropriate VMs, resulting in the efficient

Table 3 Prices for VM instances

Cores	Storage	Memory	Prices ($/Hour)
2	16	8	0.08
4	32	16	0.16
8	64	32	0.33
12	96	48	0.49
16	128	64	0.66
20	160	80	0.83

Table 4 Tasks properties

Number of tasks	Tasks' lengths	Characteristic
250–2000	10,000–80000	independent

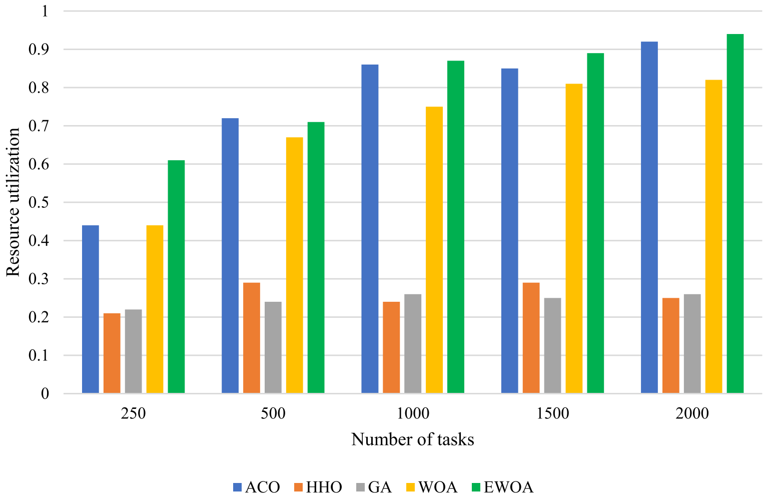

Fig. 2 Resource utilization for HPC2N workload with 40 VMs

utilization of VMs. As a consequence, EWOA significantly enhances overall performance by optimizing resource utilization, thereby establishing its effectiveness in inefficient task scheduling and resource allocation in comparison to WOA, ACO, HHO, and GA.

In figs. 4, 5 and 6, the algorithms are compared based on energy consumption. This metric is essential to minimize scheduling costs and durations, considering the relationship between scheduling energy, cost, and length. It is evident that as the task count increases, the energy consumption rises linearly for WOA, ACO, GA, and HHO algorithms. Contrarily, EWOA demonstrates a slower increase in energy consumption, indicating its efficiency in managing energy consumption despite a growing number of tasks. The flexibility of task count adaptation, distinguishing it from other approaches that overlook workload variations for resource allocation, contributes to the effectiveness of EWOA. Leveraging this trait, EWOA excels in assigning tasks with higher costs and durations to VMs with greater capacities. Hence, it can be concluded that EWOA

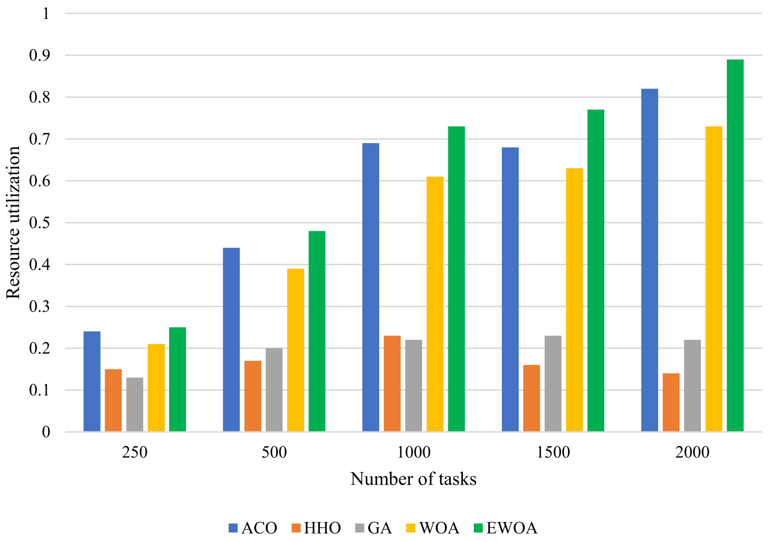

Fig. 3 Resource utilization for HPC2N workload with 80 VMs

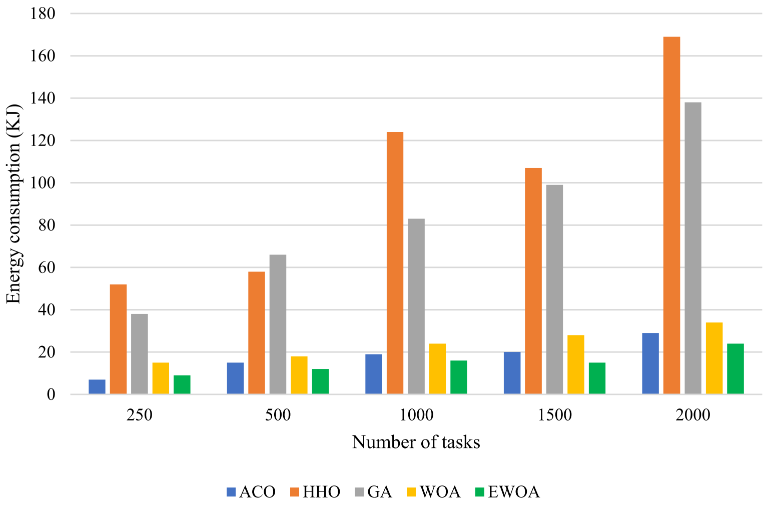

Fig. 4 Energy consumption for HPC2N workload with 40 VMs

excels in task assignment and resource allocation, facilitating the efficient utilization of higher-capacity VMs.

Figures 4, 5, and 6 compare algorithms in terms of execution cost for HPC2N and synthetic workloads. Execution costs can vary based on factors like VM type and task duration. Figures 4 and 5 evidence that EWOA surpasses the conventional WOA algorithm in both synthetic and HPC2N workloads across varying task numbers. For real tasks ranging from 250 to 2000, EWOA showcases an average reduction in execution costs of 9% to 41% compared to WOA. Similarly, for the synthetic workload and task numbers between 500 and 2000, the average reduction ranges from 7 to 57% in comparison to WOA. Thus, EWOA not only minimizes execution costs but consistently fulfills the goal of cost reduction across diverse scenarios, as evident in the mentioned figures.

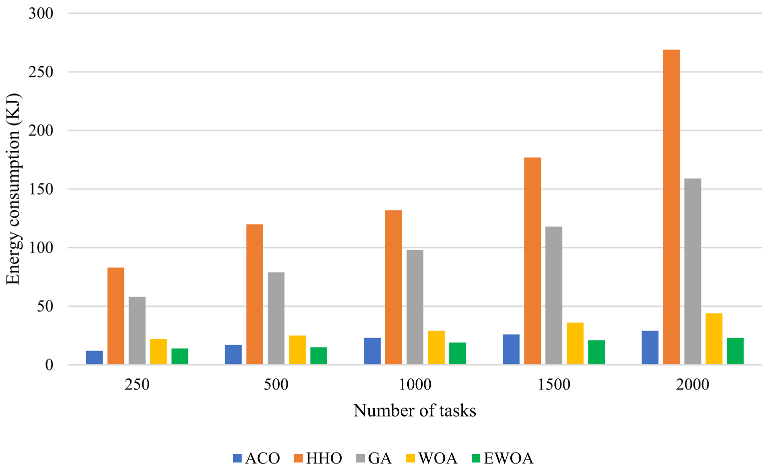

Fig. 5 Energy consumption for HPC2N workload with 80 VMs

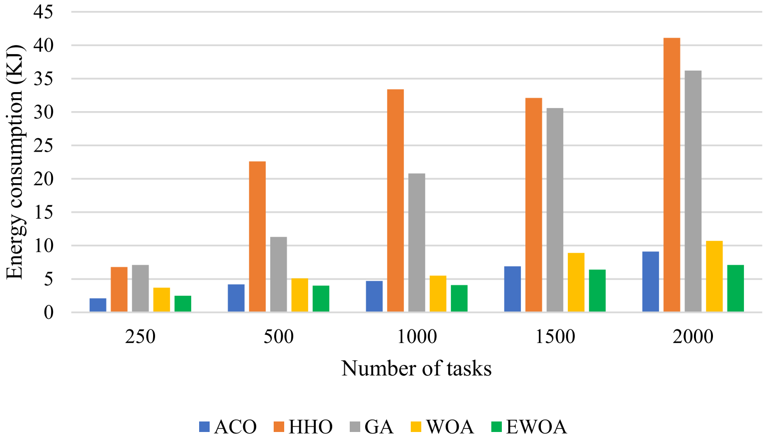

Fig. 6 Energy consumption for synthetic workload with 120 VMs

Furthermore, Wilcoxon's signed-rank test is a non-parametric statistical method used to compare two related samples. In the context of cloud computing resource scheduling, this test is applied to evaluate the performance of the proposed EWOA algorithm against other existing scheduling algorithms, such as WOA, ACO, GA, and HHO. Specifically, the test examines the differences in energy consumption and execution costs between EWOA and each of the other algorithms across multiple workload scenarios.

For each workload scenario, the energy consumption and execution costs obtained from running EWOA and the comparison algorithms are paired. The differences between the corresponding pairs are calculated and ranked according to their absolute values. Wilcoxon's signed-rank test then assesses whether these differences significantly deviate from zero, indicating a statistically significant improvement or degradation in performance.

By utilizing Wilcoxon's signed-rank test, the study ensures a rigorous statistical analysis that accounts for the paired nature of the comparisons and the non-normal distribution of data commonly encountered in cloud computing environments. This approach provides robust evidence regarding the generalizability and reliability of the proposed EWOA algorithm's performance improvements in terms of energy efficiency and cost reduction across a range of workload scenarios.

Morover, in this study, an ablation study was conducted to meticulously analyze the individual contributions of each component integrated into the Improved Whale Search Optimization (IWSO) algorithm, including the Lévy flight mechanism, adaptive crossover strategy, and other enhancements. By systematically isolating and evaluating the impact of each component on the overall performance of the algorithm, this approach provides invaluable insights into the effectiveness and significance of these enhancements. Such granular analysis enables a deeper understanding of how each element influences the algorithm's performance metrics, such as convergence speed, solution quality, and robustness to parameter variations. Consequently, the findings of this ablation study offer valuable guidance for future researchers seeking to harness these improvements in their own proposed methods. By elucidating the relative importance of each component, researchers can make informed decisions about which enhancements to prioritize or customize based on the specific requirements and characteristics of their optimization problems. Ultimately, this approach facilitates the development of more efficient and tailored optimization algorithms, driving advancements in various fields where optimization plays a critical role.

As results, EWOA's superiority over other algorithms is substantiated by quantitative evidence across multiple metrics. In the analysis of resource utilization depicted in figs. 2 and 3, EWOA outperforms competing algorithms including WOA, ACO, HHO, and GA when utilizing HPC2N workloads, even with task quantities ranging from 250 to 2000 and VMs ranging from 40 to 80. This demonstrates EWOA's effectiveness in maximizing resource utilization. Additionally, in figs. 4, 5 and 6, where energy consumption is evaluated, EWOA exhibits a slower increase in energy consumption compared to WOA, ACO, GA, and HHO as the task count rises. This indicates EWOA's efficiency in managing energy consumption, crucial for minimizing scheduling costs and durations. Furthermore, figs. 7, 8 and 9 illustrate EWOA's superiority over WOA in terms of execution cost across both synthetic and HPC2N workloads, underscoring its efficacy in optimizing scheduling outcomes across various scenarios and task numbers. Thus, EWOA emerges as a standout solution, delivering superior performance in resource utilization, energy efficiency, and execution cost reduction compared to existing algorithms.

The performance improvements achieved by EWOA compared to existing algorithms, such as WOA, ACO, GA, and HHO, are significant and multifaceted. Firstly, in terms of energy consumption, EWOA demonstrates superior efficiency, particularly as the task count increases. While other algorithms show a linear increase in energy consumption with the growing number of tasks, EWOA exhibits a slower rate of increase. This indicates its effectiveness in managing energy consumption even under higher task loads. The adaptability of EWOA to varying task counts, a feature absent in other approaches, allows it to allocate tasks efficiently to VMs with greater capacities, thereby minimizing energy consumption. Consequently, EWOA excels in task assignment and resource allocation, leading to the efficient utilization of higher-capacity VMs.

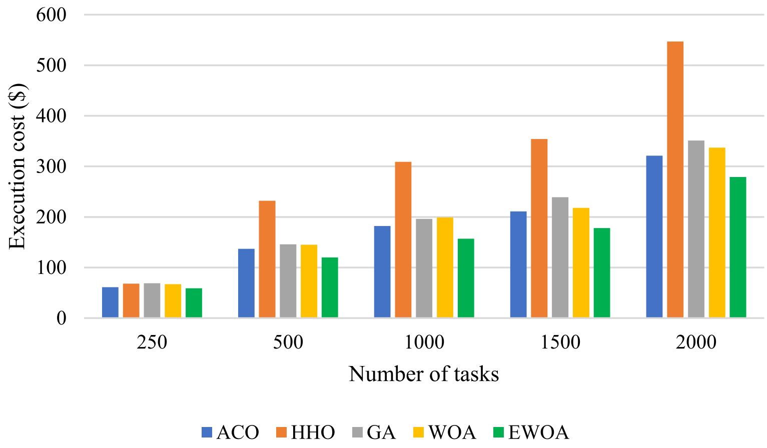

Fig. 7 Execution cost for HPC2N workload with 40 VMs

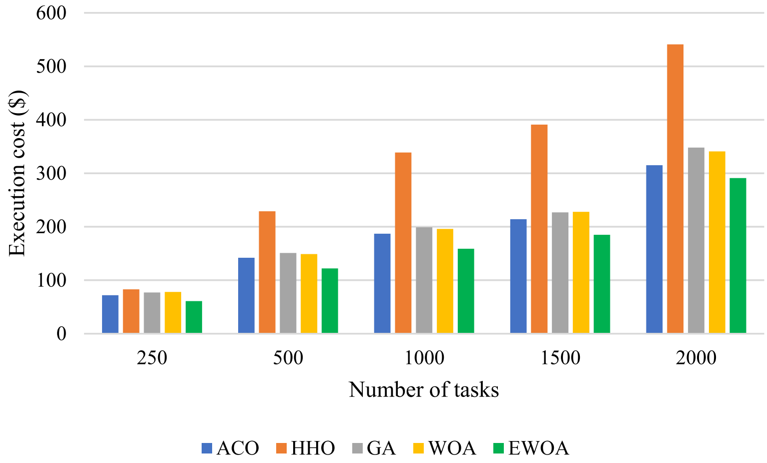

Fig. 8 Execution cost for HPC2N workload with 80 VMs

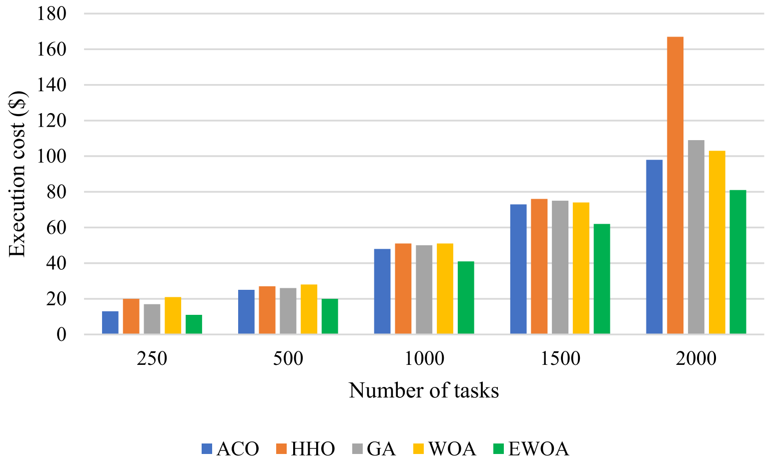

Fig. 9 Execution cost for synthetic workload with 120 VMs

Furthermore, when considering execution costs, EWOA outperforms the conventional WOA algorithm across both synthetic and HPC2N workloads, consistently delivering reductions in execution costs across diverse scenarios. For real tasks ranging from 250 to 2000 and synthetic workloads with task numbers between 500 and 2000, EWOA achieves average reductions in execution costs ranging from 9 to 57% compared to WOA. This demonstrates not only the effectiveness of EWOA in minimizing execution costs but also its consistency in achieving cost reduction goals across various workload scenarios.

Therefore, the performance improvements achieved by EWOA, as evidenced by reduced energy consumption and execution costs compared to existing algorithms like WOA, highlight its effectiveness in optimizing task assignment, resource allocation, and overall scheduling efficiency in cloud computing environments.

Conclusions

The ever-growing demand for cloud services necessitates efficient and multi-objective scheduling techniques to manage tasks effectively within cloud computing environments. This research introduced the EWOA as a novel approach to address the complexities of multi-objective task scheduling in the cloud. EWOA integrating Lévy flight strategy into the WOA demonstrated promising results in optimizing cloud task scheduling problems. By expanding the search space with Lévy flight, EWOA suggested superior performance in diverse test conditions simulated using the Cloudsim tool. The evaluation of EWOA against existing algorithms revealed its effectiveness in achieving enhanced resource utilization, reduced energy consumption, and minimized execution costs. These results underscored the potential of EWOA to address the intricate challenges of multi-objective cloud task scheduling. The findings of this research contribute significantly to the field of cloud computing by presenting a comprehensive, efficient, and innovative approach to multi-objective task scheduling. The EWOA offers promise for real-world applications, providing a means to optimize resource management in cloud environments while considering multiple conflicting objectives.

However, while EWOA demonstrates considerable advancements in multi-objective task scheduling, there are opportunities for further enhancements and exploration. Moreover, limitations in the managerial implications of fault detection within the study may arise from several factors. Firstly, while fault detection strategies may enhance system reliability and performance, their implementation often involves additional costs and resource allocation. Therefore, it’s essential to consider the cost–benefit analysis associated with deploying and maintaining fault detection mechanisms, especially for smaller-scale or budget-constrained organizations. Additionally, the effectiveness of fault detection methods may vary depending on the complexity and heterogeneity of cloud environments, as well as the specific types of faults encountered. Consequently, generalizing managerial implications across different cloud architectures and industries may pose challenges.

For future research, there are several directions that could be explored to address these limitations and further enhance the managerial implications of fault detection in cloud computing. Firstly, investigating the trade-offs between the costs of fault detection mechanisms and the potential savings achieved through improved system reliability and reduced downtime would provide valuable insights for decision-makers. Additinalltym, the future research could focus on refining the algorithm's parameters, scaling it for larger and more complex cloud environments, and investigating its adaptability to dynamic and real-time scenarios.

Acknowledgements
I would like to take this opportunity to acknowledge that there are no individuals or organizations that require acknowledgment for their contributions to this work.

Authors' contributions
All authors contributed to the study's conception and design. Data collection, simulation and analysis were performed by " Yanfeng ZHANG and Jiawei WANG". The first draft of the manuscript was written by " Yanfeng ZHANG and Jiawei WANG commented on previous versions of the manuscript. All authors have read and approved the manuscript.

Funding
This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Availability of data and materials
Data can be shared upon request.

Declarations

Competing interests
The authors declare no competing interests.

Received: 5 December 2023 Accepted: 29 April 2024
Published online: 20 May 2024

References

1. Pourghebleh B, Anvigh AA, Ramtin AR, Mohammadi B (2021) The importance of nature-inspired meta-heuristic algorithms for solving virtual machine consolidation problem in cloud environments. Cluster Comput 24(3):1–24
2. Hayyolalam V, Pourghebleh B, Kazem AAP, Ghaffari A (2019) Exploring the state-of-the-art service composition approaches in cloud manufacturing systems to enhance upcoming techniques. Int J Adv Manufact Technol 105(1–4):471–498
3. Hayyolalam V, Pourghebleh B, Chehrehzad MR, PourhajiKazem AA (2022) Single-objective service composition methods in cloud manufacturing systems: Recent techniques, classification, and future trends. Concurr Comput Pract Exp 34(5):6698
4. Bakaraniya P, Patel S, Singh P (2022) 5G Enabled Smart City Using Cloud Environment. In Predictive Analytics in Cloud, Fog, and Edge Computing: Perspectives and Practices of Blockchain, IoT, and 5G. Springer, Germany, p 199–226
5. Hanini M, Kafhali SE, Salah K (2019) Dynamic VM allocation and traffic control to manage QoS and energy consumption in cloud computing environment. Int J Comput Appl Technol 60(4):307–316
6. Nabi S, Ahmad M, Ibrahim M, Hamam H (2022) AdPSO: adaptive PSO-based task scheduling approach for cloud computing. Sensors 22(3):920
7. Yu Y, Su Y (2019) Cloud task scheduling algorithm based on three queues and dynamic priority. In 2019 IEEE International Conference on Power, Intelligent Computing and Systems (ICPICS), IEEE, United States of America, p 278–282
8. Minarolli D (2022) A Distributed Task Scheduling Approach for Cloud Computing Based on Ant Colony Optimization and Queue Load Information. International Conference on P2P, Parallel, Grid, Cloud and Internet Computing. Springer, pp 13–24
9. Zhang Z, Zhao M, Wang H, Cui Z, Zhang W (2022) An efficient interval many-objective evolutionary algorithm for cloud task scheduling problem under uncertainty. Inf Sci 583:56–72
10. Alsaidy SA, Abbood AD, Sahib MA (2022) Heuristic initialization of PSO task scheduling algorithm in cloud computing. J King Saud Univ-Comp Inform Sci 34(6):2370–2382
11. Dubey K, Sharma SC (2021) A novel multi-objective CR-PSO task scheduling algorithm with deadline constraint in cloud computing. Sustain Comput Inform Syst 32:100605
12. Emami H (2022) Cloud task scheduling using enhanced sunflower optimization algorithm. ICT Express 8(1):97–100
13. Gong R, Li D, Hong L, Xie N (2024) Task scheduling in cloud computing environment based on enhanced marine predator algorithm. Cluster Comput 27(1):1–15
14. Hu Q, Wu X, Dong S (2023) A two-stage multi-objective task scheduling framework based on invasive tumor growth optimization algorithm for cloud computing. J Grid Comput 21(2):31
15. Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67