Cognitive Computation (2023) 15:1497–1525
https://doi.org/10.1007/s12559-022-10099-z

Improving Whale Optimization Algorithm with Elite Strategy and Its Application to Engineering-Design and Cloud Task Scheduling Problems

Sanjoy Chakraborty[1,2] · Apu Kumar Saha[3] · Amit Chhabra[4]

Received: 8 February 2022 / Accepted: 9 December 2022 / Published online: 23 January 2023
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2023

Abstract
The whale optimization algorithm (WOA), a biologically inspired optimization technique, is known for its straightforward design and effectiveness. Despite many advantages, it has certain disadvantages, such as a limited exploration capacity and early convergence as a result of the minimal exploration of the search process. The WOA cannot bypass the local solution; consequently, the search is unbalanced. This study introduces a new variant of WOA, namely elite-based WOA (EBWOA), to address the inherent shortcomings of traditional WOA. Unlike the three phases used in the traditional WOA, only the encircling prey and bubble-net attack phases are applied in the new variant. Using the local elite method, exploration will be conducted with an encircling prey phase to ensure some exploitation during exploration. The choice between exploration and exploitation is achieved by introducing a new choice parameter. An inertia weight (ω_i) is used in both phases to scour the region. The EBWOA is used to evaluate twenty-five benchmark functions, IEEE CEC 2019 functions, and two design problems and compared to several fundamental techniques and WOA variants. In addition, the EBWOA is used to solve the practical cloud scheduling problem. Performance is compared against a variety of metaheuristics using real cloud workloads by running experiments on the standard CloudSim simulator. Comparing the numerical results of benchmark functions, IEEE CEC 2019 functions, statistical verification, and the solution generation speed of EBWOA confirmed the effectiveness of the proposed EBWOA approach. It has also shown a great improvement over baseline algorithms in creating efficient scheduling solutions by significantly reducing makespan time and energy consumption targets.

Keywords Metaheuristics · Whale optimization algorithm · Elite mechanism · Modified WOA · Cloud scheduling problem · Real-world application

Introduction

Efficient optimization techniques are essential to tackle countless real-world complex optimization applications across multiple technical and academic disciplines [1]. Traditional solution search strategies, such as enumeration search, branch and bound, and others struggle to address major optimization problems with acceptable convergence and accuracy in a reasonable time frame. In contrast to conventional methods, metaheuristic algorithms deliver better and optimal results in a reasonable amount of time. These algorithms are relatively easy to implement and are designed to provide global optima without falling into local optima regions. Metaheuristics are generally developed by taking inspiration from collaboration and indirect communication mechanisms inherited by natural and biological evolution, e.g., genetic programming (GP) [2]; differential evolution

✉ Apu Kumar Saha
apusaha_nita@yahoo.co.in

✉ Amit Chhabra
amit.cse@gndu.ac.in

[1] Department of Computer Science and Engineering, National Institute of Technology Agartala, Agartala, Tripura, India

[2] Department of Computer Science and Engineering, Iswar Chandra Vidyasagar College, Belonia, Tripura, India

[3] Department of Mathematics, National Institute of Technology Agartala, Agartala, Tripura 799046, India

[4] Department of Computer Engineering & Technology, Guru Nanak Dev University, Amritsar, Punjab 143005, India

(DE) [3]; immune network algorithm (INA) [4]; dendritic cell algorithm (DCA) [5]; physical sciences, such as equilibrium optimizer (EO) [6]; Henry's gas solubility optimization (HGSO) [7]; swarm-based behavior, e.g., Symbiotic Organism Search (SOS) [8] and whale optimization algorithm (WOA) [9]; and imitating problem-solving ways by humans such as teaching learning-based optimization (TLBO) [10].

The development of new metaheuristic algorithms with novel concepts is a common practice. Some of the recently published algorithms are discussed below. Abdollahzadeh et al. [11] designed the artificial gorilla troops optimizer (GTO) based on the social behavior of gorillas. Azizi [12] proposed the Atomic Orbital Search (AOS) algorithm inspired by quantum mechanics and the quantum-based atomic model concept. Hashim et al. [13] developed the Honey Badger Algorithm (HBA), motivated by honey badgers' sophisticated foraging behavior. Hasani Zade and Mansouri [14] used the prey-predator interaction of animals to develop the predator–prey optimization (PPO) algorithm, etc.

The metaheuristic algorithms demonstrate effective solutions compared to the traditional optimization techniques, especially when applied to highly nonlinear, multidimensional, and large-scale problems. Apart from the different working and inspiration mechanisms of these algorithms, they have a common way of searching the solution space using exploration and exploitation processes. The algorithms efficiently explore the entire search space with the maximum number of random moves in the exploration phase, while the exploitation phase is responsible for finding better solutions close to the current global best solution. Balancing the two phases of an algorithm is the most critical task and the key to success [15].

Besides the numerous advantages of metaheuristics, the no-free-lunch (NFL) [16] hypothesis argues that none of these algorithms can solve all kinds of problems. There is no guarantee that the algorithm that provides the best solutions to a particular set of problems will perform consistently better than another set of functions or problems. In addition, the choice of values for various metaheuristic parameters influences the final solution quality. In addition, the metaheuristic often struggles with inherent problems, such as were already in practice.

Several researchers have improved the WOA throughout the years to address its shortcomings. The following are some of WOA's recent enhancements and modifications: Kaur and Arora [17] developed the chaotic WOA (CWOA) by using chaotic maps to change WOA's parameters and speed up convergence. Sun et al. [18] proposed the modified WOA (MWOA), which used a non-linear dynamic strategy, Levy flight, and quadratic interpolation to avoid local optima and make solutions more accurate. Chen et al. [19] employed the Levy flight and a chaotic local search mechanism in the balanced WOA (BWOA) to avoid early convergence by enhancing the solution variety. Laskar et al. [20] incorporated WOA in particle swarm optimization (PSO). They came up with the hybrid whale–PSO algorithm (HWPSO) to avoid the stagnation effect. The authors also added the forced whale and capping phenomenon ideas to avoid local optima and speed up convergence. Bozorgi et al. [21] presented two WOA variations: IWOA and IWOA+. They have increased WOA's exploration capability by utilizing DE's superior exploration ability.

A DE-based WOA with chaotic map and opposition-based learning (DEWCO) was proposed by Elaziz et al. [22]. To increase the solution-finding speed of WOA, Yildiz [23] put forward the hybrid whale–Nelder–Mead algorithm (HWOANM), a hybrid WOA with the aid of the Nelder–Mead (NM) algorithm. Chakraborty et al. [24] devised a new version of the WOA algorithm called WOAmM. The authors changed the mutualism strategy of the SOS algorithm and then used it in WOA to balance the search process. Khadanga et al. [25] suggested a modified WOA (MWOA) by using the encircling prey phase and a bubble-net attacking phase to avoid trapping at local optima and used the algorithm in the load frequency controller design of a power system consisting of a PV grid and thermal generator. In the random spare reinforced WOA (RDWOA) [26], the authors used a double adaptive weight mechanism to improve the ability to explore at the beginning of the search and the ability to exploit at the end. In success history–based adaptive DE with WOA (SHADE-WOA) [27], the authors merged success history–based adaptive DE (SHADE) with updated WOA to create a hybrid algorithm. An information-sharing mechanism was used to assist the algorithms in efficiently exploring and exploiting the search space.

In [28], the authors introduced an improved version of WOA, called the Levy-flight-based WOA (LWOA); the levy-flight mechanism was incorporated with the WOA to enhance the ability to avoid premature convergence and boost global searchability. The method was used to solve the underwater image-matching problem in an unmanned underwater vehicle vision system. Kushwah et al. [29] suggested a new WOA variant with a roulette wheel selection strategy to enhance the convergence speed of WOA and applied it to the weight-updating technique of artificial neural networks. Fuqiang et al. [30] designed a bi-level WOA to solve the scheduling of risk management problems from IT projects.

Anitha et al. [31] designed a modified whale optimization algorithm (MWOA). The authors controlled the whale positions using the cosine function, and the whales' movements were controlled by applying correction factors

while updating their positions. The hunger search–based WOA (HSWOA) [32] was proposed by Chakraborty et al., integrating the concept of hunger into the WOA to minimize the demerits of WOA. An improved WOA (ImWOA) [33] was proposed by altering the exploration phase of the basic WOA and incorporating a new whale hunting concept, "Cooperative hunting," in the exploration phase of the WOA to balance the search activity. Lin et al. [34] developed the niching hybrid heuristic WOA (NHWOA); the niching strategy was used to diversify the solutions and control early convergence. Parameters of WOA were modified heuristically to encourage search agents' capacity for exploration during evolution. Avoidance of local solutions was ensured by executing a perturbation to the location of all the solutions. An enhanced WOA (EWOA) [35] was designed by Cao et al. to introduce improved dynamic opposition-based learning, and they converted the "Encircling Prey" phase into an adaptive phase. The modifications struck a balance between global and local searches in the algorithm.

Contrary to the previous research, only the local or global elite solution is used in this work, and an elite-based form of WOA (EBWOA) is proposed. Choosing a local elite solution from a group of random solutions allows the search process to shift the quest into different regions of the search domain. Thus, the algorithm explores the local best solution, and using inertia weight, the process examines the surrounding of the potential solution during both exploration and exploitation. The algorithm's convergence speed is accelerated by the use of the global best solution during the bubble-net attack phase. The following are the main contributions of the study:

- The encircling prey or the bubble-net attack phase is selected with the local best or the global solution, and an inertia weight using a traversing parameter Ω is utilized to accomplish exploration or exploitation. The search prey phase of basic WOA is eliminated to reduce run time.
- The numerical results of benchmark functions are compared with basic algorithms and WOA variants. The evaluated results of the IEEE CEC 2019 function set are compared with a list of modified variants.
- Performance is verified using statistical tests and a variety of analytics.
- EBWOA also solves two real engineering design problems and the classical cloud scheduling problem to schedule bag-of-tasks applications over cloud resources.

The rest of the work is structured as follows: "Whale Optimization Algorithm" presents the traditional WOA. "Proposed Elite-Based Whale Optimization Algorithm (EBWOA)" contains a complete discussion of the proposed algorithm. "Discussion of Numerical Results" compares the results of EBWOA with numerous basic and modified algorithms and two real engineering problems. "Analysis of EBWOA's Performance with Various Metrics" examines the performance of EBWOA using various performance measurement metrics. "Solving Cloud Scheduling Problem using EBWOA" describes the cloud scheduling problem and compares the evaluated results. "Conclusion" concludes the research carried out with concluding remarks.

Whale Optimization Algorithm

The WOA was developed to pursue the behavior of humpback whales, and it comes under swarm-based techniques. WOA, like other metaheuristic algorithms, begins with a set of parameters and a set of search agents that make up the underlying population. The search cycle alternates between local and global search phases, with each iteration relying on parameter selection to discover the best solution. After a certain number of cycles, the method will be over, and the best value for the objective function and the solution that goes with it will be the result. The different phases of the WOA are discussed below:

Exploration Phase

The most random motions possible are preferable during this algorithm phase to explore the search space efficiently. The whales move around in this phase, investigating the whole search area. This method's procedure can be stated numerically as follows:

$$\overline{Dt} = |C.Sol_r^{(i)} - Sol^{(i)}| \tag{1}$$

$$Sol^{(i+1)} = Sol_r^{(i)} - A'.\overline{Dt} \tag{2}$$

Sol represents a population solution, Sol_r is selected arbitrarily from the present population, i is the current value of the iteration, and $\overline{Dt}$ is the difference between $Sol_r^{(i)}$ and $Sol^{(i)}$ in Eqs. (1) and (2). The (.) operator represents component-by-component multiplication, and | | denotes the absolute value.

The following equations are used to calculate parameters A' and C:

$$A' = 2a^{\wedge} \times rnd - a^{\wedge} \tag{3}$$

$$C = 2 \times rnd \tag{4}$$

With the rising iteration value, the variable $a^{\wedge}$ traverses directly from 2 to 0, and rnd is an arbitrary value between [0, 1].

Exploitation Phase

Two hunting tactics used in WOA to accomplish local search are encircling the target prey and the bubble-net attacking approach. The following is a summary of these phases:

Encircling Prey Phase

The search agent with the best objective function value is considered the target solution during this phase. Other whales in the population are updated using the present best whale value. The updating method can be stated mathematically as follows:

$$Dt1 = \left| C \cdot Sol^{(i)}_{\text{best}} - Sol^{(i)} \right| \tag{5}$$

$$Sol^{(i+1)} = Sol^{(i)}_{\text{best}} - A' \cdot Dt1 \tag{6}$$

Sol_{best} is the best solution evaluated up to the present iteration. $Dt1$ is the distance between the best solution and the current solution.

Bubble-Net Phase

The whales move in a spiral path during the attack. The process is mathematically expressed as follows:

$$Dt2 = |Sol^{(i)}_{\text{best}} - Sol^{(i)}| \tag{7}$$

$$Sol^{(i+1)} = Dt2 \cdot e^{bl} \cdot \cos(2\pi l) + Sol^{(i)}_{\text{best}} \tag{8}$$

The spiral path is denoted by using the variable b in Eq. (8), with b having a constant value of 1, and the value of l is a random number calculated using the equation below:

$$l = \left(a^{\$} - 1\right) rnd + 1 \tag{9}$$

As the search process advances, the variable $a^{\$}$ changes between [-1,-2], and rnd is used to signify a random value inside [0, 1]. $Dt2$ is the distance between the best solution and the current solution.

The requirement for moving between the global and local search stages is the absolute value of A. If $|A|$ is less than 1, the algorithm runs Eq. (2) and then searches the search space. Otherwise, exploitation is done with Eq. (6) or Eq. (8). A probability value of 0.5 is used to confirm the choice between the exploitation strategies. The mathematical expression is as follows:

$$\begin{cases} Sol^{(i+1)} = Sol^{i}_{\text{best}} - A' \cdot Dt1 & if\ pr < 0.5 \\ Sol^{(i+1)} = Dt2 \cdot e^{bl} \cos(2\pi l) + Sol^{i}_{\text{best}} & if\ pr \geq 0.5 \end{cases} \tag{10}$$

where pr is a random positive value between 0 and 1.

Proposed Elite-Based Whale Optimization Algorithm (EBWOA)

A swarm-based metaheuristic optimization algorithm, the whale optimization algorithm, was designed by Mirjalili and Lewis [9], impersonating humpback whales' hunting behavior. WOA employs a basic yet effective mechanism with minimal control parameters [36]. WOA has a low convergence rate and cannot escape the best local solution due to the insufficient study of the search zone. This new variant is proposed as a means of overcoming these inherent limitations of WOA. The search prey phase and the prey circling or bubble-net attack approach have been used in basic WOA to conduct global and local searches. EBWOA, on the other hand, uses only modified encirclement and bubble-net attack strategies. Basic EBWOA no longer includes the search for prey phase of basic WOA. Modified equations for encircling prey and bubble-net attack phases are as follows:

Modified Encircling Prey Phase

$$Dt1 = \left| C \cdot l\_Sol^{(i)}_{\text{best}} - Sol^{(i)} \right| \tag{11}$$

$$Sol^{(i+1)} = \omega_i \cdot l\_Sol^{(i)}_{\text{best}} - A' \cdot Dt1 \tag{12}$$

In the above Eqns., l_Sol_{best} is the local best solution and ω_i is the inertia weight calculated as

$$\omega_i = 0.3 + 0.3 * rnd \tag{13}$$

Modified Bubble-Net Attack Phase

$$Dt2 = |Sol^{(i)}_{\text{best}} - Sol^{(i)}| \tag{14}$$

$$Sol^{(i+1)} = Dt2 \cdot e^{bl} \cdot \cos(2\pi l) + \omega_i \cdot Sol^{(i)}_{\text{best}} \tag{15}$$

In Eqs. (14) and (15), Sol_{best} is the global best solution.

EBWOA uses local and global elite solutions to update the solutions during the search process. A group solution is chosen. The solution with the minimum fitness value from the group is called the local elite solution, and the solution with the minimum fitness value in the entire population is used as the global elite solution. Choosing a local elite solution from a group of solutions allows the process to move to different regions of the search space. While exploring the search domain, updating solutions using the local elite value incrementally pushed the algorithm toward the best value. The inertia weight ω_i allows the process to exploit the nearby region effectively. The algorithm's convergence speed is accelerated by updating other solutions with the global best solution during the bubble-net attack phase. In the bubble-net phase, the area surrounding the global elite solution is searched using the global best solution and the inertia weight w. The selection parameter Ω is implemented to move between the phases. The parameter value progresses from 1 to 0 with the

increase of the iteration value. A probability value is used to compare the value, and if it is higher, the modified encircling prey phase is selected. If this is not the case, a modified bubble-net attack phase is used. Figure 1 displays the suggested EBWOA's pseudo-code, and Fig. 2 shows the algorithm's flowchart.

Discussion of Numerical Results

A total of twenty-five benchmark functions are used to assess the performance of the proposed EBWOA. The functions used in the study can be found in Appendix (Table 21). Functions F1–F13 are of the unimodal type. They have a global optimum and are used to assess the algorithm's local search capacity and convergence speed. Functions F14 and F25 are of the multimodal type. They have an abundance of local responses that grow exponentially as the size of the area grows. Solving these functions can test the algorithm's local search capacity and ability to overcome the local optima. The results of evaluating benchmark functions are compared with basic algorithms and modified WOA variants. EBWOA is also used to assess the capabilities of IEEE CEC 2019. The function set contains ten multimodal, non-separable functions. There are many local optima in most of these functions. The definition of these functions can be found in [1]. The results of the IEEE CEC 2019 functions are compared against a list of modified algorithms.

The system parameters include an Intel I3 processor, 8 GB of RAM, and MATLAB 2015a software. A population of a size of 30 with over 24,000 function evaluations is kept as termination criteria. Most WOA variants are judged using 500 or 1000 iterations as the termination criteria. Our algorithm records convergence in around 500 to 800 iterations for most functions. For this reason, we kept the end criteria of the program to 24,000 function evaluations, equivalent to 800 iterations. Because metaheuristic algorithms are stochastic, the comparison is based on the mean and standard deviation of findings from 30 independent runs. All the algorithms being compared have the same parameters as their original studies.

Comparison of Optimization Results of Benchmark Functions with Basic Algorithms

Evaluated results of the benchmark functions are compared with the tunicate swarm algorithm (TSA) [37], bald eagle search (BES) [38], WOA, symbiotic organisms search

Fig. 1 Pseudo-code of the proposed EBWOA algorithm

1. Initialize all the required parameters of WOA
2. Initialize the whale population
3. while (nfes<max_nfes) repeat the following
4. Find the best fitness and its corresponding solution Sol_{best}
5. Calculate Ω
6. For each solution in the whale population
7. Evaluate the weight ω_i
8. Select a group of random solutions
9. In the group find the random solution with minimum fitness (l_Sol_{best})
10. If (rand< Ω)
11. Update the present solution with Eqn. (12)
12. Else
13. Update the present solution with Eqn. (15)
14. End If
15. Check boundary condition for the updated solution
16. End For
17. End While
18. Return best fitness and Sol_{best}

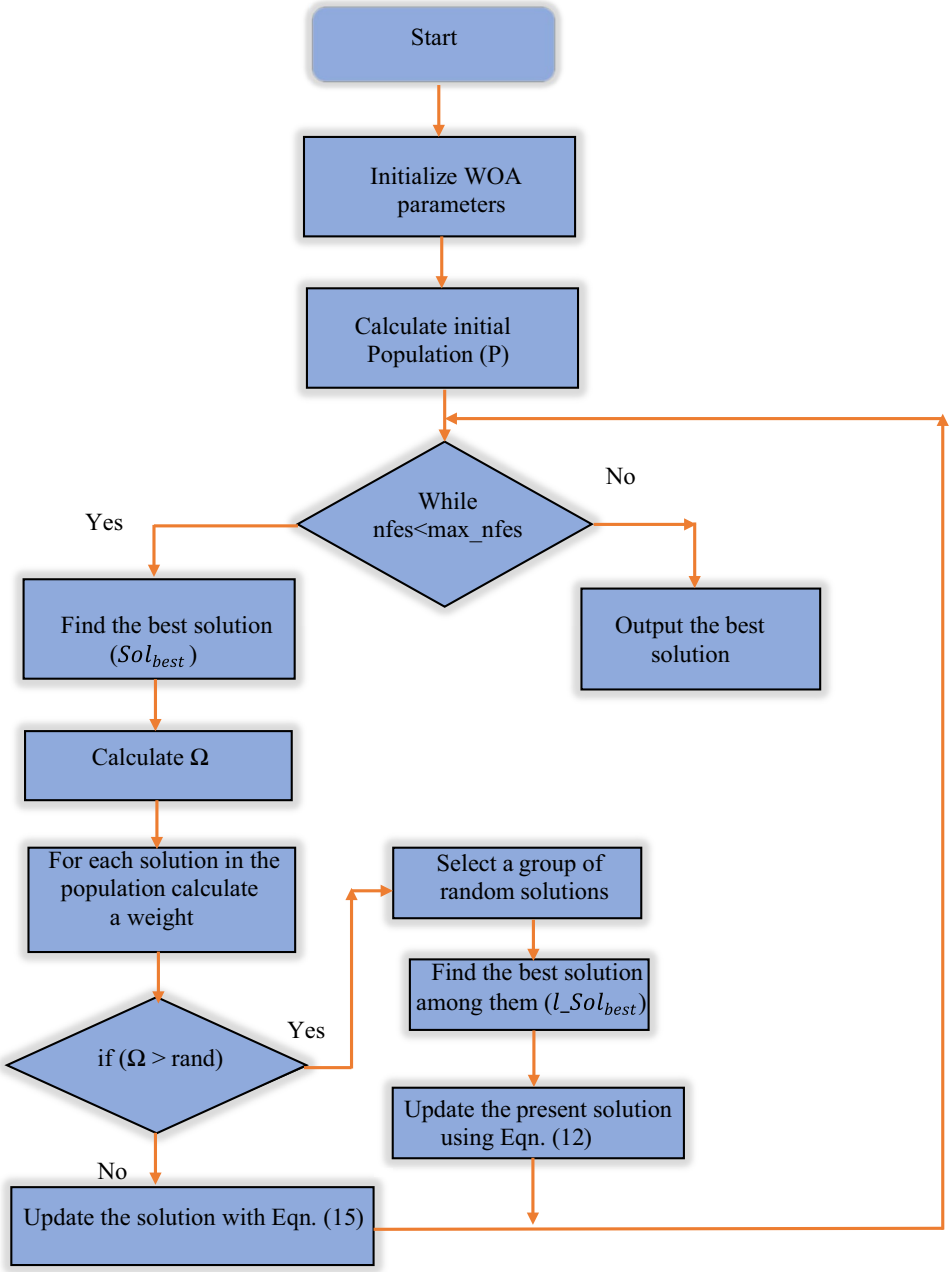

Fig. 2 Flowchart of the proposed EBWOA algorithm

(SOS), and teaching-learning-based optimization (TLBO). The comparison algorithms' parameters were set similarly to the values provided in the studies. Table 1 shows the mean and standard deviation (SD) values evaluated by all algorithms. EBWOA outperformed all other comparison algorithms on functions F1, F2, F3, F4, F6, F7, F8, F9, F10, F11, F12, F15, F17, F18, F20, F21, F22, F23, and F24. This shows that the algorithm can solve unimodal and multimodal problems. This is only possible if the algorithm's global and local search phases are balanced. The encircling prey algorithm phase is used for exploration using the best local solution. Choosing between exploration and exploitation allows the algorithm to move randomly between these phases. Since the local best value is chosen in the encircling prey phase, the search process gradually progresses to the optimal value.

For this reason, omitting the search prey stage does not adequately reflect the exploratory capability of the algorithm. Table 2 shows the pairwise comparison of EBWOA with different algorithms. EBWOA outperforms TSA, BES, WOA, SOS, and TLBO at 22, 20, 19, 21, and 22 functions. Identical results are obtained with the algorithms on 3, 5,

Table 1 Comparison of EBWOA results with the basic algorithms

Algorithm	F_1		F_2		F_3		F_4		F_5	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
EBWOA	**0.00**	0.00	**6.780E−313**	0.00	**0.00**	0.00	0.00	0.00	**0.00**	0.00
TSA	6.75E−39	1.12E−38	1.05E−23	1.76E−23	1.20E−08	5.33E−08	1.99E−02	3.42E−02	**0.00**	0.00
BES	2.39E−204	1.11E−201	5.40E−272	0.00	2.71E−216	0.00	5.72E−232	0.00	**0.00**	0.00
WOA	1.37E−137	7.53E−137	9.03E−102	4.91E−101	2.35E+04	9.99E+03	4.46E+01	2.92E+01	**0.00**	0.00
SOS	1.32E−54	1.62E−54	4.57E−28	4.75E−28	1.76E−15	4.68E−15	3.54E−22	2.88E−22	**0.00**	0.00
TLBO	8.22E−74	1.39E−73	1.44E−37	1.07E−37	8.63E−15	1.45E−14	2.81E−30	2.64E−30	**0.00**	0.00
Algorithm	F_6		F_7		F_8		F_9		F_{10}	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
EBWOA	**4.49E−05**	5.02E−05	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00
TSA	6.22E−03	2.89E−03	6.24E−19	1.70E−18	3.69E−31	1.19E−30	8.86E−09	1.38E−08	7.58E−86	2.36E−85
BES	5.45E−05	7.92E−05	7.29E−208	1.20E−111	3.19E−217	5.20E−206	0.00	0.00	0.00	0.00
WOA	1.81E−03	2.31E−03	5.01E+02	1.17E+02	6.90E−145	3.17E−144	1.01E−05	7.01E−06	4.33E−214	0.00
SOS	1.34E−03	7.17E−04	8.16E−09	9.43E−09	8.09E−47	1.38E−46	6.68E−11	2.87E−10	2.16E−124	8.14E−124
TLBO	1.15E−03	3.95E−04	2.06E−07	2.64E−07	1.58E−66	3.26E−66	3.16E−09	3.55E−09	9.84E−165	0.00
Algorithm	F_{11}		F_{12}		F_{13}		F_{14}		F_{15}	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
EBWOA	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**8.88E−16**	0.00
TSA	2.26E−32	6.78E−32	7.24E−41	1.38E−40	9.89E−124	4.97E−123	1.71E+02	4.20E+01	1.49E+00	1.54E+00
BES	2.91E−153	0.00	5.16E−122	0.00	**0.00**	0.00	**0.00**	0.00	4.26E−15	0.00
WOA	1.41E−141	7.60E−141	3.66E−144	2.00E−143	**0.00**	0.00	**0.00**	0.00	4.20E−15	2.63E−15
SOS	9.76E−48	2.28E−47	3.33E−57	8.24E−57	6.02E−86	2.17E−85	**0.00**	0.00	4.44E−15	0.00
TLBO	1.28E−67	1.51E−67	8.66E−77	1.27E−76	9.86E−110	5.39E−109	1.30E+01	8.20E+00	6.69E−15	1.74E−15
Algorithm	F_{16}		F_{17}		F_{18}		F_{19}		F_{20}	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
EBWOA	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00
TSA	6.93E−03	7.13E−03	2.40E−05	5.51E−05	8.57E−70	4.57E−69	1.58E+00	1.37E+00	3.47E−01	8.60E−02
BES	**0.00**	0.00	9.56E−09	4.17E−08	6.20E−261	4.67E−169	**0.00**	0.00	2.91E−172	3.20E−99
WOA	3.37E−03	1.85E−02	2.43E−12	7.10E−12	4.05E−199	0.00	**0.00**	0.00	1.07E−01	7.39E−02
SOS	**0.00**	0.00	4.02E−152	1.37E−151	1.96E−151	6.21E−151	**0.00**	0.00	9.99E−02	2.55E−08
TLBO	**0.00**	0.00	6.07E−68	3.33E−67	6.30E−201	0.00	**0.00**	0.00	1.34E−01	4.75E−02

Table 1 (continued)

Algorithm	F_{21}		F_{22}		F_{23}		F_{24}		F_{25}	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
EBWOA	**4.19E−04**	1.01E−04	**−9.19E+00**	2.30E+00	**−9.94E+00**	1.61E+00	−9.67E+00	2.15E+00	**0.00**	0.00
TSA	7.87E−03	9.76E−03	−5.83E+00	3.08E+00	−6.63E+00	3.49E+00	−6.51E+00	3.91E+00	**0.00**	0.00
BES	9.92E−03	1.03E−02	−8.91E+00	2.08E−02	−9.15E+00	1.39E+00	−9.54E+00	4.32E−04	**0.00**	0.00
WOA	7.32E−04	4.55E−04	−7.57E+00	2.56E+00	−8.29E+00	2.87E+00	−6.26E+00	3.00E+00	**0.00**	0.00
SOS	4.42E−04	1.68E−04	−9.13E+00	2.07E+00	−9.23E+00	9.70E−01	**−1.05E+01**	1.51E−02	**0.00**	0.00
TLBO	4.34E−04	2.21E−04	−9.18E+00	9.30E−01	−9.46E+00	2.00E+00	−9.28E+00	1.73E+00	**0.00**	0.00

Table 2 EBWOA results of pairwise comparison with the basic algorithms using Table 1 data

EBWOA	TSA	BES	WOA	SOS	TLBO
Superior to	22	20	19	20	22
Similar to	3	5	6	4	3
Inferior to	0	0	0	1	0

6, 4, and 3 functions. The numerical results and statistical analysis in Table 3 show that the EBWOA outperforms all other tested algorithms.

Comparison of Optimization Results of Benchmark Functions with Modified WOAs

The modified algorithms used for comparison in this study are ESSAWOA [39], WOAmM, and whale optimization algorithm modified with SOS and DE (m-SDWOA) [1], SHADE-WOA, and HSWOA. All comparison techniques used here are effective and recently published. All comparison methods use the same parameter settings proposed in the respective study. The evaluated results are shown in Table 4. The data in the table shows that EBWOA can solve both unimodal and multimodal functions. The local best solution improved the exploration ability of the algorithm during the encircling prey phase. The method uses the encircling prey phase to perform a survey using the locally best solution while being progressively exploited during exploration. By choosing the Ω value, the algorithm can alternate between exploration and exploitation at random. The random inertia weight (w) helps the search process to explore and exploit the nearby region. Being highly balanced, the algorithm can solve both types of functions effectively. Analyzing table data, EBWOA outperformed all other compared algorithms in eleven unimodal functions (F1, F2, F3, F4, F6, F7, F8, F9, F10, F11, and F12) out of thirteen evaluated functions. In function F5, EBWOA is superior only to SHADE-WOA; all other algorithms evaluate optimal results like EBWOA. EBWOA has obtained superior optimal outcomes in multimodal functions F18 and F21. In functions F14, F15, F17,

Table 3 Statistical test results using Friedman's rank test

Method	Rank sum	Average rank	Rank
EBWOA	38	1.52	1
TSA	135	5.40	6
BES	68	2.72	2
WOA	99.5	3.98	5
SOS	93	3.72	4
TLBO	91.5	3.66	3

Table 4 Comparison of EBWOA results with the WOA variants

Algorithm	F_1		F_2		F_3		F_4		F_5	
	Mean	**SD**	**Mean**	**SD**	**Mean**	**SD**	**Mean**	**SD**	**Mean**	**SD**
EBWOA	**0.00**	0.00	**6.780E-313**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00
ESSAWOA	9.26E-20	1.56E-19	1.17E-10	1.29E-10	2.54E-17	5.77E-17	5.43E-11	5.52E-11	**0.00**	0.00
WOAmM	1.15E-55	4.13E-55	1.01E-28	2.22E-28	3.97E-41	1.60E-40	1.04E-27	2.50E-27	**0.00**	0.00
m-SDWOA	3.94E-46	1.10E-45	6.89E-24	1.88E-23	2.16E-25	2.38E-25	9.48E-23	1.58E-22	**0.00**	0.00
SHADE-WOA	1.90E-47	1.02E-46	1.24E-32	6.55E-32	1.34E+02	8.51E+01	2.76E+00	1.40E+00	3.67E-01	1.07E+00
HSWOA	2.56E-319	0.00	5.10E-216	0.00	3.21E-198	0.00	5.61E-226	0.00	**0.00**	0.00
Algorithm	F_6		F_7		F_8		F_9		F_{10}	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
EBWOA	**4.49E−05**	5.02E−05	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00
ESSAWOA	9.31E−05	7.46E−05	1.22E−10	2.41E−10	4.58E−12	1.04E−11	1.12E−13	1.51E−13	2.23E−19	4.24E−19
WOAmM	8.36E−04	7.85E−04	2.39E−12	7.46E−12	5.31E−48	1.86E−47	1.07E−12	2.98E−12	8.74E−101	4.79E−100
m-SDWOA	4.61E−04	3.57E−04	1.35E−22	2.31E−22	1.64E−39	2.79E−39	1.38E−46	3.47E−46	1.22E−100	6.69E−100
SHADE-WOA	9.92E−03	8.59E−03	5.73E−130	3.13E−129	7.79E−39	4.25E−38	3.28E−25	1.06E−24	1.13E−50	6.19E−50
HSWOA	7.06E−05	7.51E−05	6.11E−209	4.11E−128	4.29E−281	7.19E−103	8.94E−39	4.89E−38	0.00	0.00
Algorithm	F_{11}		F_{12}		F_{13}		F_{14}		F_{15}	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
EBWOA	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**8.88E−16**	0.00
ESSAWOA	6.59E−13	1.71E−12	1.09E−14	1.92E−14	1.17E−24	2.21E−24	**0.00**	0.00	6.31E−11	8.10E−11
WOAmM	1.43E−48	5.31E−48	2.14E−57	1.14E−56	9.41E−95	5.15E−94	**0.00**	0.00	8.88E−16	0.00
m-SDWOA	3.58E−40	7.68E−40	3.42E−49	9.55E−49	8.41E−102	2.93E−101	2.19E+00	6.77E+00	3.73E−15	1.45E−15
SHADE-WOA	2.56E−39	1.40E−38	2.01E−50	1.10E−49	0.00	0.00	2.98E+01	1.76E+01	5.15E−15	2.70E−15
HSWOA	6.58E−189	5.89E−107	3.25E−222	1.98E−107	7.12E−301	2.10E−117	**0.00**	0.00	8.88E−16	0.00
Algorithm	F_{16}		F_{17}		F_{18}		F_{19}		F_{20}	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
EBWOA	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00
ESSAWOA	**0.00**	0.00	5.37E−13	5.16E−13	9.61E−69	3.59E−68	**0.00**	0.00	2.72E−11	2.51E−11
WOAmM	**0.00**	0.00	1.99E−09	4.85E−09	1.01E−160	5.52E−160	**0.00**	0.00	6.53E−14	3.44E−13
m-SDWOA	**0.00**	0.00	8.05E−54	4.41E−53	2.65E−140	1.42E−139	**0.00**	0.00	9.99E−02	1.10E−09
SHADE-WOA	1.15E−03	4.78E−03	2.02E−162	1.11E−161	5.56E−08	3.04E−07	**0.00**	0.00	4.07E−01	2.27E−01
HSWOA	**0.00**	0.00	**0.00**	0.00	1.09E−221	0.00	**0.00**	0.00	**0.00**	0.00

Table 4 (continued)

Algorithm	F_{21}		F_{22}		F_{23}		F_{24}		F_{25}	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
EBWOA	**4.19E−04**	1.01E−04	−9.19E+00	2.30E+00	−9.94E+00	1.61E+00	−9.67E+00	2.15E+00	**0.00**	0.00
ESSAWOA	1.97E−03	2.13E−03	−1.01E+01	5.12E−02	−1.03E+01	2.05E−01	**−1.05E+01**	1.29E−01	**0.00**	0.00
WOAmM	4.69E−04	2.16E−04	−6.24E+00	2.19E+00	−6.15E+00	2.16E+00	−6.75E+00	2.51E+00	**0.00**	0.00
m-SDWOA	4.68E−04	2.49E−04	−8.79E+00	2.29E+00	−1.00E+01	1.35E+00	−1.04E+01	9.87E−01	**0.00**	0.00
SHADE-WOA	1.13E−03	3.65E−03	**−1.02E+01**	6.68E−15	**−1.04E+01**	1.09E−15	−1.03E+01	1.48E+00	**0.00**	0.00
HSWOA	5.61E−04	1.62E−04	−2.67E+00	2.51E+00	−3.77E+00	2.72E+00	−5.18E+00	3.20E+00	**0.00**	0.00

Table 5 EBWOA results of pairwise comparison with the WOA variants using Table 3 data

EBWOA	ESSAWOA	WOAmM	m-SDWOA	SHADE-WOA	HSWOA
Superior to	18	19	19	19	17
Similar to	4	6	4	3	8
Inferior to	3	0	2	3	0

F19, F20, and F25, EBWOA evaluated the optimal outcome, though a few other algorithms also generated similar results. ESSAWOA and SHADE-WOA outperformed EBWOA in three multimodal functions (F22, F23, and F24).

Table 5 shows the pairwise comparison of numerical results with the WOA variants. From Table 5, it can be seen that EBWOA outperformed the compared algorithms in most features. A statistical comparison of the algorithms given in Table 6 also confirms the improved performance of EBWOA.

Comparison of Optimization Results of IEEE CEC 2019 Function Set with Modified Algorithms

Along with EBWOA, IEEE CEC 2019 functions are also evaluated using the methods, namely, modified whale optimization algorithm with population reduction (mWOAPR) [40], enhanced whale optimization algorithm (eWOA), enhanced whale optimization algorithm integrated with Salp Swarm Algorithm (ESSAWOA), self-adaptation butterfly optimization algorithm (SABOA) [41], sine cosine grey wolf optimizer (SC_GWO) [42], and improved sine cosine algorithm (ISCA) [43]. The optimal value of each function in the IEEE 2019 function set is 1. The results calculated by all algorithms are tabulated in Table 7. The function numbers F26 to F35 denote the IEEE CEC 2019 functions. Table 8 shows a pairwise comparison of results from EBWOA and other algorithms. According to tabular data, EBWOA

Table 6 Friedman's rank test with the WOA variants

Methods	Rank sum	Average rank	Rank
EBWOA	46.5	1.86	1
ESSAWOA	112.5	4.50	6
WOAmM	93.5	3.74	3
m-SDWOA	95	3.80	4
SHADE-WOA	107.5	4.30	5
HSWOA	70	2.80	2

Table 7 EBWOA results compared with the modified algorithms with the IEEE CEC 2019 function set

Algorithm	F_{26}		F_{27}		F_{28}		F_{29}		F_{30}	
	Mean	**SD**	**Mean**	**SD**	**Mean**	**SD**	**Mean**	**SD**	**Mean**	**SD**
EBWOA	**1.00**	0.00E+00	**5.00E+00**	0.00E+00	**4.73E+00**	1.08E+00	9.02E+01	1.22E+01	8.43E+01	1.74E+01
ImWOA	1.03E+07	1.05E+07	2.94E+03	1.50E+03	4.89E+00	7.71E−01	**3.00E+01**	1.02E+01	8.42E+00	5.53E+00
mWOAPR	3.25E+07	2.30E+07	9.07E+03	2.97E+03	6.64E+00	2.50E+00	4.33E+01	1.66E+01	4.02E+00	1.85E+00
eWOA	**1.00**	0.00E+00	5.13E+00	0.00E+00	6.33E+00	1.73E+00	7.75E+01	2.65E+01	6.54E+01	2.70E+01
ESSAWOA	1.00	3.39E−14	5.00E+00	1.04E−07	1.06E+01	1.15E+00	1.31E+02	3.16E+01	1.45E+02	4.99E+01
LWOA	1.21E+07	1.35E+07	7.60E+03	3.06E+03	4.75E+00	2.03E+00	5.86E+01	2.09E+01	**2.12E+00**	5.77E−01
SABOA	1.00	0.00E+00	5.00E+00	1.20E−02	1.25E+01	9.59E−01	1.39E+02	1.05E+01	1.82E+02	2.52E+01
SC_GWO	9.74E+01	4.74E+02	1.15E+01	2.86E+01	6.55E+03	3.30E+04	1.12E+02	3.46E+01	5.05E+01	2.99E+01
ISCA	**1.00**	0.00E+00	**5.00E+00**	0.00E+00	7.73E+00	1.46E+00	1.21E+02	1.13E+01	1.17E+02	2.61E+01
Algorithm	F_{31}		F_{32}		F_{33}		F_{34}		F_{35}	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
EBWOA	**1.04E+01**	1.48E+00	**1.87E+03**	2.05E+02	**4.95E+00**	1.68E−01	2.15E+00	7.47E−01	**2.15E+01**	1.08E−01
ImWOA	1.14E+01	1.13E+00	1.97E+03	2.85E+02	4.96E+00	2.63E−01	1.42E+00	1.15E−01	2.15E+01	1.10E−01
mWOAPR	1.17E+01	1.57E+00	1.90E+03	2.92E+02	4.98E+00	3.42E−01	1.39E+00	1.61E−01	2.16E+01	1.28E−01
eWOA	1.07E+01	1.06E+00	1.89E+03	3.31E+02	4.99E+00	2.93E−01	2.97E+00	9.61E−01	2.15E+01	1.49E−01
ESSAWOA	1.17E+01	1.48E+00	2.20E+03	2.21E+02	5.00E+00	1.34E−01	4.47E+00	8.66E−01	2.16E+01	1.35E−01
LWOA	1.26E+01	1.63E+00	1.93E+03	2.78E+02	4.96E+00	3.51E−01	**1.39E+00**	1.52E−01	2.18E+01	1.72E+00
SABOA	1.21E+01	9.66E−01	2.74E+03	2.77E+02	5.31E+00	2.29E−01	5.74E+00	9.17E−01	2.16E+01	1.32E−01
SC_GWO	1.79E+01	4.99E+00	3.94E+03	7.99E+02	5.99E+00	2.53E−01	3.63E+00	1.51E+00	2.27E+01	2.68E−01
ISCA	1.22E+01	8.80E−01	2.22E+03	2.06E+02	5.23E+00	1.34E−01	4.29E+00	5.82E−01	2.16E+01	1.13E−01

outperformed all other comparison algorithms in five functions. In function F26, eWOA, and ISCA and function F27, ISCA achieved similar optimal results with EBWOA. The data from Table 8 show that mWOAPR, eWOA, and SC-GWO can only outperform EBWOA on 3, 2, and 1 occasions, respectively. This confirms the superiority of the proposed EBWOA in solving complex optimization problems. The search process slowly proceeds to the optimal solution by checking the surrounding area for the local or global best solution. All these newly incorporated properties made the algorithm efficient. The statistical analysis results in Table 9 further support the dominance of EBWOA.

Comparison of Design Concepts of WOA Variants Used for Comparison and EBWOA

EBWOA is compared with a total of nine WOA variants. ESSAWOA, WOAmM, m-SDWOA, SHADE-WOA, and HSWOA are compared using the classical benchmark functions, whereas LWOA, mWOAPR, eWOA, and ImWOA are compared using IEEE CEC 2019 functions. In LWOA, the "Levy flight" mechanism was used with WOA to increase diversity in the solution and skip the local solution. The idea of a hybrid algorithm was used to create ESSA-WOA. Two algorithms, the Salp Warm Algorithm (SSA) and WOA, were merged to develop it. Firstly, SSA was modified with a non-linear parameter to strengthen the convergence function of SSA; then, it was merged with WOA. A lens opposition-based learning strategy was used in the algorithm to amplify the diversity in the solution. The mutualism phase of symbiotic organisms search (SOS) was modified and used in WOA to increase solution diversity; the new method was named WOAmM. In m-SDWOA, a modified mutualism phase and DE mutation strategy were used to enhance the exploration capacity of WOA. The commensalism phase of the SOS algorithm was used to increase solution accuracy. While making SHADE-WOA, SHADE and WOA were combined with a way to share information and a new way to hunt called "cooperative hunting."

Table 8 EBWOA results of pairwise comparison with the modified algorithms using Table 7 data

EBWOA	ImWOA	mWOAPR	eWOA	ESSAWOA	LWOA	SABOA	SC_GWO	ISCA
Superior to	7	7	7	10	7	10	9	8
Similar to	0	0	1	0	0	0	0	2
Inferior to	3	3	2	0	3	0	1	0

Table 9 Friedman's rank test with the modified algorithms

Methods	Rank sum	Average rank	Rank	P value	Remark
EBWOA	26.5	2.65	1	0.001	*P*-value (0.001 < 0.01) => Ho is rejected at a 1% significance level, meaning there is a significant difference in the performance of different algorithms at a 1% significance level
ImWOA	36.5	3.65	2		
mWOAPR	45.5	4.55	4		
eWOA	37	3.7	3		
ESSAWOA	58.5	5.85	6		
LWOA	46	4.6	5		
SABOA	68	6.8	8		
SC_GWO	73	7.3	9		
ISCA	59	5.9	7		

The concept of hunger from the algorithm Hunger Games Search (HGS) was introduced in WOA to develop HSWOA. mWOAPR was proposed by introducing random initialization of the solution in the "Search for Prey" phase of WOA. Moreover, the values of parameters "A" and "C" were modified to explore in the beginning and exploit later in the search. Population reduction was employed to make the convergence faster. Another variant of WOA, namely eWOA, was proposed by modifying the parameters "*A*" and "*C*" and introducing a random movement while exploring to lessen the computational burden. An exhaustive search near the potential solution was confirmed by employing an inertia weight. ImWOA is a recent variant of WOA that was designed by modifying the random solution selection process of the "Search Prey" phase in WOA. The other modifications in the algorithm include incorporating "cooperative hunting" to exploit easily and dividing total iterations into two halves, one for exploration and the other for exploitation. Unlike all the WOA variants, EBWOA uses local and global solutions for exploration and exploitation. The local best solution is a randomly selected solution from the group of cluster best solutions. In EBWOA, the "Search for Prey" phase used in WOA for exploration is omitted; instead, exploration is confirmed with the "Encircling Prey" phase. Exploration is preferred in the algorithm as exploration and exploitation are performed with either local or global solutions.

Real-World Engineering Problem

The gear train design problem, a real-world, unconstrained engineering problem, is resolved using EBWOA. "Gear Train Design" presents a description of the problem and an analysis of the evaluation results.

Gear Train Design

Sandgren [44] presented this design challenge, which is unconstrained in nature. There are four choice variables, y^1, y^2, y^3, and y^4, which represent the number of teeth in each gear wheel. All variables fall inside the range [12—60] and are positive integers. The angular velocity of the output shaft and the ratio of the input shaft were used to define the gear ratio for decreasing a gear train. The objective of this design challenge was to reduce the cost of the gear ratio to as close to 1/6.931 as possible. This problem's mathematical formulation is given below.

Objective Function

$$Minf(y) = \left[\left(\frac{1}{6.931}\right) - \left(y^3 y^4 / y^1 y^4\right)^2\right] \tag{16}$$

Subject To

$$12 \leq y^p \leq 60,\ p = 1, 2, \ldots, 4.$$

Analysis of Outcome Calculated results from EBWOA are compared to four basic versions of the metaheuristic and six WOA variants. Table 10 contains the results evaluated by the proposed algorithm and the algorithms used for comparison. EBWOA and SHADE-WOA achieved the optimal result, and their evaluated results are similar. The component algorithm of EBWOA, i.e., WOA produced the worst result on this problem. This authenticates the extension of WOA.

Table 10 Evaluated results of the gear train design problem

Method	Mean	SD	Best
EBWOA	**2.7755E−17**	0	2.7755E−17
WOA	6.2485E+02	2.3126E−13	6.2485E+02
SOS	6.3990E−06	6.4998E−06	2.7755E−17
TLBO	9.6440E−07	2.1258E−06	2.7755E−17
TSA	4.2423E−06	4.6445E−06	1.4557E−07
LWOA	1.0709E−06	1.4947E−06	4.5467E−08
WOAmM	5.5275E−08	1.3886E−07	5.0281E−10
m-SDWOA	4.5835E−08	2.5084E−07	2.7755E−17
SHADE-WOA	**2.7755E−17**	0	2.7755E−17
HSWOA	3.0080E−05	3.3073E−05	1.9838E−07
ImWOA	9.0754E−09	2.4560E−08	4.9821E−14

Three-Bar Truss Design

The issue involves minimizing the volume of a three-bar truss that is statically loaded while meeting three limitations on stress, deflection, and buckling. To change the sectional areas, this problem has to optimize two variables (x^1 and x^2). The search space for this topic is challenging and restricted. The following is the mathematical formulation for this issue:

$$\vec{x} = \{x^1, x^2\}$$

Objective Function

$$Min.f(x) = L\left\{x^2 + 2\sqrt{2}x^1\right\}, \tag{17}$$

Subject to

$$h_1(x) = \frac{x^2}{2x^2x^1 + \sqrt{2}\left(x^1\right)^2}P - \sigma \leq 0,$$

$$h_2(x) = \frac{x^2 + \sqrt{2}x^1}{2x^2x^1 + \sqrt{2}\left(x^1\right)^2}P - \sigma \leq 0,$$

$$h_3(x) = \frac{1}{x^1 + \sqrt{2}x^2}P - \sigma \leq 0,$$

where

$$0 \leq x^1, x^2 \leq 1, \text{ and}$$
$$P = 2,\ L = 100\ \&\ \sigma = 2.$$

Analysis of Outcome The optimal solution for this problem is 2.6389584338E+02. Table 11 shows the evaluated results. SOS, TLBO, m-SDWOA, SHADE-WOA, and ImWOA are the methods whose results are similar. However, among them, the standard deviation of EBWOA is the minimum. It reflects the consistency of the algorithm. Therefore, EBWOA has emerged as the best method among comparative methods.

Table 11 Evaluated results of three bar truss design problems

Method	Mean	SD	Best
EBWOA	**2.6389E+02**	2.5824E−16	2.6389E+02
WOA	NA	NA	NA
SOS	2.6389E+02	2.3918E−04	2.6389E+02
TLBO	2.6389E+02	1.0298E−04	2.6389E+02
TSA	2.6390E+02	8.0176E−03	2.6389E+02
LWOA	2.6399E+02	6.1201E−02	2.6390E+02
WOAmM	2.6401E+02	1.5084E−01	2.6389E+02
m-SDWOA	2.6389E+02	8.7416E−09	2.6389E+02
SHADE-WOA	2.6389E+02	1.0555E−14	2.6389E+02
HSWOA	2.7312E+02	5.5345E+00	2.6503E+02
ImWOA	2.6399E+02	1.8722E−01	2.6390E+02

Analysis of EBWOA's Performance with Various Metrics

In this section, the solution-finding speed of the proposed method, the time needed to search for the optimal solution, the exploration with the exploitability of the algorithm, and the performance index are analyzed.

Convergence Study

The algorithm's ability to find solutions quickly is tested using the convergence curve. This section compared the solution-finding speed of the proposed algorithm with its segment WOA. The curves are plotted with a population size of 30, and the algorithms determine the best fitness value for a single function with a termination condition of 100 iterations. Figure 3 shows the comparison curves of some randomly chosen fundamental functions of the unimodal type, the multimodal types, and the IEEE CEC 2019 functions. The figure's first six curves (a–f) are drawn using the benchmark functions, and the curves from (g–i) are generated using IEEE CEC 2019 functions. In each curve in the figure, EBWOA converges much faster than WOA. This means that WOA's search speed has been increased after the modification.

Runtime Analysis

Run time is the time taken by an algorithm to execute and produce the output. Here, we have evaluated the execution time by assessing the first function from the IEEE CEC 2019 function set, i.e., F_{26} in this study. The execution time of all the compared algorithms is given in Table 12. The table data reveals that EBWOA takes slightly greater time for execution than WOA. Similarly, SOS and TLBO are also faster than EBWOA. However, EBWOA takes less time than BES and TSA. Among the seven WOA variants used for runtime comparison, only two methods, m-SDWOA and mWOAPR, have less execution time than WOA. But analysis of the numerical outcomes already ensured that the performance of EBWOA is far better than the algorithms WOA, SOS, TLBO, m-SDWOA, and mWOAPR. Therefore, considering the high performance of EBWOA, a slight increase in run time compared to the component algorithm is acceptable.

Analysis of Exploration with Exploitation Capacity

Exploration and exploitation are the two basic phases of an optimization algorithm. The distance between the solutions

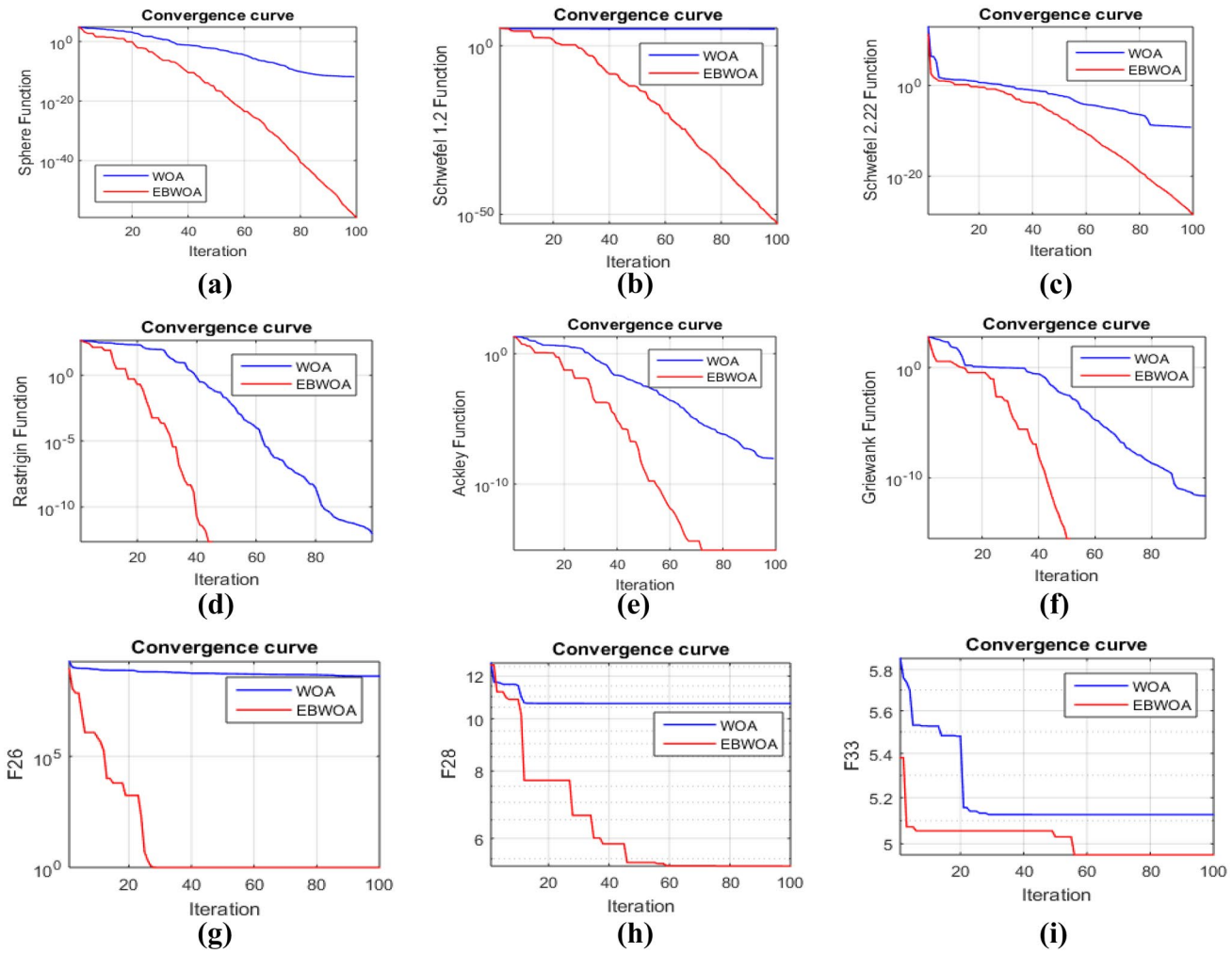

Fig. 3 Comparison of convergence curves of EBWOA with WOA

grows during exploration but reduces during exploitation. Diversity measurement is looked at and defined to determine how far apart search agents are getting closer or farther apart.

Table 12 Comparison of Run time with basic and WOA variants

Method	Mean run time
EBWOA	2.5264E+01
WOA	2.4611E+01
SOS	1.8875E+01
TLBO	1.6699E+01
BES	2.8735E+01
TSA	2.6177E+01
LWOA	1.4806E+02
WOAmM	2.8518E+01
m-SDWOA	1.5963E+01
SHADE-WOA	2.5286E+01
HSWOA	6.1163E+01
mWOAPR	1.3218E+01
ImWOA	2.6653E+01

$$div^j = \frac{1}{n}\sum_{i=1}^{n}\left|\text{median}\left(Sol^j\right) - Sol_i^j\right| \tag{18}$$

$$div = \frac{1}{dim}\sum_{j=1}^{dim} div^j \tag{19}$$

n and dim stand for the number of search agents and design variables, respectively. Sol_i^j is the dimension j of the ith search agent, and Sol^j is the median of the population for that dimension. div^j is the diversity in each dimension, and mathematically, it is defined as the distance between each search agent's jth dimension and the dimension's median. The variety of the entire population (div) is then determined by averaging each div.

$$\text{exploration percentage} = \left(\frac{div}{div_{\text{maxi}}}\right) \times 100 \tag{20}$$

$$\text{exploitation percentage} = \left(\frac{\left|div - div_{\text{maxi}}\right|}{div_{\text{maxi}}}\right) \times 100 \tag{21}$$

where "div_{maxi}" is the maximum diversity value attained over the entire optimization process. The exploration percentage links the diversity in each iteration to the largest variety found during the search. The exploitation level, measured by the exploitation percentage, is the difference between the maximum diversity and the diversity of an iteration at the moment. The concentration of search agents causes this difference.

Figures 4 and 5 represent the exploration and exploitation graphs showing their percentage for EBWOA and WOA, respectively, on six random benchmark functions. In both the figures, diagrams (a), (b), and (c) depict the exploration and exploitation of unimodal functions, whereas (d), (e), and (f) show the graphs obtained by evaluating three multimodal functions. A comparison of diagrams in both the figures reveals that the exploration and exploitation ability of EBWOA is more balanced than that of WOA in function types, unimodal and multimodal.

Performance Index Evaluation

The performance index (PI) of EBWOA is evaluated in terms of an increase or decrease in performance. Performance upsurge or reduction of an algorithm is calculated using the below-given formula.

$$PI(\%) = \frac{\text{Performance (other algorithm)} - \text{Performance (EBWOA)}}{\text{Performance (EBWOA)}} \times 100\% \tag{22}$$

The performance of EBWOA is compared to the modified methods using the evaluated outcomes of the IEEE CEC 2019 function set which are given in Table 7. Table 13 holds the function-wise comparison data. The positive value in the table indicates an increase, and the negative value specifies a decrease in the performance of EBWOA on that particular function compared to the algorithm concerned. The value of 0.00 designates no improvement in the performance of EBWOA on that function compared to the specific algorithm.

Solving Cloud Scheduling Problem using EBWOA

In this section, the proposed EBWOA strategy is applied to the classical NP-hard cloud scheduling problem for executing multiple independent bag-of-tasks (BoT) applications over virtual machines (VMs) of a cloud computing system [45, 46]. Each BoT application consists of several independent tasks requiring an equal number of processing elements for execution [47, 48]. The next sub-section briefly describes problem objectives, fitness functions, workloads, experimental setup, results, and analysis. In

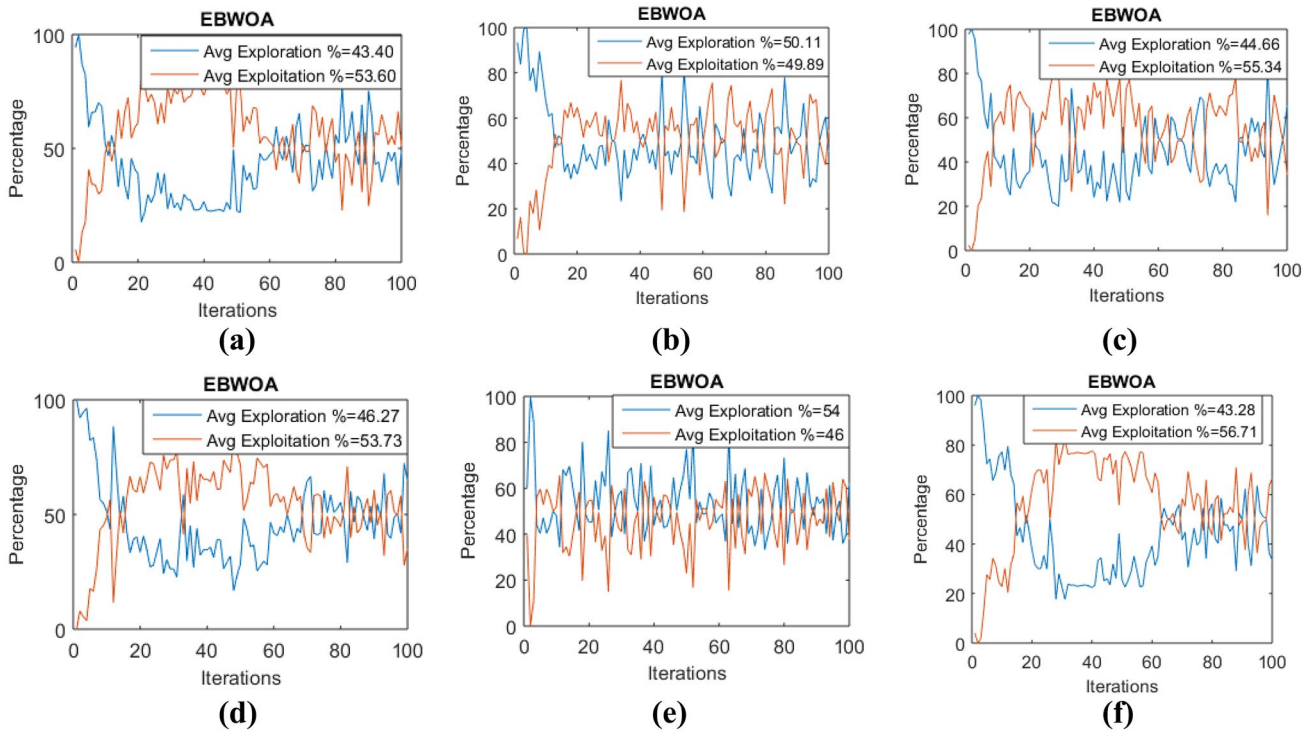

Fig. 4 Exploration vs. exploitation percentage of EBWOA

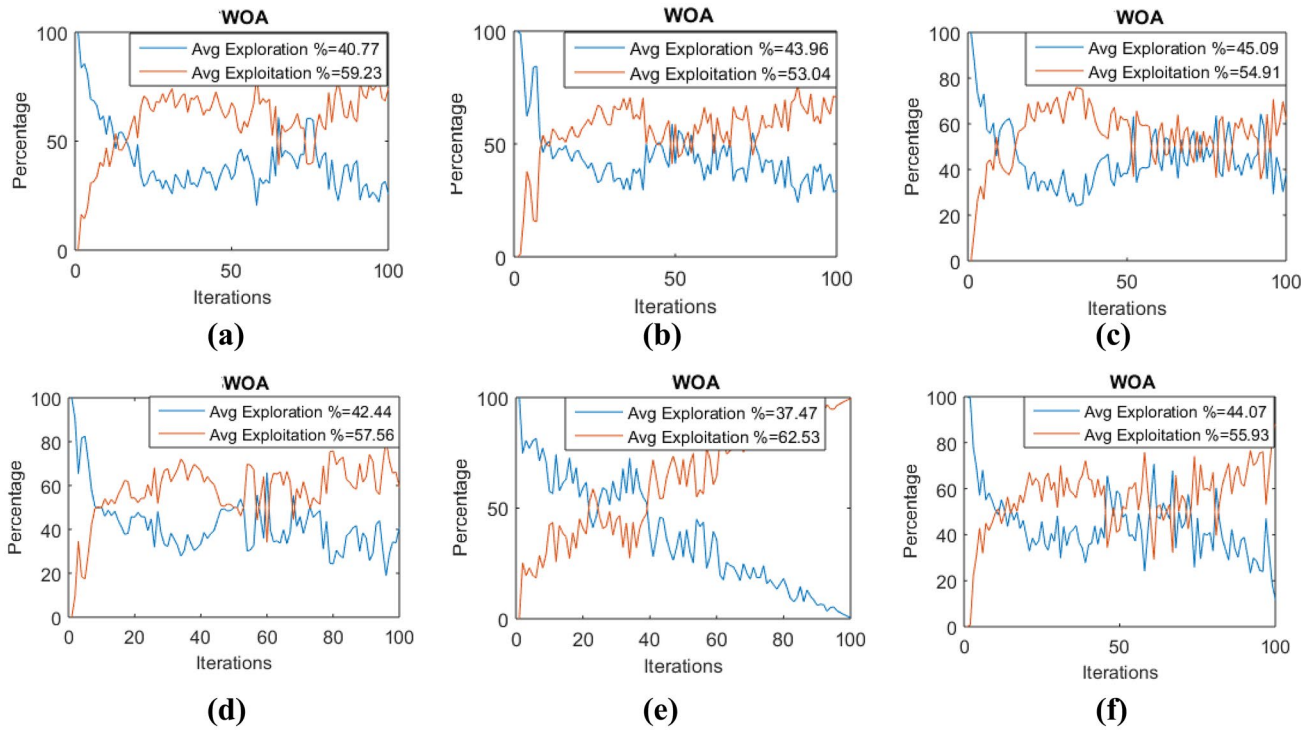

Fig. 5 Exploration vs. exploitation percentage of WOA

this paper, we aim to optimize both makespan (users' perspective) and energy consumption (service providers' perspective) metrics for the scheduling problem. The next sub-section briefly describes related work, objectives, fitness function, workloads, experimental setup, results, and analysis associated with the undertaken cloud scheduling problem.

Related Works

Using metaheuristic algorithms, a lot of researchers have tried to figure out how to solve the scheduling problem in the cloud [46–49]. This is because exhaustive solutions to the task scheduling problem are not feasible with large-scale scheduling problems [49]. The most common scheduling objectives addressed in the literature are makespan, utilization, energy efficiency, execution cost, degree of imbalance, etc. In [49], the authors proposed a fuzzy-based security-aware and energy-aware task scheduling algorithm called SAEA by introducing a parallel version of the squirrel search algorithm. The SAEA resulted in significant performance improvement over the baseline metaheuristics in terms of energy cost, makespan, degree of imbalance, and security levels. The authors in [50] introduced an improved ACO algorithm to schedule independent tasks over cloud resources to address three objectives minimizing waiting time, improving the degree of resource load balance, and reducing task completion time. On the other hand, authors in [51] presented a hybrid task scheduling algorithm by combining methods PSO and GA, which resulted in a reduction in total task completion time and improved convergence accuracy compared to the compared algorithms.

In [52], a multi-objective workflow scheduling method was presented for finding an optimal trade-off between makespan and execution cost by combining heterogeneous earliest end time (HEFT) and the ACO algorithm. Recently, a task scheduling approach called Parallel Reinforcement Learning Caledonian Crow (PRLCC) has been proposed by combining the New Caledonian crow learning algorithm (NCCLA), reinforcement learning (RL), and parallel strategy with the objectives of improving waiting time, energy consumption, security guaranty, and resource utilization [53]. The authors in [54] proposed a modified GA algorithm combined with a greedy strategy (MGGS) to optimize the task scheduling

Table 13 Performance index of EBWOA showing the percentage of increase or decrease in capacity

Algorithm	F_{26} ***PI***	F_{27} ***PI***	F_{28} ***PI***	F_{29} ***PI***	F_{30} ***PI***
ImWOA	1033499900.00[a]	58714.00[a]	3.41	−66.77	−90.01
mWOAPR	3247599900.00[a]	181282.00[a]	40.56	−52.06	−95.23
eWOA	0.00	2.60	33.89	−14.14	−22.46
ESSAWOA	0.00	0.00	123.93	45.14	71.74
LWOA	1206199900.00[a]	151830.00[a]	0.48	−35.08	−97.48
SABOA	0.00	0.00	163.46	53.55	115.47
SC_GWO	9644.30	130.98	138424.33[a]	24.34	−40.09
ISCA	0.00	0.00	63.56	7.67	39.09
Algorithm	F_{31} ***PI***	F_{32} ***PI***	F_{33} ***PI***	F_{34} ***PI***	F_{35} ***PI***
ImWOA	9.38	5.79	0.28	−33.94	0.00
mWOAPR	11.83	2.03	0.65	−35.23	0.32
eWOA	2.30	1.30	0.77	38.49	0.00
ESSAWOA	11.77	17.82	1.07	108.21	0.29
LWOA	20.92	3.15	0.17	−35.26	1.36
SABOA	16.05	46.89	7.35	167.43	0.49
SC_GWO	71.32	111.23	20.94	69.07	5.25
ISCA	17.03	19.12	5.66	99.64	0.50

[a]Indicates the percentage of rising capacity is very high due to the huge difference in the evaluated optimal value of the concerned algorithm and EBWOA

process, reduce the total completion time and average response time, and improve QoS parameters. The authors of [55] presented a multi-objective hybrid Fuzzy Hitchcock Bird-inspired approach (HBIA) with fuzzy logic and levy flight mechanism to address makespan and resource utilization goals. A recent research work [56] introduced a hybrid multi-verse optimizer with a genetic algorithm (MVO-GA) for independent scheduling tasks in a cloud environment, solving the task scheduling problem.

In another attempt, a hybrid metaheuristic solution was presented by combining WOA, Henry's gas solubility optimization (HGSO), and comprehensive opposition-based learning (COBL) for task scheduling problems to reduce makespan [57]. The authors in [58] presented an enhanced version of the MVO algorithm (EMVO) for improving makespan, throughput, and utilization. Table 14 shows the comparison of a few task-scheduling algorithms.

Problem Objectives and Fitness Function

The objectives and fitness function of the cloud scheduling problem are described as follows:

Makespan Model

The makespan objective is the latest finish time of tasks in a set of BoT applications, which is calculated as per the following:

$$\text{Makespan} = \max_{j \in PTK}\left(\text{FinishTime}_j\right) \tag{23}$$

where j is a task belonging to a distinct BoT application and PTK is the set of BoT applications. A shorter makespan is desired since it indicates faster processing [48].

Energy Model

The energy consumption (energy consumption) of an individual CPU core $\left(C_k\right)$ can be expressed as follows:

$$\text{EnergyConsumption}\left(C_k\right) = \int_0^{\text{Makespan}} \text{EnergyConsumption}_{\text{comp}}\left(C_k, t\right) + \text{EnergyConsumption}_{\text{idle}}\left(C_k, t\right) dt, \tag{24}$$

where $\text{EnergyConsumption}_{\text{comp}}$ and $\text{EnergyConsumption}_{\text{idle}}$ are the energy consumed during execution and during idle time, respectively [49].

Table 14 Comparison of task scheduling algorithms

Year	Name	Algorithm type	Performance objectives	Advantages and highlights	Limitations
2020	SAEA [49]	Improved metaheuristic	Makespan, energy consumption, degree of imbalance, and security	It presented a multi-objective approach by proposing a parallel version of the SSA technique combined with a fuzzy concept to improve solution quality and convergence rate	1. Performance of the SAEA was not tested with the real-cloud benchmarks 2. The exploration–exploitation phase trade-off still needs to be improved
2020	IACO [50]	Improved metaheuristic	Wait time, degree of resource load balance, and the cost of task completion	Traditional ACO was enhanced by adding reward and punishment coefficients and a dynamic update of the volatility coefficient to improve the convergence rate and overall solution quality	1. It did not consider the energy consumption parameter 2. It was used only for synthetic workload for benchmarking 3. Single objective
2021	PSO_PGA [51]	Hybrid metaheuristic	Makespan	Each PSO generation was divided, and particle positions were updated by the phagocytosis mechanism and operators of the GA algorithm to expand the search range of the solution space and avoid local minima	1. Single objective 2. Higher computation complexity 3. Do not consider energy consumption 4. Benchmarking with the real-workload logs was missing
2022	HEFT-ACO [52]	Standard metaheuristic	Makespan, execution cost	A multi-objective strategy was designed, combining the HEFT and ACO to obtain non-dominated Pareto fronts for finding trade-offs between conflicting objectives	1. Found inferior in comparison with the latest metaheuristics 2. Load balancing of resources was not considered 3. Inadequate balance among exploration and exploitation phases was noticed
2022	PRLCC [53]	Improved metaheuristic	Waiting time, energy consumption, security guarantee, and resource utilization	Parallel reinforcement learning was added to NCCLA to improve the quality of the solutions and avoid getting stuck in a locally optimal solution	1. Benchmarking with the real-workload logs was not available 2. Slightly higher computation complexity was evaluated
2020	MGGS [54]	Improved metaheuristic	Makespan, average response time, and degree of workload balance	MGGS tried to find an optimal solution for GA using less iteration with the help of a greedy strategy	1. Found inferior in comparison with the latest metaheuristics 2. No benchmarking with the real-workload logs 3. No discussion on the computation complexity of the proposed method
2021	MOHFHB [55]	Hybrid metaheuristic	Makespan, resource utilization, energy consumption, latency, and degree of load balance	A multi-objective approach uses HBIA, CSO operators, Sugeno-signature fuzzy, and Levy flight mechanisms to improve the exploration, trade-off between exploration and exploitation, rate of convergence, the diversity of the population, and the quality of solutions	1. No benchmarking with the real-workload logs 2. Found inferior in discussion on computation complexity of the proposed method
2022	MVO-GA [56]	Hybrid metaheuristic	Total expected execution tim	Hybrid of MVO and GA to improve its searching ability	1. Found inferior in benchmarking with the real-workload logs 2. Inadequate exploration–exploitation balance 3. Do not discuss the computational complexity of the proposed method
2021	HGSWC [57]	Hybrid metaheuristic	Makespan	WOA and COBL were added to the HGSO to improve the local search and convergence rate	1. Energy consumption parameter was not considered 2. High runtime

The overall energy usage of the cloud data center, including all CPU cores and the number of virtual machines (Nvm), can be represented as follows:

$$\text{EnergyConsumption} = \sum_{C_k}^{Nvm} \text{EnergyConsumption}(C_k) \tag{25}$$

Objective Function

The cloud scheduling problem is modeled as a bi-objective combinatorial optimization problem in this research, and the weighted-sum method is used to reduce the workload makespan and overall energy consumption of cloud computing resources. The following is the definition of the cloud scheduling problem's fitness function:

$$\text{Fitness function}, F(X) = \min(w1 \times \text{Makespan} + w2 \times \text{EnergyConsumption}) \tag{26}$$

The weights of the makespan and energy consumption objectives are $w1$ and $w2$, respectively. This paper determines optimal weight values by conducting several independent experiments with varying weights.

Experimental Setup and Workloads

This section provides the details of real benchmarking workloads, experimental configurations, experimental results, and observations. The proposed EBWOA and baseline algorithms are implemented using Java and the JMetal 5.4 metaheuristic framework (http://jmetal.github.io/jMetal/) on the CloudSim 3.0.3 simulator. Experiments are done on a computer with an Intel i7-8550U processor running at 1.80–2.0 GHz (8 cores), 16 GB of RAM, and Windows 10 installed. Each experiment is done 30 times with the same input workload and experimental settings to eliminate any bias.

The experimental workloads in this research were derived from the logs of two real-supercomputing sites, CEA-Curie and HPC2N, which can be found at http://www.cs.huji.ac.il/labs/parallel/workload. Table 15 shows that the cloud computing system has a single data center with five different VMs that are already set up [59].

Results and Analysis

The evaluated result of EBWOA on cloud scheduling problem and their comparison with other algorithms is discussed in this sub-section.

Statistical Results in Terms of Best, Average, and Worst Values

This sub-section analyzes the performance of the proposed EBWOA using a few statistical indicators against state-of-the-art baseline algorithms, viz. WOA [10], HSWOA [33], Gaussian cloud-whale optimization algorithm (GCWOAS2) [60], binary-enhanced WOA (BE-WOA) [61], multi-objective particle swarm optimization (MOPSO) [62], butterfly optimization algorithm (BOA) [63], moth flame optimization (MFO) [64], and improved WOA (IWOA) [22] algorithms. Three measures, e.g., best, average, and worst values of obtained results, have been considered for the analysis. The best, average, and worst values are the minimum, average, and maximum values, respectively, among 30 repeated executions of each tested algorithm's independent experiment for makespan and energy consumption metrics. Before conducting final experiments, a convergence analysis is conducted to determine the optimal values of parameters of all metaheuristics involved in the cloud task scheduling problem (note: convergence study details can be obtained from the corresponding author at any time).

Tables 16 and 17 present the statistical findings of makespan and energy consumption for all scheduling algorithms for CEA-Curie workloads, whereas Tables 18 and 19 show the statistical results for HPC2N workloads. The least values are shown in **bold**. It is clear from Tables 16, 17, 18, and 19 that the proposed EBWOA method yielded significantly better minimum, average, and maximum values of makespan and energy consumption metrics than baseline algorithms.

Table 15 Description of the VM configuration adopted for experimentation

VM instance type	VM instance ID	# VMs	# CPU cores per VM	MIPS per core	CPU model	EC_{idle}(W/h)	EC_{comp}(W/h)
Type 1	T2.nano	20	1	3400	Xeon E5-2637 V4	23.625	33.75
Type 2	T2.xlarge	10	4	2600	Xeon E5-2623 V4	59.5	85
Type 3	T2.2xlarge	8	8	2100	Xeon E5-2620 V4	59.5	85
Type 4	M5.4xlarge	6	16	2500	Xeon Plat. 8180 M	82	117.14
Type 5	M4.10xlarge	4	40	2400	Xeon E5-2686 V4	225.55	322.22

Table 16 Statistical results of the makespan metric for CW0-CW9 workloads

Workload	Best/Avg./Worst	WOA	IWOA	HSWOA	GCWOAS2	BE-WOA	PSO	BOA	MFO	EBWOA	Rank (EBWOA)
CW0	Best	11363.45	11235.10	10547.31	10724.31	10587.44	11960.26	11545.10	10651.75	**10389.66**	1
	Avg.	13941.60	13181.24	12340.4	12257.33	12113.47	14454.11	13491.24	12449.01	**11954.27**	1
	Worst	19272.46	16635.47	15670.49	15223.02	15072.81	21599.99	16945.47	15779.38	**14894.87**	1
CW1	Best	14273.73	14270.85	14030.82	13121.40	12985.04	14443.43	14580.85	14130.81	**12860.73**	1
	Avg.	16724.08	16353.68	15617.21	15747.13	15603.72	17483.91	16663.68	15725.85	**15447.62**	1
	Worst	21726.84	20160.21	19008.62	19072.03	18951.10	25807.67	20470.21	19111.50	**18699.4**	1
CW2	Best	7323.90	7937.81	7619.24	7585.89	7485.58	8811.59	8247.81	7722.08	**7195.93**	1
	Avg.	10791.37	10621.14	10357.46	10177.28	10030.72	12080.79	10931.14	10466.98	**9859.77**	1
	Worst	14752.78	14153.57	13346.98	12366.49	12245.15	16524.91	14463.57	13453.06	**11966.24**	1
CW3	Best	18629.59	18619.71	18357.62	18576.42	18419.61	19848.23	18929.71	18457.46	**18222.75**	1
	Avg.	22521.89	22505.07	21849.06	21503.97	21356.55	24679.88	22815.07	21956.68	**21205.72**	1
	Worst	27867.30	27013.36	25790.83	24780.24	24664.57	29426.71	27323.36	25889.87	**24437.52**	1
CW4	Best	15092.87	14880.50	14931.41	13808.59	13696.47	15174.35	15190.50	15038.71	**13521.7**	1
	Avg.	17631.72	17511.62	17278.79	17112.11	16961.93	18752.94	17821.62	17388.33	**16812.44**	1
	Worst	23612.27	21530.89	**20120.93**	21515.73	21362.65	25005.48	21840.89	20239.83	**21307.37**	2
CW5	Best	4122.69	4195.65	4368.58	4077.45	3917.96	4517.03	4505.65	4483.09	**3805.21**	1
	Avg.	5519.68	5394.51	5279.36	5276.21	5121.63	6274.81	5704.51	5389.55	**4973.92**	1
	Worst	7930.09	7006.66	6901.61	6968.55	6769.61	9284.97	7316.66	7013.47	**6574.39**	1
CW6	Best	9183.54	9250.03	9014.72	8740.02	8612.88	9058.90	9560.03	9123.01	**8414.02**	1
	Avg.	12420.86	11866.79	11325.70	11003.33	10858.90	12689.67	12176.79	11433.84	**10687.41**	1
	Worst	16761.14	14863.11	13563.41	13152.68	12964.96	19732.64	15173.11	13670.13	**12861.1**	1
CW7	Best	3600.28	**3605.61**	3832.20	3971.24	3782.59	3746.42	3915.61	3933.73	3694.79	2
	Avg.	4857.15	4708.27	4622.19	4587.59	4435.12	5614.67	5018.27	4731.31	**4285.79**	1
	Worst	8658.96	6686.05	**6043.23**	6513.01	6368.77	8717.20	6996.05	6156.96	6263.07	4
CW8	Best	5265.80	5505.43	5090.78	4812.19	4634.92	5975.56	5815.43	5209.97	**4480.74**	1
	Avg.	7182.61	7125.08	6261.64	6214.90	6068.30	8785.39	7435.08	6373.44	**5920.05**	1
	Worst	9935.95	9983.07	8685.42	7579.10	7453.74	12786.17	10293.07	8789.48	**7360.92**	1
CW9	Best	5581.55	5523.68	4540.57	4704.54	4517.78	5883.46	5833.68	4648.56	**4390.24**	1
	Avg.	7318.54	7128.89	7048.32	7074.55	6927.60	7564.67	7438.89	7156.61	**6775.17**	1
	Worst	9576.50	8504.70	8091.04	8409.53	8293.92	10204.26	8814.70	8204.87	**8012.30**	1

Bold data indicates the minimum values

Table 17 Statistical results of the energy consumption metric for CW0-CW9 workloads

Workload	Best/Avg./Worst	WOA	IWOA	HSWOA	GCWOAS2	BE-WOA	PSO	BOA	MFO	EBWOA	Rank (EBWOA)
CW0	Best	7374.80	6769.26	6757.75	6638.74	6461.21	7206.21	6945.26	6880.32	**6419.89**	1
	Avg.	8325.78	7975.56	7988.37	8155.80	7989.02	8721.13	8340.70	8198.12	**7907.02**	1
	Worst	11093.31	9499.02	**9062.11**	9820.83	9703.80	11899.30	9674.02	9180.12	9587.39	5
CW1	Best	7519.42	**8431.48**	9126.01	8721.39	8565.48	8633.28	8739.48	9229.56	8445.21	2
	Avg.	10406.16	10121.09	10062.02	10019.32	9833.48	11053.11	10456.31	10238.58	**9767.75**	2
	Worst	14904.43	11855.60	**11759.39**	12115.87	11951.06	17715.73	12087.60	11859.77	11884.81	3
CW2	Best	5369.94	5448.73	5317.59	5342.22	5141.59	5582.60	5737.73	5393.17	**5089.72**	1
	Avg.	6703.09	6391.15	6458.04	6586.59	6408.36	7165.61	6608.40	6525.21	**6345.78**	1
	Worst	8810.50	**7794.96**	8017.71	8758.92	8657.83	8970.75	7997.96	8102.24	8487.74	6
CW3	Best	12257.39	12660.75	12941.02	12955.46	12796.99	13506.06	12874.75	13026.76	**12715.30**	1
	Avg.	14613.78	14282.05	14300.94	14438.32	14265.00	15468.06	14594.40	14481.83	**14192.18**	1
	Worst	17154.44	16650.80	16363.84	16594.57	16433.90	17426.96	16758.80	16491.68	**16323.19**	1
CW4	Best	9445.15	9282.88	9169.04	8833.90	8655.26	9899.06	9601.81	9286.66	**8572.04**	1
	Avg.	10912.80	10674.53	10484.79	10410.50	10239.39	12001.09	10935.66	10611.28	**10158.62**	1
	Worst	12427.49	**12260.75**	12338.15	12733.03	12567.17	14625.70	12509.75	12427.57	12522.47	5
CW5	Best	2800.97	2581.03	2695.53	2808.14	2639.28	3007.40	2824.03	2772.98	**2519.35**	1
	Avg.	3501.10	3215.76	3255.65	3429.69	3266.25	3809.02	3511.13	3355.32	**3183.87**	1
	Worst	5043.90	4044.96	4009.42	4176.86	4016.95	4468.13	4333.96	4111.37	**3900.03**	1
CW6	Best	6720.48	**6516.86**	6630.69	6786.18	6606.82	6815.78	6643.86	6696.47	6521.18	2
	Avg.	7877.79	7538.10	7560.22	7691.67	7504.10	7954.14	7774.60	7627.41	**7437.44**	1
	Worst	10097.10	8967.87	8236.05	8407.44	8196.56	9842.82	9248.67	8327.51	**8158.81**	1
CW7	Best	2592.72	**2206.52**	2331.39	2520.34	2377.52	2561.79	2420.52	2422.04	**2319.82**	2
	Avg.	3304.50	2935.50	2934.57	3088.19	2910.65	3797.62	3193.88	3048.35	**2838.28**	1
	Worst	4366.37	4027.40	3951.92	3862.34	3680.03	5173.74	4157.40	4040.33	**3587.49**	1
CW8	Best	3895.17	3713.60	3588.45	3624.79	3415.77	3970.40	3919.93	3698.84	**3378.58**	1
	Avg.	4666.43	4320.73	4356.97	4499.18	4331.11	4905.71	4624.10	4487.93	**4245.63**	1
	Worst	6560.29	5607.50	**5013.65**	5538.14	5385.59	6194.94	5779.50	5084.30	5311.45	3
CW9	Best	3416.60	3096.43	3055.97	3064.98	2837.60	2868.46	3458.43	3106.15	**2793.19**	1
	Avg.	4713.41	**4281.87**	4353.87	4558.32	4396.88	4849.14	4651.67	4573.47	4305.86	2
	Worst	6633.52	**5258.32**	5264.39	6057.81	5900.03	7715.02	5642.32	5375.55	5793.80	5

Bold data indicates the minimum values

Table 18 Statistical results of the makespan metric for HW0-HW9 workloads

Workload	Best/Avg./Worst	WOA	IWOA	HSWOA	GCWOAS2	BE-WOA	PSO	BOA	MFO	EBWOA	Rank (EBWOA)
HW0	Best	22180.61	21766.17	20158.09	19430.61	19254.74	22935.48	22066.17	20136.63	**19157.34**	1
	Avg.	26960.33	26659.94	22884.88	22165.40	21958.09	31659.31	26959.94	22358.71	**21884.45**	1
	Worst	36712.86	36809.26	34961.67	34264.85	34055.81	47578.36	37109.26	35391.42	**33961.02**	1
HW1	Best	16787.62	15619.99	16223.03	15439.17	15157.88	18980.14	15919.99	16622.34	**15222.86**	1
	Avg.	19853.62	19648.61	18859.23	18122.71	17914.15	22903.96	19948.61	19041.76	**17858.70**	1
	Worst	24409.71	26874.21	23624.50	22849.57	22599.96	28885.86	27174.21	22796.08	**22624.46**	1
HW2	Best	19095.65	18864.36	18007.67	17301.90	17087.95	20212.26	19164.36	17910.82	**17006.81**	1
	Avg.	22505.35	22272.30	20644.24	19913.73	19737.29	25260.03	22572.30	20408.97	**19643.65**	1
	Worst	30292.31	30001.89	27943.20	27186.83	27075.30	32357.04	30301.89	29412.09	**26942.37**	1
HW3	Best	14836.84	**13419.31**	14657.62	13898.78	13638.05	16746.90	13719.31	14615.07	13657.52	2
	Avg.	17599.70	17348.52	17529.56	16810.82	16619.99	19585.00	17648.52	17288.48	**16529.08**	1
	Worst	20977.23	21213.13	21571.95	20886.08	20631.90	22251.07	21513.13	**20432.90**	20571.75	2
HW4	Best	11420.90	**10176.61**	11179.64	10521.11	10366.73	11622.14	10476.61	11716.94	10179.31	2
	Avg.	13369.59	13354.34	13578.59	12848.86	12653.53	13851.12	13654.34	13167.45	**12578.13**	1
	Worst	14897.75	16100.40	15775.65	14982.55	14853.66	18141.63	16400.40	15015.63	**14775.54**	1
HW5	Best	4462.61	**3622.51**	4927.27	4140.67	4014.98	4524.74	3922.51	4117.70	3927.13	3
	Avg.	5880.41	5657.67	6598.49	5858.32	5663.52	7052.93	5957.67	5676.38	**5598.00**	1
	Worst	8596.71	8705.34	9275.37	8527.69	8403.77	12901.66	9005.34	**7568.60**	8275.08	2
HW6	Best	8262.12	**7918.58**	8928.68	8197.13	7898.03	9200.91	8218.58	7928.44	7928.10	2
	Avg.	10391.64	10326.45	11136.92	10398.97	10188.95	11268.08	10626.45	10210.14	**10136.35**	1
	Worst	12549.33	14276.44	13482.20	12732.02	12463.85	15773.25	14576.44	**12091.02**	12481.25	2
HW7	Best	4106.28	3787.09	4579.94	3912.25	3737.47	4738.35	4087.09	3878.44	**3579.58**	1
	Avg.	5671.69	5447.06	6441.52	5711.94	5513.12	6559.59	5747.06	5480.60	**5440.96**	1
	Worst	7792.12	8781.96	9189.30	8493.74	8330.63	9001.94	9081.96	**8080.22**	8188.71	2
HW8	Best	3388.89	**2460.23**	4189.28	3486.39	3273.30	3047.82	2760.23	3206.86	3188.61	3
	Avg.	4148.83	**3984.68**	5140.47	4417.80	4213.91	4386.93	4284.68	4064.54	4139.95	3
	Worst	10071.87	11894.68	7078.77	6365.87	6179.96	6202.69	12194.68	7978.97	**6078.36**	1
HW9	Best	2370.83	**1875.34**	3189.23	2458.78	2165.94	2804.61	2175.34	2230.94	2188.88	3
	Avg.	3928.33	3761.85	4737.53	4016.27	3825.04	4179.46	4061.85	3797.48	**3737.02**	1
	Worst	9687.98	11909.61	6816.40	6045.10	5772.91	7970.18	12209.61	7332.97	**5816.17**	1

Bold data indicates the minimum values

Table 19 Statistical results of the energy consumption objective for HW0-HW9 workloads

Workload	Best/Avg./Worst	WOA	IWOA	HSWOA	GCWOAS2	BE-WOA	PSO	BOA	MFO	EBWOA	Rank (EBWOA)
HW0	Best	15457.21	15092.49	15346.65	14757.01	14605.88	17723.35	15342.49	14711.92	**14191.74**	1
	Avg.	20314.62	18790.17	17892.51	16744.26	16491.87	23907.99	19040.17	17170.77	**16264.16**	1
	Worst	26287.40	22668.38	21330.20	20125.86	19932.21	31626.77	22918.38	20451.14	**19818.35**	1
HW1	Best	10707.41	10481.61	10676.07	9994.28	9662.62	11258.19	10731.61	10055.60	**9650.903**	1
	Avg.	12473.69	12152.66	12512.52	11786.58	11544.86	13991.44	12402.66	11750.63	**11347.36**	1
	Worst	15469.51	14848.86	15967.21	15136.18	14800.41	18560.88	15098.86	15055.03	**14773.28**	1
HW2	Best	**10298.37**	10984.09	12252.77	11207.83	10985.65	11558.36	11234.09	11286.86	10798.42	2
	Avg.	13442.37	13114.66	13660.30	12927.07	12657.75	14067.10	13364.66	12898.74	**12466.51**	1
	Worst	16522.92	**15321.12**	16482.10	16211.43	15874.38	17854.48	15571.12	15658.78	15696.14	3
HW3	Best	8941.88	9194.43	10078.66	9353.38	9054.24	10186.48	9444.43	9224.07	**8961.955**	1
	Avg.	10811.72	10729.42	11257.47	10469.55	10223.34	12652.43	10979.42	10456.97	**10009.86**	1
	Worst	13166.02	12166.42	12601.58	11812.70	11712.67	15289.81	12416.42	11679.04	**11290.35**	1
HW4	Best	6661.51	7305.62	7795.80	7122.98	6959.74	7506.78	7555.62	6974.34	**6806.813**	1
	Avg.	8650.12	8327.42	9025.17	8418.04	8172.58	9044.92	8577.42	8285.65	**7973.815**	1
	Worst	10888.54	9993.36	10403.79	9975.55	9647.39	11732.53	10243.36	9565.08	**9428.835**	1
HW5	Best	**2332.03**	2735.37	3545.35	2956.54	2570.87	3340.41	2985.37	2845.06	2641.271	2
	Avg.	3741.86	3429.90	4242.05	3814.88	3539.75	4210.07	3679.90	3540.14	**3355.756**	**1**
	Worst	5524.02	**4734.47**	6254.58	6265.61	5871.71	5197.82	4984.47	5405.52	5795.239	6
HW6	Best	**4949.54**	5406.58	6141.63	5679.24	5279.33	5576.80	5656.58	5582.38	5213.25	2
	Avg.	6275.30	6253.26	7065.24	6535.61	6283.70	7145.42	6503.26	6328.16	**6084.29**	1
	Worst	8747.99	**7340.18**	8602.47	8390.41	8167.03	10248.08	7590.18	7911.92	7916.16	4
HW7	Best	2119.43	2564.00	3029.76	2638.06	2298.27	2984.80	2814.00	2427.06	**2263.94**	1
	Avg.	3308.29	3149.06	3916.14	3442.05	3207.41	3888.74	3399.06	3190.15	**2986.52**	1
	Worst	4142.45	**3856.29**	4857.46	4392.87	4233.57	5374.67	4106.29	4010.12	4046.17	3
HW8	Best	1669.36	**1260.62**	2008.54	1856.09	1574.30	1767.25	1510.62	1422.13	1391.22	2
	Avg.	2158.86	**1820.95**	2638.70	2335.08	2087.64	2517.81	2070.95	1914.90	1889.48	2
	Worst	4953.49	**4340.59**	5559.44	5296.52	5087.47	6170.04	4590.59	4765.55	4867.63	4
HW9	Best	1258.37	**905.77**	1860.97	1791.93	1431.71	1518.28	1155.77	1138.99	1325.71	5
	Avg.	2113.03	1764.08	2464.74	2194.35	1908.65	2644.26	2014.08	**1701.94**	1756.20	2
	Worst	4636.22	3602.78	3697.88	3768.24	3415.42	5604.93	3852.78	**2768.03**	3388.88	2

Bold data indicates the minimum values

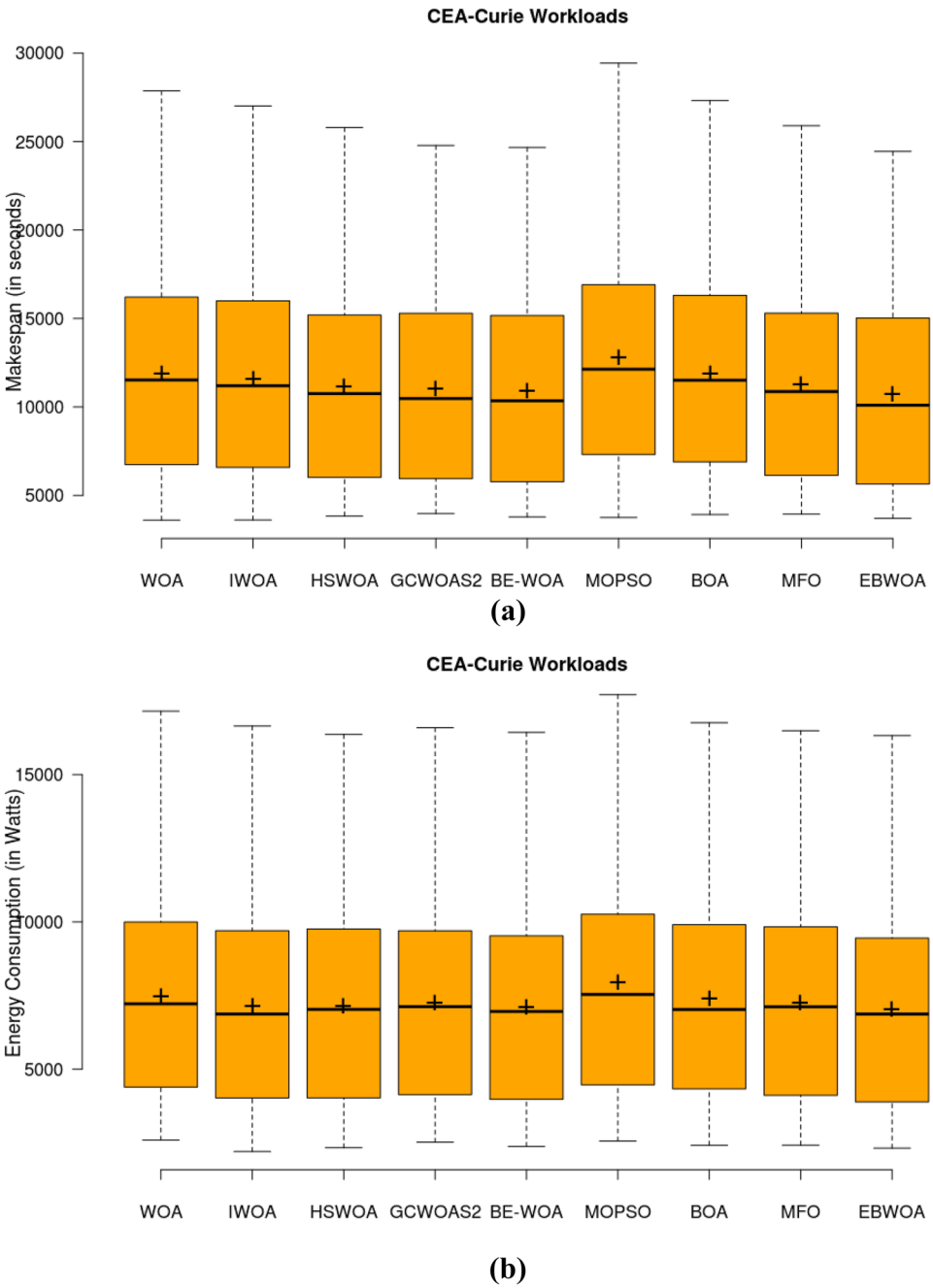

Fig. 6 Box plots for CEA-Curie workloads. **a** Makespan. **b** Energy consumption

Overall Makespan and Energy Consumption Results

The makespan and energy usage box plots of all scheduling experiments conducted on CEA-Curie and HPC2N workloads are shown in Figs. 6 and 7. The proposed EBWOA algorithm has produced a significantly better makespan and energy consumption than each tested baseline algorithm. The BE-WOA, GCWOAS2, and HSWOA algorithms alternatively ended up in first, second, and third runner-up positions compared to the EBWOA approach. These findings show that the EBWOA algorithm outperforms the baseline algorithms in terms of performance, robustness, and stability.

Summarizing the Overall Cloud Scheduling Results

Finally, the overall experimental outcomes of the suggested EBWOA strategy over baseline approaches are reported using the overall mean, median, and percentage of performance improvement rate (PIR%). PIR% helps in calculating the percentage reduction in makespan and energy

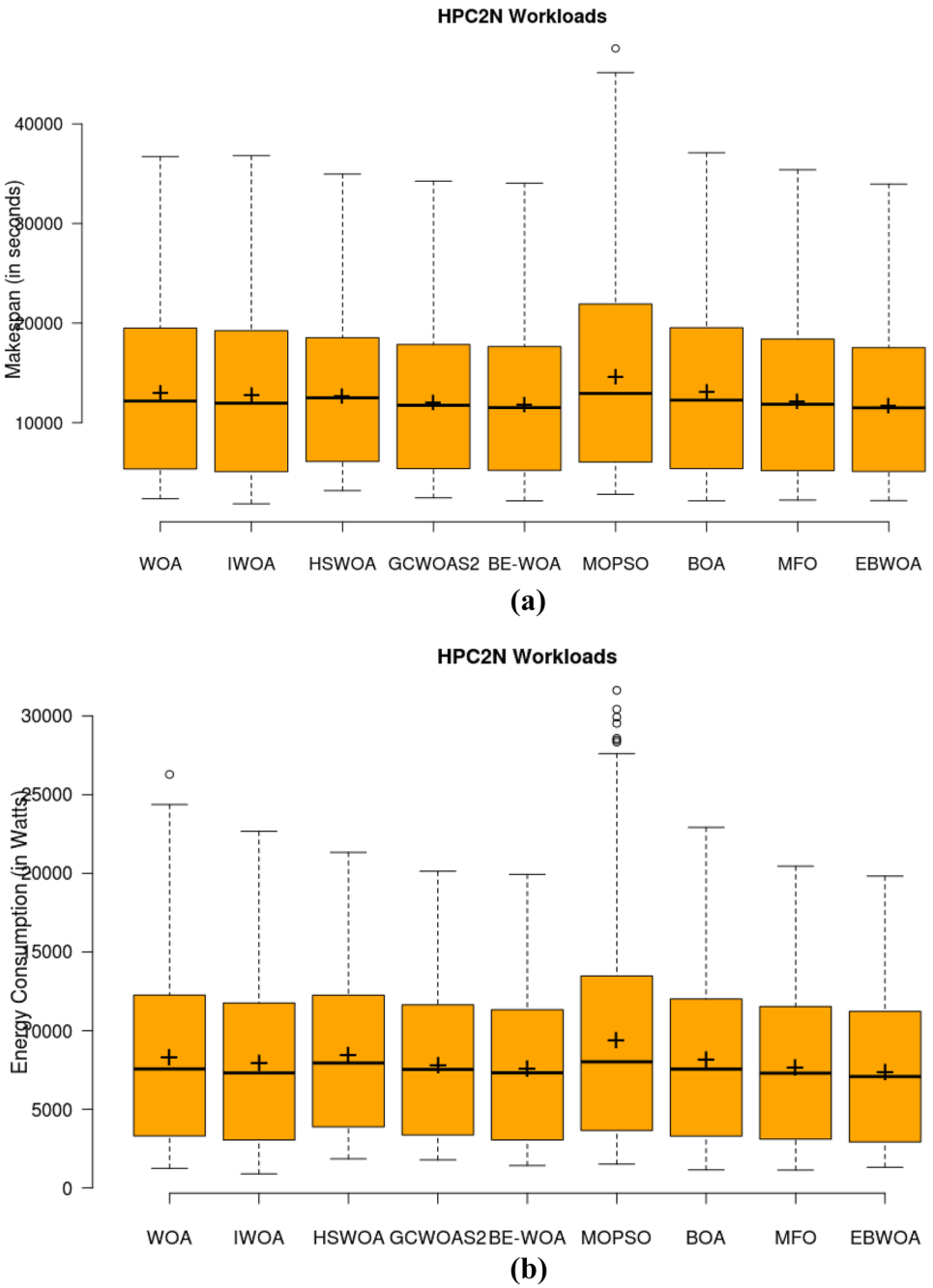

Fig. 7 Box plots for HPC2N workloads. **a** Makespan. **b** Energy consumption

consumption achieved by the EBWOA over baseline techniques, and it is calculated as follows:

approaches for both CEA-Curie and HPC2N workloads. In the case of CEA-Curie workloads, EBWOA resulted in

$$\text{PIR}(\%) = \frac{\text{Performance (other Algorithm)} - \text{Performance (EBWOA)}}{\text{Performance (EBWOA)}} \times 100\% \tag{27}$$

Table 20 shows that the EBWOA approach significantly reduces makespan and energy consumption, as indicated by outstanding PIR% results over baseline scheduling makespan and energy consumption reductions in the range of 1.44–18.96% and 1.08–13.27%, respectively, over the baseline metaheuristics. On the other hand, for HPC2N

Table 20 Overall mean, median, and PIR% results

Policy	WOA	IWOA	HSWOA	GCWOAS2	BE-WOA	MOPSO	BOA	MFO	EBWOA
CEA-Curie									
Makespan (s)									
Mean	11,890.95	11,639.63	11,198.02	11,095.44	10,947.80	12,838.08	11,949.63	11,307.16	10,792.22
Median	11,522.28	11,199.14	10,751.37	10,475.4	10,344.18	12,130.13	11,509.14	10,866.13	10,097.09
PIR% of EBWOA over	10.18%	7.85%	3.76%	2.81%	1.44%	18.96%	10.72%	4.77%	
Energy consumption (W)									
Mean	7502.48	7173.63	7175.54	7287.76	7114.42	7972.46	7418.61	7265.13	7038.24
Median	7218.97	6872.21	7032.31	7122.41	6961.81	7539.61	7028.03	7117.54	6871.73
PIR% of EBWOA over	6.60%	1.92%	1.95%	3.55%	1.08%	13.27%	5.40%	3.22%	
HPC2N									
Makespan (s)									
Mean	13,030.95	12,846.14	12,755.14	12,026.48	11,828.76	14,670.64	13,146.14	12,149.45	11,754.63
Median	12,165.13	11,958.84	12,494.82	11,748.11	11,512.05	12,945.09	12,258.84	11,851.78	11,494.37
PIR% of EBWOA over	10.86%	9.29%	8.51%	2.31%	0.63%	24.81%	11.84%	3.36%	
Energy consumption (W)									
Mean	8328.99	7953.16	8467.48	7866.75	7611.75	9407.02	8203.16	7723.8	7413.4
Median	7568.49	7311.03	7946.03	7533.84	7320.68	8019.48	7561.03	7297.58	7086.23
PIR% of EBWOA over	12.35%	7.28%	14.22%	6.12%	2.68%	26.89%	10.65%	4.19%	

workloads, EBWOA's performance improvement in the range of 0.63–24.81% (for makespan) and 2.68–26.89% (for energy consumption) has been observed over the baseline metaheuristics.

Conclusion

WOA has several advantages, such as a simple structure, fewer parameters, and simplified implementation. Besides the advantages, WOA also has disadvantages, such as low exploratory ability, early convergence, and low solution accuracy. This research features a new WOA (EBWOA) variant with improved exploration capabilities and a balance between exploration and exploitation. The method updates solutions in the population using the local or global elite solution. The distinctiveness of the newly developed method is that the exploration is carried out using the encircling prey phase, which is used in the basic WOA for exploitation. Unlike simple WOA, it uses only two steps to circle the prey and bubble-net methods to update solutions. In the encircling prey phase, the locally best solution to update other solutions promoted exploitation while exploring the quest region. The addition of inertia weight enabled the method to conduct an exhaustive search for the best local and global solutions. The effectiveness of the proposed methods is evaluated using twenty-five classic benchmark functions, IEEE CEC 2019 functions, two design problems, and a cloud task scheduling problem. Comparisons of numerical results using various basic and modified algorithms, statistical analysis, convergence analysis, runtime analysis, exploration vs. exploitation capability, and performance index verification all show that the changes proposed in this document make WOA better at finding solutions.

Appendix

Table 21 Variable and fixed dimension unimodal and multimodal functions

Function	Equation	Search region	*D*	Optimal result
Sphere	$F_1(x) = \sum_{k=1}^{D} x_k^2$	[-100,100]	30	0
Schwefel 2.22	$F_2(x) = \sum_{k=1}^{D} \lvert x_k \rvert + \prod_{k=1}^{D} \lvert x_k \rvert$	[-10,10]	30	0
Schwefel 1.12	$F_3(x) = \sum_{k=1}^{D} \left(\sum_{l=1}^{k} x_l \right)^2$	[-100,100]	30	0
Schwefel 2.21	$F_4(x) = \max_k \left[\lvert x_k \rvert, 1 \le k \le D \right]$	[-100,100]	30	0
Step	$F_5(x) = \sum_{k=1}^{D} \left(\lvert x_k + 0.5 \rvert \right)^2$	[-100,100]	30	0
Quartic	$F_6(x) = \sum_{k=1}^{D} x_k^4 + random(0,1)$	[-1.28,1.28]	30	0
Zakharov	$F_7(x) = \sum_{k=1}^{D} x_k^2 + \left(\sum_{k=1}^{D} 0.5kx_k \right)^2 + \left(\sum_{k=1}^{D} 0.5kx_k \right)^4$	[-5,10]	30	0
Cigar	$F_8(x) = x_1^2 + 10^6 \sum_{k-2}^{D} x_k^6$	[-100,100]	30	0
Powell	$F_9(x) = \sum_{k=1}^{D/4} \left[\left(x_{4k-3} + 10x_{4k-2} \right)^2 + 5\left(x_{4k-1} + x_{4k} \right)^2 + \left(x_{4k-2} + 2x_{4k-1} \right)^4 + 10\left(x_{4k-3} + 10x_{4k} \right)^4 \right]$	[-4, 5]	30	0
Tablet	$F_{10}(x) = 10^6 x_1^2 + \sum_{k-2}^{D} x_k^6$	[-1,1]	30	0
Elliptic	$F_{11}(x) = \sum_{k=2}^{D} \left(10^6 \right)^{(k-1)(D-1)} \cdot x_k^2$	[-100,100]	30	0
Brown	$F_{12}(x) = \sum_{k=1}^{n-1} \left(x_k^2 \right)^{\left(x_{k+1}^2 + 1 \right)} + \left(x_{k+1}^2 \right)^{\left(x_k^2 + 1 \right)}$	[-1, 4]	30	0
Matyas	$F_{13}(x) = 0.26\left(\left(x_1^2 + x_2^2 \right) \right) - 0.48x_1x_2$	[-10,10]	2	0
Rastrigin	$F_{14}(x) = \sum_{k=1}^{D} \left[x_k^2 - 10\cos(2\Pi x) + 10 \right]$	[-5.12,5.12]	30	0
Ackley	$F_{15}(x) = -20\exp(-0.2 \sqrt{\frac{1}{D} \sum_{k=1}^{D} x_k^2} - exp\left(\frac{1}{D} \sum_{k=1}^{D} \cos 2\Pi x_k \right)) + 20 + e$	[-32,32]	30	0
Griewank	$F_{16}(x) = \frac{1}{4000} \sum_{K=1}^{D} x_k^2 - \prod_{k=1}^{D} \cos(\frac{x_k}{\sqrt{k}}) + 1$	[-600,600]	30	0
Alpine	$F_{17} = \sum_{k=1}^{D} \left\lvert x_k \sin\left(x_k \right) + 0.1x_k \right\rvert$	[-10, 10]	30	0
Csendes	$F_{18}(x) = \sum_{k=1}^{D} x_k^6 (2 + sin\frac{1}{x_k})$	[-1,1]	30	0
Inverted Cosine Mixture	$F_{19}(x) = 0.1D - (0.1 \sum_{k=1}^{D} \cos\left(5\Pi x_k \right) - \sum_{k=1}^{D} x_k^2)$	[-1,1]	30	0
Salomon	$F_{20}(x) = 1 - \cos\left(2\Pi \sqrt{\sum_{k=1}^{D} x_k^2} \right) + 0.1\sqrt{\sum_{k=1}^{D} x_k^2}$	[-100,100]	30	0
Kawalik	$F_{21}(x) = \sum_{k=1}^{11} \left[a_k - \frac{x_1 \left(b_k^2 + b_k x_2 \right)}{b_k^2 + b_k x_3 + x_4} \right]^2$	[-5,5]	4	0.0003
Shekel1	$F_{22}(x) = -\sum_{k=1}^{5} \left[\left(x - a_k \right) \left(x - a_k \right)^T + c_k \right]^{-1}$	[0,10]	04	-10.1532
Shekel2	$F_{23}(x) = -\sum_{k=1}^{7} \left[\left(x - a_k \right) \left(x - a_k \right)^T + c_k \right]^{-1}$	[0,10]	04	-10.4028
Shekel3	$F_{24}(x) = -\sum_{k=1}^{10} \left[\left(x - a_k \right) \left(x - a_k \right)^T + c_k \right]^{-1}$	[0,10]	04	-10.5363
Bohachevsky1	$F_{25}(x) = x_1^2 + 2x_2^2 - 0.3\cos\left(3\Pi x_1 \right) - 0.4\cos\left(4\Pi x_2 \right) + 0.7$	[-100,100]	02	0

Data Availability All data generated or analyzed during this study are included in the article.

Declarations

Ethical Approval This article does not contain any studies with human participants or animals performed by any of the authors.

Conflict of Interest The authors declare no competing interests.

References

1. Chakraborty S, Saha AK, Sharma S, Chakraborty R, Debnath S. A hybrid whale optimization algorithm for global optimization. J Ambient Intell Human Comput. 2021;1–37.
2. Angeline PJ. Genetic programming: on the programming of computers by means of natural selection. Biosystems. 1994;33(1):69–73. https://doi.org/10.1016/0303-2647(94)90062-0.
3. Storn R, Price K. Differential evolution – a simple and Efficient heuristic for global optimization over continuous spaces.

J Global Optim. 1997;11(4):341–59. https://doi.org/10.1023/a:1008202821328.
4. Tian X, Yang HD, Deng FQ. A novel artificial immune network algorithm. IEEE 2006 international conference on machine learning and cybernetics; 2006. p. 2159–65.
5. Anandita S, Rosmansyah Y, Dabarsyah B, Choi JU. Implementation of dendritic cell algorithm as an anomaly detection method for port scanning attack. International Conference on Information Technology Systems and Innovation (ICITSI); 2015. https://doi.org/10.1109/icitsi.2015.7437688.
6. Faramarzi A, Heidarinejad M, Stephens B, Mirjalili S. Equilibrium optimizer: a novel optimization algorithm. Knowl Based Syst. 2019;191:105190. https://doi.org/10.1016/j.knosys.2019.105190.
7. Hashim FA, Houssein EH, Mabrouk MS, Al-Atabany W, Mirjalili S. Henry gas solubility optimization: a novel physics-based algorithm. Futur Gener Comput Syst. 2019;101:646–67. https://doi.org/10.1016/j.future.2019.07.015.
8. Cheng MY, Prayogo D. Symbiotic organisms search: a new metaheuristic optimization algorithm. Comput Struct. 2014;139:98–112. https://doi.org/10.1016/j.compstruc.2014.03.007.
9. Mirjalili S, Lewis A. The whale optimization algorithm. Adv Eng Softw. 2016;95:51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008.
10. Rao RV, Savsani VJ, Vakharia DP. Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems. Comput Aided Des. 2011;43(3):303–15. https://doi.org/10.1016/j.cad.2010.12.015.
11. Abdollahzadeh B, Soleimanian Gharehchopogh F, Mirjalili S. Artificial gorilla troops optimizer: a new nature-inspired metaheuristic algorithm for global optimization problems. Int J Intell Syst. 2021;36(10):5887–958.
12. Azizi M. Atomic orbital search: a novel metaheuristic algorithm. Appl Math Model. 2021;93:657–83.
13. Hashim FA, Houssein EH, Hussain K, Mabrouk MS, Al-Atabany W. Honey Badger Algorithm: new metaheuristic algorithm for solving optimization problems. Math Comput Simul. 2022;192:84–110.
14. Mohammad Hasani Zade B, Mansouri N. PPO: a new nature-inspired metaheuristic algorithm based on predation for optimization. Soft Comput. 2022;26(3):1331–402.
15. Nama S, Saha AK, Ghosh S. Improved backtracking search algorithm for pseudo-dynamic active earth pressure on retaining wall supporting c-Φ backfill. Appl Soft Comput. 2017;52:885–97. https://doi.org/10.1016/j.asoc.2016.09.037.
16. Wolpert DH, Macready WG. No free lunch theorems for optimization. IEEE Trans Evol Comput. 1997;1(1):67–82. https://doi.org/10.1109/4235.585893.
17. Kaur G, Arora S. Chaotic whale optimization algorithm. J Comput Des Eng. 2018;5(3):275–84. https://doi.org/10.1016/j.jcde.2017.12.006.
18. Sun Y, Wang X, Chen Y, Liu Z. A modified whale optimization algorithm for large-scale global optimization problems. Expert Syst Appl. 2018;114:563–77. https://doi.org/10.1016/j.eswa.2018.08.027.
19. Chen H, Xu Y, Wang M, Zhao X. A balanced whale optimization algorithm for constrained engineering design problems. Appl Math Model. 2019;71:45–59.
20. Laskar NM, Guha K, Chatterjee I, et al. HWPSO: A new hybrid whale-particle swarm optimization algorithm and its application in electronic design optimization problems. Appl Intell. 2019;49:265–91. https://doi.org/10.1007/s10489-018-1247-6.
21. Mostafa Bozorgi S, Yazdani S. IWOA: An improved whale optimization algorithm for optimization problems. J Comput Des Eng. 2019;6(3):243–59.
22. Abd Elaziz M, Mirjalili S. A hyper-heuristic for improving the initial population of whale optimization algorithm. Knowl Based Syst. 2019;172:42–63.
23. Yildiz AR. A novel hybrid whale–Nelder–Mead algorithm for optimization of design and manufacturing problems. Int J Adv Manuf Technol. 2019;105(12):5091–104. https://doi.org/10.1007/s00170-019-04532-1.
24. Chakraborty S, Saha AK, Sharma S, Mirjalili S, Chakraborty R. A novel enhanced whale optimization algorithm for global optimization. Comput Ind Eng. 2021;153:107086. https://doi.org/10.1016/j.cie.2020.107086.
25. Khadanga RK, Kumar A, Panda S. A novel modified whale optimization algorithm for load frequency controller design of a two-area power system composed of PV grid and thermal generator. Neural Comput Appl. 2020;32:8205–16. https://doi.org/10.1007/s00521-019-04321-7.
26. Chen H, Yang C, Heidari AA, Zhao X. An efficient double adaptive random spare reinforced whale optimization algorithm. Expert Syst Appl. 2020;154:113018. https://doi.org/10.1016/j.eswa.2019.113018.
27. Chakraborty S, Sharma S, Saha AK, Chakraborty S. SHADE-WOA: A metaheuristic algorithm for global optimization. Appl Soft Comput. 2021;113:107866. https://doi.org/10.1016/j.asoc.2021.107866.
28. Yan Z, Zhang J, Tang J. Modified whale optimization algorithm for underwater image matching in a UUV vision system. Multimed Tools Appl. 2021;80:187–213. https://doi.org/10.1007/s11042-020-09736-2.
29. Kushwah R, Kaushik M, Chugh K. A modified whale optimization algorithm to overcome delayed convergence in artificial neural networks. Soft Comput. 2021;25:10275–86. https://doi.org/10.1007/s00500-021-05983-z.
30. Fuqiang L, Tongren Y, Hualing B, Ming F, Suxin W, Min H. A bilevel whale optimization algorithm for risk management scheduling of information technology projects considering outsourcing. Knowl Based Syst. 2022;235:107600. https://doi.org/10.1016/j.knosys.2021.107600.
31. Anitha J, Pandian SIA, Agnes SA. An efficient multilevel color image thresholding based on modified whale optimization algorithm. Expert Syst Appl. 2021;178:115003. https://doi.org/10.1016/j.eswa.2021.115003.
32. Chakraborty S, Saha AK, Chakraborty R, Saha M, Nama S. HSWOA: An ensemble of hunger games search and whale optimization algorithm for global optimization. Int J Intell Syst. 2021. https://doi.org/10.1002/INT.22617.
33. Chakraborty S, Sharma S, Saha AK, Saha A. A novel improved whale optimization algorithm to solve numerical optimization and real-world applications. Artif Intell Rev. 2022;1–112.
34. Lin X, Yu X, Li W. A heuristic whale optimization algorithm with the niching strategy for global multi-dimensional engineering optimization. Comput Ind Eng. 2022;171:108361.
35. Cao D, Xu Y, Yang Z, Dong H, Li X. An enhanced whale optimization algorithm with improved dynamic opposite learning and adaptive inertia weight strategy. Complex Intell Syst. 2022;1–29.
36. Chakraborty S, Saha AK, Chakraborty R, Saha M. An enhanced whale optimization algorithm for large-scale optimization problems. Knowl Based Syst. 2021;233:107543. https://doi.org/10.1016/j.knosys.2021.107543.
37. Kaur S, Awasthi LK, Sangal AL, Dhiman G. Tunicate Swarm Algorithm: a new bio-inspired based metaheuristic paradigm for global optimization. Eng Appl Artif Intell. 2020;90:103541.
38. Alsattar HA, Zaidan AA, Zaidan BB. Novel meta-heuristic bald eagle search optimisation algorithm. Artif Intell Rev. 2020;53(3):2237–64.
39. Fan Q, Chen Z, Zhang W, Fang X. ESSAWOA: enhanced whale optimization algorithm integrated with salp swarm algorithm for global optimization. Eng Comput. 2020;1–18.
40. Chakraborty S, Saha AK, Nama S, Debnath S. COVID-19 X-ray image segmentation by modified whale optimization algorithm

with population reduction. Comput Biol Med. 2021;139:104984. https://doi.org/10.1016/j.compbiomed.2021.104984.
41. Fan Y, Shao J, Sun G, Shao X. A self-adaption butterfly optimization algorithm for numerical optimization problems. IEEE Access. 2020;8:88026–41.
42. Gupta S, Deep K, Moayedi H, Foong LK, Assad A. Sine cosine grey wolf optimizer to solve engineering design problems. Eng Comput. 2021;37(4):3123–49.
43. Long W, Wu T, Liang X, Xu S. Solving high-dimensional global optimization problems using an improved sine cosine algorithm. Expert Syst Appl. 2019;123:108–26.
44. Sandgren E. Nonlinear Integer and Discrete Programming in Mechanical Design Optimization. J Mech Des. 1990;112(2):223. https://doi.org/10.1115/1.2912596.
45. Moghaddam SK, Buyya R, Ramamohanarao K. Performance-aware management of cloud resources: a taxonomy and future directions. ACM Comput Surv. 2019;52(4):1–37. https://doi.org/10.1145/3337956.
46. Amini Motlagh A, Movaghar A, Rahmani AM. Task scheduling mechanisms in cloud computing: a systematic review. Int J Commun Syst. 2020;33(6):e4302. https://doi.org/10.1002/dac.4302.
47. Chhabra A, Singh G, Singh Kahlon K. QoS-aware energy-efficient task scheduling on HPC cloud infrastructures using swarm-intelligence meta-heuristics. Comput Mater Cont. 2020;64(2):813–34. https://doi.org/10.32604/cmc.2020.010934.
48. Chhabra A, Singh G, Kahlon KS. Multi-criteria HPC task scheduling on IaaS cloud infrastructures using meta-heuristics. Clust Comput. 2021;24(2):885–918. https://doi.org/10.1007/s10586-020-03168-1.
49. Mohammad Hasani Zade B, Mansouri N, Javidi MM. SAEA: A security-aware and energy-aware task scheduling strategy by Parallel Squirrel Search Algorithm in cloud environment. Expert Syst Appl. 2021;176:114915. https://doi.org/10.1016/j.eswa.2021.114915.
50. Wei X. Task scheduling optimization strategy using improved ant colony optimization algorithm in cloud computing. J Ambient Intell Humaniz Comput. 2020. https://doi.org/10.1007/s12652-020-02614-7.
51. Fu X, Sun Y, Wang H, Li H. Task scheduling of cloud computing based on hybrid particle swarm algorithm and genetic algorithm. Clust Comput. 2021. https://doi.org/10.1007/s10586-020-03221-z.
52. Belgacem A, Beghdad-Bey K. Multi-objective workflow scheduling in cloud computing: trade-off between makespan and cost. Clust Comput. 2022;25(1):579–95. https://doi.org/10.1007/s10586-021-03432-y.
53. Mohammad Hasani Zade B, Mansouri N, Javidi MM. A two-stage scheduler based on New Caledonian Crow Learning Algorithm and reinforcement learning strategy for cloud environment. J Netw Comput Appl. 2022;202:103385. https://doi.org/10.1016/j.jnca.2022.103385.
54. Zhou Z, Li F, Zhu H, Xie H, Abawajy JH, Chowdhury MU. An improved genetic algorithm using greedy strategy toward task scheduling optimization in cloud environments. Neural Comput Appl. 2020;32(6):1531–41. https://doi.org/10.1007/s00521-019-04119-7.
55. Mohammad Hasani Zade B, Mansouri N, Javidi MM. Multi-objective scheduling technique based on hybrid hitchcock bird algorithm and fuzzy signature in cloud computing. Eng Appl Artif Intell. 2021;104:104372. https://doi.org/10.1016/j.engappai.2021.104372.
56. Abualigah L, Alkhrabsheh M. Amended hybrid multi-verse optimizer with genetic algorithm for solving task scheduling problem in cloud computing. J Supercomput. 2022;78(1):740–65. https://doi.org/10.1007/s11227-021-03915-0.
57. Abd Elaziz M, Attiya I. An improved Henry gas solubility optimization algorithm for task scheduling in cloud computing. Artif Intell Rev. 2021;54(5):3599–637. https://doi.org/10.1007/s10462-020-09933-3.
58. Shukri SE, Al-Sayyed R, Hudaib A, Mirjalili S. Enhanced multi-verse optimizer for task scheduling in cloud computing environments. Expert Syst Appl. 2021;168:114230. https://doi.org/10.1016/j.eswa.2020.114230.
59. Vila S, Guirado F, Lerida JL, Cores F. Energy-saving scheduling on IaaS HPC cloud environments based on a multi-objective genetic algorithm. J Supercomput. 2019;75(3):1483–95. https://doi.org/10.1007/s11227-018-2668-z.
60. Ni L, Sun X, Li X, Zhang J. GCWOAS2: Multiobjective task scheduling strategy based on gaussian cloud-whale optimization in cloud computing. Comput Intell Neurosci. 2021. https://doi.org/10.1155/2021/5546758.
61. Nadimi-Shahraki MH, Zamani H, Mirjalili S. Enhanced whale optimization algorithm for medical feature selection: a COVID-19 case study. Comput Biol Med. 2022;148:105858. https://doi.org/10.1016/j.compbiomed.2022.105858.
62. Zhou Z, Li F, Abawajy JH, Gao C. Improved PSO algorithm integrated with opposition-based learning and tentative perception in networked data centers. IEEE Access. 2020;8:55872–80. https://doi.org/10.1109/ACCESS.2020.2981972.
63. Assiri AS. On the performance improvement of butterfly optimization approaches for global optimization and Feature Selection. PLoS ONE. 2021;16(1):e0242612. https://doi.org/10.1371/journal.pone.0242612.
64. Hussien AG, Amin M, Abd El Aziz M. A comprehensive review of moth-flame optimisation: variants, hybrids, and applications. J Exp Theor Artif Intell. 2020;32(4):705–25. https://doi.org/10.1080/0952813X.2020.1737246.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.