Journal of Computer Science and Electrical Engineering
ISSN: 2663-1946
DOI: https://doi.org/10.61784/jcsee3058

WIRELESS SENSOR NETWORKS BASED ON IMPROVED DUNG BEETLE ALGORITHM

QiYuan Shen[1*], YunLong Xia[2]
[1]*College of Electronics and Information (Micro-Nano Technology College), Qingdao University, Qingdao 266071, Shandong, China.*
[2]*College of Electrical Engineering and Information Technology, Lanzhou University of Technology, Lanzhou 730050, Gansu, China.*
Corresponding Author: QiYuan Shen, Email: sqy_qdu@163.com

Abstract: To address the limitations of traditional Dung Beetle Optimization (DBO) in the wireless sensor network coverage problem, specifically like low convergence speed and susceptibility to local optima, this paper proposes an optimization scheme based on the Improved DBO algorithm (IDBO). The algorithm combines three key strategies: first, a Logistic chaos initialization strategy is used to generate a more optimal initial solution; second, a Levy flight strategy is introduced to enhance the global search capability; and finally, the search process is further optimized by using a dynamic nonlinear convergence factor to adaptively adjust the search step size. With the above improvements, the algorithm is significantly improved in global search performance. Experiments on the CEC2005 benchmark suite show that IDBO outperforms similar algorithms and approaches the global optimum. Finally, at last, the proposed IDBO algorithm is applied to the wireless sensor network coverage optimization problem for simulation experiments. The simulation results show that the improved IDBO algorithm improves the coverage of the network nodes by 4.9% compared to the basic DBO algorithm and enhances the overall performance of the network with good practicality, stability and robustness.
Keywords: Dung beetle optimization; Wireless sensor network coverage; Logistic chaotic initialization; Levy flight strategy; Dynamic nonlinear convergence factor

1 INTRODUCTION

In recent years, with the continuous progress of social science and technology, the Internet, and the continuous development of artificial intelligence technology, the Internet of Things (IoT) has become one of the popular research fields nowadays [1]. Wireless Sensor Networks (WSNs), as the core carrier of IoT technology, plays an irreplaceable role in the fields of environmental monitoring, smart city management, industrial automation, and disaster warning [2]. Its core objective is to realize efficient coverage and real-time monitoring of the target area through cooperative sensing and data transmission of distributed nodes [3]. However, the layout strategy of sensor nodes directly determines the coverage quality, energy efficiency and deployment cost of the network [4]. Especially in complex scenarios, how to optimize node layout through scientific methods and break through the limitations of traditional algorithms has become the focus and difficulty of WSNs research [5].

Currently, WSNs coverage optimization problems usually involve nonlinear constraints, multi-objective trade-offs and large-scale search spaces, and traditional deterministic algorithms (e.g., gradient descent, exhaustive search, etc.) can hardly meet the practical needs because they are prone to fall into the local optimal solutions and have low computational efficiency [6]. In recent years, meta-heuristic algorithms (e.g., genetic algorithm [7], particle swarm optimization [8], etc.) have gradually become an effective tool for solving such complex optimization problems by virtue of their global search capability and adaptability.

Coverage optimization for wireless sensor networks aims to maximize the coverage of the monitored area by rationally deploying nodes while taking into account energy consumption, network lifetime and cost. In the last five years, meta-heuristic algorithms have been widely used in academia and industry, for example, literature proposes a node localization optimization method based on the bat algorithm for solving the localization error problem due to environmental noise in WSNs [9]. The algorithm significantly improves the localization accuracy by dynamically adjusting the frequency and pulse firing rate, and reduces the average localization error by about 25% in complex environments Literature designs a dynamic cluster head selection strategy based on a differential evolutionary algorithm [10], which extends the network lifetime by optimizing the energy and distance parameters of the candidate nodes. Simulation results show that this algorithm improves the network survival time by about 30% compared to the traditional LEACH protocol Literature applies Teaching-Learning-Based Optimization Algorithm (TLBO) to the problem of energy balancing in WSNs to optimize the node duty cycle by simulating the “teacher-student” interaction mechanism [11]. Experiments show that the network energy distribution uniformity is improved by 40% and the overall lifetime is extended by 20%. Literature proposes the multi-objective Sparrow Search Algorithm (MSSA) to solve the trade-off problem between coverage voids and energy consumption in WSNs [12]. By introducing the Pareto optimal solution set, the coverage reaches 96% at 1000 node size while energy consumption is reduced by 15%. Literature combines the Sine Cosine Algorithm (SCA) with Q-learning and proposes an adaptive routing protocol (SCA-QL) to

optimize the data transmission paths in WSNs [13]. The scheme reduces end-to-end delay by 15% and improves link reliability in dynamic environments.

Dung Beetle Optimizer (DBO) is a hyper-inspired algorithm proposed by Prof. Bo Shen's team in 2023, inspired by the bionics of five different behaviors: ball-rolling, dancing, reproduction, foraging, and stealing. With its simplicity, small number of parameters, and powerful search and exploration capabilities, DBO is widely used in optimization tasks, including high-dimensional feature selection and data clustering. Despite its advantages, DBO has the disadvantages of slow convergence and convergence to a local optimum. Based on this drawback, this study introduces an improved DBO algorithm that combines the mechanisms of Logistic Chaos Mapping, Nonlinear Convergence Factor with levy flight strategy for improvement.

Logistic chaotic mapping, as a classical chaotic system with traversal, randomness and sensitivity to initial conditions, improves the diversity of initial solutions by about 20-30% in complex multi-peak function optimization, which greatly enhances the population diversity. In addition, the nonlinear convergence factor can dynamically adjust the exploration step size according to the iteration progress, with a large step size approaching the optimal solution quickly at the initial stage and a small step size finely adjusted at the later stage, thus accelerating the convergence speed, enhancing the global search capability and robustness, and ultimately improving the quality of the solution. levy flight strategy further enhances the population diversity, and the algorithm is capable of exploring a wider solution space through the long-step jumps, which reduces the local optimal trapping risk. Empirical results from the CEC2005 benchmark suite, using 23 test functions, verify that the IDBO algorithm greatly improves global optimization performance. This leads to a significant acceleration of convergence and improved accuracy. The experimental data show that the algorithm proposed in this study overcomes the above five algorithms in terms of computational complexity and node resource constraints, also makes up for the lack of dynamic topology adaptation, achieves multi-objective optimization capability and reduces the complexity of parameter tuning as compared to the five algorithms: GWO, COA, WOA, HHO and the original DBO algorithm. Therefore, the main advantage of IDBO algorithm applied in wireless sensor networks (WSN) is that it shows higher stability, accuracy and efficiency in finding the optimal solution, which highlights its superior overall performance.

2 BASIC DUNG BEETLE ALGORITHM

The Dung Beetle Optimization Algorithm classifies dung beetles into four species based on the division of labor in the dung beetle population: ball dung beetles, breeding dung beetles, little dung beetles, and thief dung beetles. 20%, 20%, 25%, and 35% of the dung beetles in the population are of the four species, respectively. Different types of dung beetles update their positions according to different position updating methods, and the position represents the solution to the problem.

2.1 Ball Dung Beetle

Dung beetles in nature will move in a straight line using natural light as a navigation when rolling a dung ball, the intensity of the light source will affect the path of the dung beetles' movement, at this time their position is updated as shown in (1). However, the natural environment is full of unknowns, when encountering impassable places, at this time they will redefine the target of their actions through unique behaviors, the dancing formula is shown in (2):

$$\begin{cases} x_i(t+1) = x_i(t) + \alpha \times k \times x_i(t-1) + b \times \Delta x \\ \Delta x = |x_i(t) - X^w| \end{cases} \tag{1}$$

$$x_i(t+1) = x_i(t) + \tan(\theta)|x_i(t) - x_i(t-1)| \tag{2}$$

Here, t denotes the current iteration number; $x_i(t)$represents the position of the i-th dung beetle at the t-th iteration; $k \in (0,0.2]$, indicates the deflection coefficient; $b \in (0,1)$; Δx quantifies light intensity, where X^W denotes the globally worst position; α is probabilistically assigned a value of 1 or -1.; When $\alpha = 1$, the direction remains unchanged, When $\alpha = -1$, it deviates from the original direction; $\theta \in [0, \pi]$, represents the deflection angle.

2.2 Breeding Dung Beetles

Dung beetles will roll balls of dung to a safe location to create a safe and secluded environment for female dung beetles to lay their eggs. This safe area is defined below:

$$\begin{cases} Lb^* = max(X^* \times (1-R), Lb) \\ Ub^* = min(X^* \times (1-R), Ub) \end{cases} \tag{3}$$

where, X^* denotes the current local optimal solution; the inertia weight $R = 1 - t/T_{max}$, where T_{max} is the maximum iteration count. Lb and Ub denote the lower and upper bounds of the optimization problem, respectively, while Lb^* and Ub^* define the dynamically adjusted boundaries for the breeding region. As illustrated in Equation (3), dynamically modifying these region boundaries plays a crucial role in preventing the algorithm from prematurely converging to local optima. Accordingly, the position of individuals is updated adaptively based on the following formulation:

$$B_i(t+1) = X^* + b_1 \times (B_i(t) - Lb^*) + b_2 \times (B_i(t) - Ub^*) \tag{4}$$

In Equation (4), $B_i(t+1)$ represents the position of the t-th breeding ball at the i-th interation, b_1 and b_2 are random vectors of size $1 \times d$, where d is the problem dimension.

2.3 Small Dung Beetles

Newly hatched dung beetles search for food, and the optimal foraging area is defined as:

$$Lb^b = max\ (X^b \times (1-R), Lb)$$
$$Ub^b = min\ (X^b \times (1-R), Ub) \tag{5}$$

In Equation (5), $X^b$is the global optimal position;Lb^b and Ub^bdenote the lower and upper bounds of the foraging region, respectively.
The position update rule for small dung beetles is defined as:

$$x_i(t+1) = x_i(t) + C_1 \times \left(x_i(t) - Lb^b\right) + C_2 \times \left(x_i(t) - Ub^b\right) \tag{6}$$

In Equation (6),$C_1$is a random number following a normal distribution, and $C_2$is a random vector within the interval (0,1).

2.4 Thieving Dung Beetles

Thieving dung beetles like to get something for nothing, and their location is updated in the following way:

$$x_i(t+1) = X^b + S \times g \times (|x_i(t) - X^*| + \left|x_i(t) - X^b\right|) \tag{7}$$

Here, g is a random vector following a normal distribution with a size of $1 \times d$; and S is a constant.

3 IMPROVED DUNG BEETLE ALGORITHM

3.1 Population Initialization via Logistic Chaotic Mapping

The standard DBO algorithm uses a function to initialize the population, and the population generated by this method has a large randomness and will not cover the whole space uniformly, which reduces the algorithm's search efficiency. Therefore, in this paper, we adopt the method of initializing the population with chaotic mapping to improve the randomness of the algorithm's search, so that it can better traverse the whole search space.
Logistic chaotic mapping is publicized as follows:

$$x_{n+1} = \mu \cdot x_n \cdot (1 - x_n) \tag{8}$$

In Equation (8), $x_n \in [0,1]$ denotes the system state at step n, and μ is a control parameter typically within [0,4], When $\mu \in (3.56995,4]$,the system exhibits chaotic behavior.

3.2 Nonlinear Convergence Factors

In dung beetles in reproductive behavior, where, as the iterative process of adaptation decreases. With the gradual decrease, the influence of the initial boundary individual will become smaller and smaller, and in the late iteration, the individual will be oriented to the optimal position obtained from its own position in the previous iteration for local optimization, although this strategy improves the algorithm's optimization ability to a certain extent, however, from the above analysis, it can be seen that the method only takes into account the positive influence of the dung beetle's optimal individual on its movement, and ignores the original boundary brought about by the Positive influence, i.e., when the optimal position of the dung beetle's individual is near the original boundary, the strategy may still cause the individual to miss the better adapted position, therefore, the present improvement proposes an improved weight formula, and uses a kind of random number to perturb the weights, so that the initial boundary can be selected at a later stage. The details are as follows.

$$R = 1 - \text{rand} \times \sqrt{\frac{t}{M}} \tag{9}$$

In Equation (9), tdenotes the current iteration count,, M is the maximum iteration count, and randis a uniformly distributed random number within [0, 1], This mechanism introduces randomness to avoid local optima and enhance global search capability.

3.3 Levy Flight Strategy

In the original algorithm, the original formula corresponding to the rolling ball moves in the form of a straight line, which is not only unable to traverse the whole globe efficiently, but also does not have a deep search for the localization. Therefore the rolling ball phase formula of the dung beetle algorithm is improved.
The Levy flight strategy is a widely used technique to enhance the global search capability of optimization algorithms. It allows for dynamic adaptation to the search space and helps maintain search diversity, thereby improving the algorithm's ability to explore globally. In the context of Dung Beetle Optimization (DBO), Levy flight is typically

integrated into the position update mechanism to strengthen the global exploration performance of dung beetles. The improved position update formula incorporating Levy flight is expressed as follows:

$$x_i(t+1) = X^b + S \times g \times \left(|x_i(t) - X^*| + \text{Levy}(\beta) \cdot |x_i(t) - X^b|\right) \quad (10)$$

In Equation (10), $X_i(t)$ represents the position of the $i-th$ individual at iteration t, and X_{best} is the best-known position. α is a scaling factor controlling step size. Levy flight is characterized by step lengths following the Levy distribution with parameter β, enabling long-distance jumps in the search space to improve global.

After simulating the high-dimensional multi-peak functions such as Ackley and Griewank, it is found that the global optimal discovery probability of the DBO is significantly improved and the convergence speed is accelerated after the introduction of Levy flights. By integrating the content from the previous section with the DBO algorithm, the pseudo-code of the IDBO algorithm is presented as follows:

Algorithm 1 IDBO

Require: The maximum iterations T_{max}, the size of the particle's population *N*.
1: Utilize Logistic chaos mapping for population initialization $i \leftarrow 1, 2, ..., N$ and define its relevant parameters
2: **while** ($t \leq T_{max}$) do
3: **for** $i \leftarrow 1$ to *N* **do**
5: Synthesize novel individual positions utilizing (13) and (14), determine its fitness value f^b for the dung beetle population
4: **if** *i* == ball-rolling dung beetle **then**
5: λ = rand(1);
6: **if** $\lambda < 0.9$ **then**
8: Update the ball-rolling dung beetle's position by using(1);
9. **else**
10: Update the ball-rolling dung beetle's position by using(2);
11: **end if**
12: **end if**
13: Revise the position of the brood ball according to (4);
14: Enforce the constraints on the new position using the upper and lower bounds defined by (3) and (9);
15. Modify the position of the subordinate dung beetle in accordance with (5) and (9);
16: Update the position of the small dung beetle according to (6).;
17: Adjust the position of the thief dung beetle using (10)
18: Update the global best position X^b and the worst position X^w
19: **if** the newly generated position is better than before **then**
20: Update it;
21: **end if**
22: $t = t + 1$;
23: **end while**
24: **return** X^b and f^b

4 EXPERIMENT AND COMPREHENSIVE ANALYSIS

The simulations were conducted on a 64-bit Windows 11 operating system. All analyses were executed using MATLAB 2023b on a computing platform powered by an AMD Ryzen 7 4800H processor (2.30 GHz) and equipped with 16 GB of RAM.

4.1 Test Functions and Parameter Settings

The CEC 2005 [14] benchmark suite was utilized to assess the performance of the proposed IDBO algorithm. This suite encompasses four distinct categories of functions: unimodal, multimodal, hybrid, and composite, each designed to test specific aspects of an algorithm's capabilities. By incorporating this variety, the benchmark framework offers a thorough and systematic evaluation of the algorithm's performance and generalizability. These diverse test functions serve as a robust experimental platform for analyzing the scalability and effectiveness of optimization algorithms, allowing for a more precise assessment of IDBO's adaptability across various problem types..

4.2 Comparison with other Algorithms and Parameter Settings

The performance of the IDBO algorithm was systematically compared against 6 well-known algorithms. This includes the Particle swarm optimization (PSO) [15], Genetic algorithm (GA) [16], Grey wolf optimizer (GWO) [17], Whale Optimization Algorithm [18] (WOA), Arctic puffin optimization [19] (APO), original Dung Beetle optimization algorithm (DBO) [20].

Table 1 summarizes the parameter settings for the seven different MH algorithms. For each algorithm, 30 independent simulation runs were performed, each limited to 500 iterations with a population size of 30. The performance outcomes

of each algorithm were documented, including the mean (denoted as Ave) and standard deviation (Std). To enable a clear comparison across algorithms, the best results obtained under each test condition are highlighted in bold within the table. This visual emphasis effectively identifies the top-performing algorithms for each specific scenario.

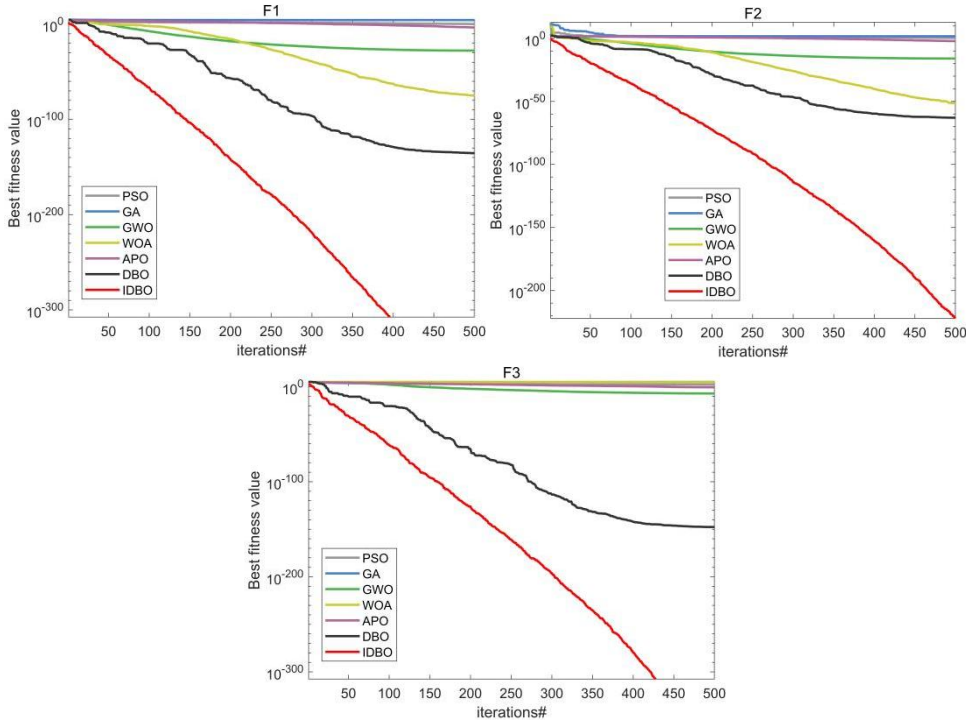
Table 1 Parameter Configurations for Competing Algorithms

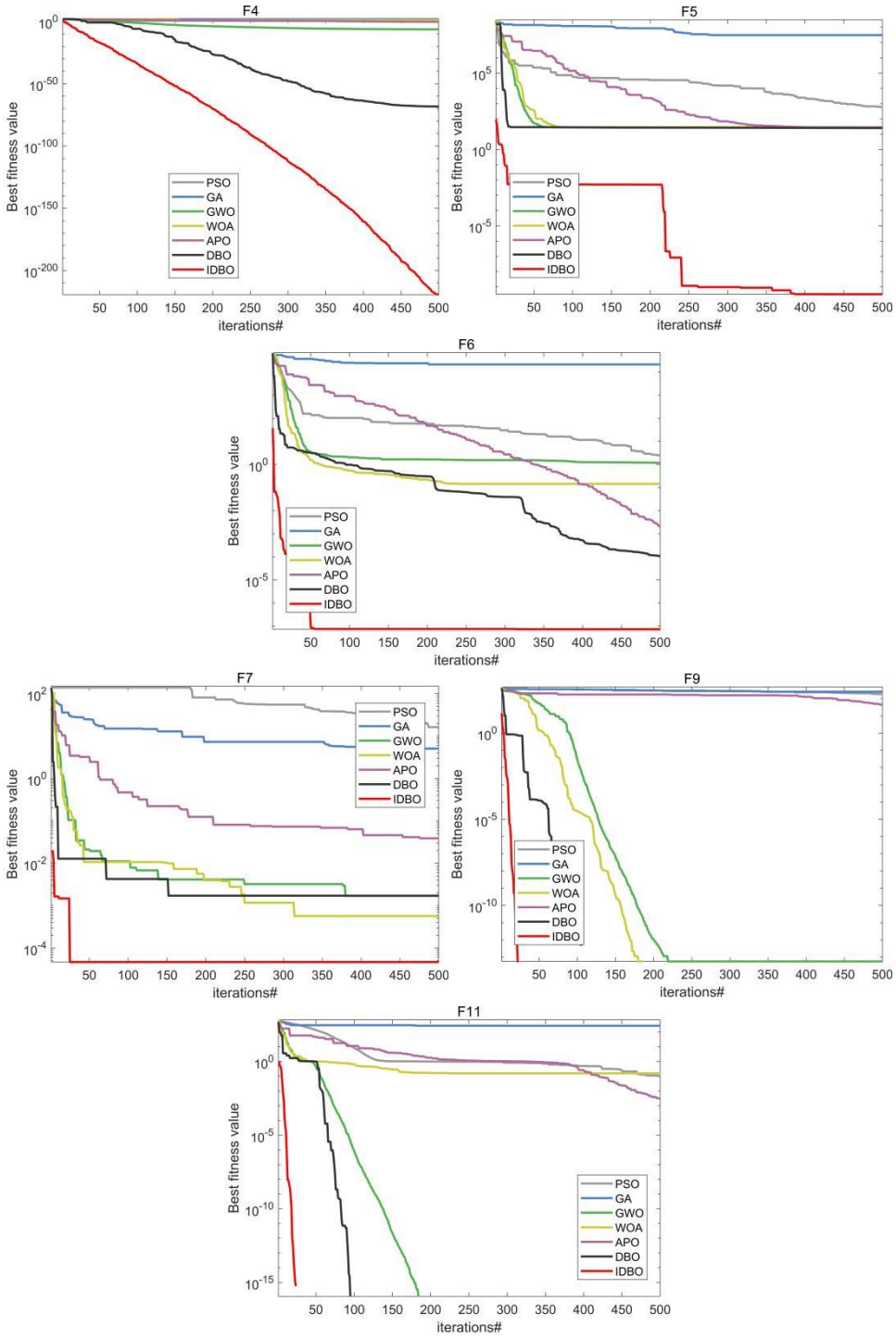
Algorithms	Parameter	Value
PSO	C_1, C_2	2, 2
GA	Cross, mutation	0.2, 0.1
GWO	α	[0,2]
WOA	a, a2, b	[0,2], [- 1,-2], 1
APO	F, C	0.5, 0.5
DBO	RDB, EDB, FDB, SDB	6, 6, 7, 11
IDBO	RDB, EDB, FDB, SDB	6, 6, 7, 11

4.3 Quantitative Evaluation

In this section, the performance of the IDBO algorithm is rigorously evaluated using the CEC2005 benchmark suite. To ensure consistency, experimental parameters were standardized: the population size was set to 30, the maximum number of iterations to 500, and each algorithm was executed over 30 independent runs. Table 2 presents a detailed comparison of the results obtained by all competing algorithms, reporting both the average (Ave) and standard deviation (Std) values. The advantages of IDBO are underscored through comprehensive statistical analysis. The first row of the performance summary displays the Friedman mean ranks for each algorithm, while the second row provides the final rankings based on the Friedman test. To highlight the most effective results, the best-performing values are bolded in the table. Additionally, Figure 1 illustrates the convergence curves of the algorithms, providing a visual representation of their optimization efficiency and iterative progress toward the global optimum. The results collectively demonstrate the robustness and adaptability of the IDBO algorithm across a range of optimization scenarios.

The detailed experimental outcomes are presented in Table 2, with the top-performing results among the seven algorithms highlighted in bold. Notably, IDBO consistently achieved the best or near-best optimization results across all test functions and never ranked last in any case. From a statistical perspective, IDBO secured the highest rank, while DBO followed with a Friedman test mean of 2.54. This not only underscores the inherent strength of the original DBO algorithm but also emphasizes the necessity for further algorithmic enhancement. These findings clearly demonstrate that the proposed IDBO outperforms several state-of-the-art algorithms within its category. Figure 1 displays the convergence curves across various dimensions, revealing that IDBO achieves the fastest convergence rate, superior solution accuracy, and exhibits the capability to escape local optima during the iteration process. In summary, the experimental results strongly confirm the effectiveness and superiority of IDBO as a robust and high-performance optimization algorithm.




F4
F5
F6
F7
F9
F11
Best fitness value
iterations#
PSO
GA
GWO
WOA
APO
DBO
IDBO

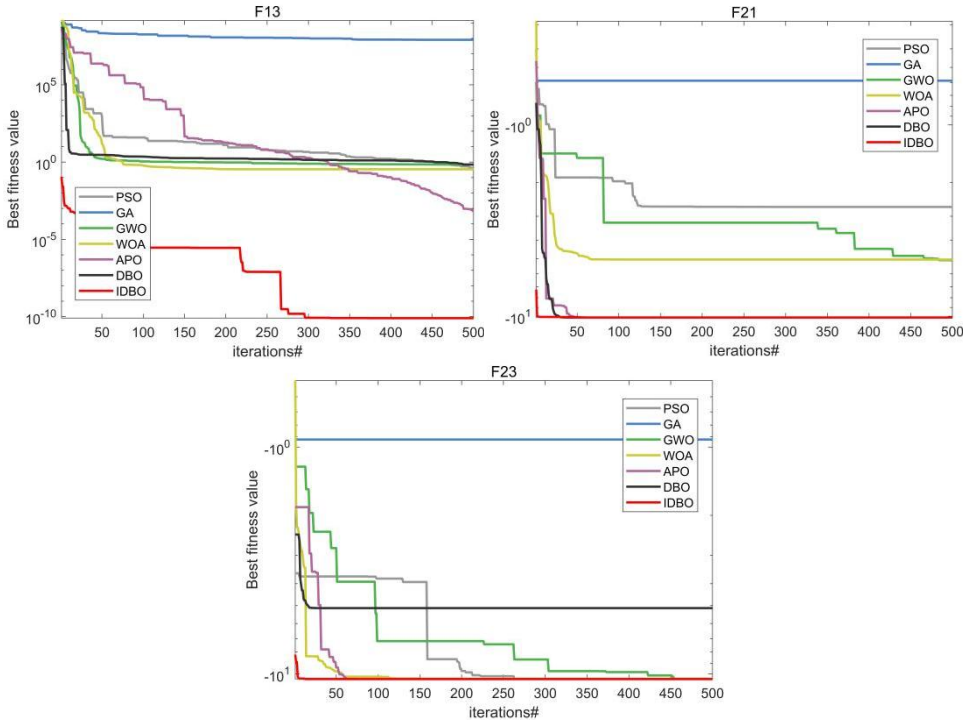


Figure 1 Convergence Curves of Different Algorithms

Table 2 Comparison of Results on CEC 2017 (Dim=30)

ID	Metric	PSO	GA	GWO	WOA	APO	DBO	IDBO
CEC2005-F1	Ave	2.1899E+00	2.0311E+04	1.1309E-27	2.7988E-71	8.0822E-04	3.2967E-111	0.0000E+00
	Std	1.1531E+00	6.7213E+03	1.5597E-27	1.5276E-70	8.7209E-04	1.8056E-110	0.0000E+00
CEC2005-F2	Ave	4.3051E+00	5.5969E+01	8.8141E-17	8.2682E-51	9.3412E-03	8.4919E-56	9.3684E-213
	Std	1.3590E+00	1.0813E+01	6.3964E-17	2.9380E-50	4.3593E-03	4.1934E-55	0.0000E+00
CEC2005-F3	Ave	1.7406E+02	5.4615E+04	9.5897E-06	4.2889E+04	2.1652E-01	1.1754E-23	0.0000E+00
	Std	4.2307E+01	1.4129E+04	1.9828E-05	1.5195E+04	1.4867E-01	6.4381E-23	0.0000E+00
CEC2005-F4	Ave	1.9450E+00	7.2056E+01	8.2342E-07	5.5313E+01	7.2780E-01	2.2579E-58	2.2095E-210
	Std	2.7640E-01	8.1057E+00	7.9958E-07	3.0214E+01	2.2914E-01	1.0107E-57	0.0000E+00
CEC2005-F5	Ave	1.1232E+03	2.1224E+07	2.7063E+01	2.7830E+01	2.4330E+01	2.5715E+01	3.6246E-04
	Std	7.1871E+02	1.7317E+07	6.8524E-01	4.6210E-01	9.0538E+00	1.8181E-01	6.7064E-04
CEC2005-F6	Ave	2.1401E+00	2.2294E+04	6.8072E-01	4.3627E-01	2.1814E-03	6.0633E-04	8.0633E-07
	Std	9.9516E-01	7.4702E+03	3.1907E-01	2.9702E-01	1.5357E-03	9.2715E-04	1.6071E-06
CEC2005-F7	Ave	2.2886E+01	1.1649E+01	1.9375E-03	3.4420E-03	2.9600E-02	1.3904E-03	3.5056E-05
	Std	1.9494E+01	8.3489E+00	1.3652E-03	3.6722E-03	1.1486E-02	1.1834E-03	2.9257E-05
CEC2005-F8	Ave	-6.3628E+03	-2.1164E+03	-6.3408E+03	-1.0706E+04	-6.7605E+03	-8.4969E+03	-1.2569E+04
	Std	1.1316E+03	5.0442E+02	7.4970E+02	1.5905E+03	1.2927E+03	1.8384E+03	8.4815E-02
CEC2005-F9	Ave	1.6674E+02	2.6023E+02	3.9434E+00	0.0000E+00	9.7766E+01	0.0000E+00	0.0000E+00
	Std	2.9415E+01	4.8325E+01	4.4291E+00	0.0000E+00	4.0324E+01	0.0000E+00	0.0000E+00
CEC2005-F10	Ave	2.6533E+00	1.9770E+01	1.0205E-13	4.2337E-15	7.3861E-03	4.4409E-16	4.4409E-16
	Std	4.2985E-01	5.3143E-01	1.8202E-14	2.0723E-15	3.3874E-03	0.0000E+00	0.0000E+00
CEC2005-F11	Ave	1.3633E-01	2.1844E+02	1.9540E-03	9.3348E-03	8.7823E-03	0.0000E+00	0.0000E+00
	Std	4.8881E-02	8.2964E+01	6.0843E-03	3.5852E-02	1.1186E-02	0.0000E+00	0.0000E+00
CEC2005-F12	Ave	7.1152E-02	2.4170E+07	4.9462E-02	2.5163E-02	1.3885E-02	3.1131E-04	5.5409E-08
	Std	1.1945E-01	3.2642E+07	2.5965E-02	3.0658E-02	3.5847E-02	1.6001E-03	9.4659E-08
CEC2005-F13	Ave	5.4968E-01	8.7242E+07	6.6725E-01	5.0125E-01	4.7669E-03	7.7574E-01	1.1949E-06
	Std	2.0763E-01	8.0554E+07	2.3682E-01	2.1816E-01	5.3068E-03	3.9018E-01	2.7725E-06
CEC2005-F14	Ave	3.3314E+00	1.1788E+00	4.0321E+00	2.8335E+00	9.9800E-01	1.0311E+00	9.9800E-01
	Std	2.5520E+00	5.2228E-01	4.1917E+00	2.9226E+00	0.0000E+00	1.8148E-01	3.3173E-10
CEC2005-F15	Ave	8.8995E-04	1.1941E-02	3.7659E-03	7.1993E-04	3.0749E-04	8.5180E-04	3.2056E-04
	Std	1.0151E-04	1.3163E-02	7.5513E-03	3.6179E-04	1.4057E-19	4.4954E-04	1.5452E-05
CEC2005-F16	Ave	-1.0316E+00	-9.4964E-01	-1.0316E+00	-1.0316E+00	-1.0316E+00	-1.0316E+00	-1.0316E+00
	Std	4.7012E-16	1.1049E-01	1.3815E-08	5.5790E-10	6.7752E-16	5.7578E-16	5.6453E-11
CEC2005-F17	Ave	3.9789E-01	7.0195E+01	3.9793E-01	3.9789E-01	3.9789E-01	3.9789E-01	3.9789E-01
	Std	0.0000E+00	7.9426E+00	1.5275E-04	9.1911E-06	0.0000E+00	3.2434E-16	1.8070E-07
CEC2005-F18	Ave	3.0000E+00	7.5344E+00	3.0000E+00	3.0000E+00	3.0000E+00	3.9000E+00	3.0065E+01
	Std	4.5944E-15	1.0296E+01	5.7361E-05	1.1427E-04	1.3143E-15	4.9295E+00	3.2483E-01
CEC2005-F19	Ave	-3.8628E+00	-3.3348E+00	-3.8609E+00	-3.8570E+00	-3.8628E+00	-3.8612E+00	-3.8037E+00
	Std	2.1615E-15	3.3167E-01	3.2182E-03	6.6394E-03	2.7101E-15	3.2065E-03	5.2125E-02

ID	Metric	PSO	GA	GWO	WOA	APO	DBO	IDBO
CEC2005-F20	Ave	-3.2863E+00	-1.5596E+00	-3.2609E+00	-3.2451E+00	-3.3180E+00	-3.2394E+00	-3.0982E+00
	Std	5.5415E-02	5.4189E-01	7.3342E-02	9.5655E-02	2.1707E-02	8.3431E-02	1.2513E-01
CEC2005-F21	Ave	-6.9735E+00	-7.7663E-01	-9.3971E+00	-8.3577E+00	-1.0153E+01	-7.4455E+00	-1.0153E+01
	Std	3.1472E+00	6.4577E-01	1.9938E+00	2.5854E+00	7.0670E-15	2.5750E+00	5.2776E-04
CEC2005-F22	Ave	-9.0125E+00	-1.2235E+00	-1.0147E+01	-8.1392E+00	-1.0403E+01	-8.9888E+00	-1.0403E+01
	Std	2.6169E+00	5.5349E-01	1.3940E+00	2.8277E+00	9.3299E-16	2.3810E+00	3.2399E-04
CEC2005-F23	Ave	-9.5982E+00	-1.2363E+00	-1.0356E+01	-6.6610E+00	-1.0536E+01	-9.3877E+00	-1.0536E+01
	Std	2.1451E+00	5.4666E-01	9.7875E-01	3.5576E+00	2.0600E-15	2.3627E+00	1.3439E-03
Friedman average		4.62	6.79	4.25	4.26	2.93	2.61	2.54
Overall Rank		6	7	4	5	3	2	1

5 MATHEMATICAL MODELING OF WIRELESS SENSOR NETWORK COVERAGE

5.1 Perceptual Models

In order to reduce the complexity of the perception model, often such problems will be simplified to a binary model, also known as Boolean perception model [21]. In the two-dimensional plane, the sensing range of a sensor node is a circular area formed by the sensor node as the center and the radius as a circle, the probability of being sensed within this range is 1, and the area beyond this range is considered unperceivable with a probability of 0. The mathematical model is:

$$\mathrm{P}(r,S_i)=\begin{cases}1 & d(r,S_i)\le R\\ 0 & \text{else}\end{cases} \tag{11}$$

In Equation (1), $\mathrm{P}(\mathrm{r},\mathrm{s_i})$ denotes the probability that sensor $\mathrm{s_i}$ detects point r and $\mathrm{d}(\mathrm{r},\mathrm{s_i})$ is the Euclidean distance between $\mathrm{s_i}$ and r.

5.2 Coverage

Most of the coverage strategy studies for wireless sensor networks are based on solving for area coverage based on area grids. The area coverage of a sensor network is a commonly used quantitative metric, which is the basis for comparing the advantages and disadvantages of optimization algorithms.

It is assumed that there are sensors in the sensor network, and its set is denoted as:

$$S=\{s_i\}, i=1,\cdots,n \tag{12}$$

In Equation (2): S denotes the sensor network; $\mathrm{s_i}$ is the $\mathrm{i-th}$ sensor node; and n represents the total number of nodes.

Area coverage is defined as the ratio of the effective coverage area of the target area by all the sensor nodes to the total area of the target area, and is expressed as [22]:

$$\eta=\frac{\sum_{i=0}^{n}\Phi_{s_i}}{A} \tag{13}$$

In Equation (3): η is the area coverage rate; $\Phi_{\mathrm{s_i}}$ is the sensing area of the i-th sensor, and A is the total area of the target region.

A point in the target area may be sensed by different nodes in the network, then the probability of the point being sensed by the sensor network is:

$$P(r,S)=1-\prod_{i=1,\cdots,n}[1-P(r,s_i)] \tag{14}$$

Assume there are m points in the target area. The total coverage rate of the sensor network over all detected points is defined as:

$$C_{(s)}=\frac{\sum_{r=1}^{n}P(r,S)}{A} \tag{15}$$

In Equation (5): $\mathrm{C(s)}$ denotes the total coverage rate of sensor network S over all detected points in the target area.

The optimization objective of this study is to deploy a fixed number of sensor nodes in the target area, and to maximize the coverage rate $\mathrm{C(s)}$ of the WSN by applying intelligent algorithms to optimize node layout.

5.3 Coverage Optimization based on IDBO Algorithm

In wireless sensor networks, the region is assumed to be a two-dimensional plane. Satisfaction:

(1) All nodes use Boolean sensing model.

(2) In the wireless sensor network coverage, all sensor nodes are isomorphic with the same communication radius Rc and perception radius Rs. To ensure the connectivity of the whole network, this mathematical relationship is set as $\mathrm{Rc} \geq 2\mathrm{Rs}$.

5.3.1 Experimental simulation and result comparison

In order to verify the superiority of the proposed improved Sparrow algorithm in wireless sensor network applications, this paper uses MATLAB2023b to simulate the proposed algorithm IDBO, and compares and analyzes it with the WSNs deployed by the other six algorithms under the same conditions. The experimental parameters are shown in Table 3, and the number of iterations is set to 200.

Table 3 Experimental Parameters

Sensing field	Nodes N	Communication radius R	Perceived radius r
50 m*50 m	30	10 m	5 m

The distribution of optimized nodes at 200 iterations is shown in Fig 2. As can be seen from Fig 2, compared with the other six high-performance algorithms, the coverage of WSN optimized by the improved IDBO is more comprehensive and nearly achieves full coverage; at the same time, from the coverage iteration curve in Fig 3, it can be seen that IDBO is significantly better than the other algorithms, and its convergence speed is faster and more stable, and it achieves the highest coverage rate of 81.84% within 200 iterations, which demonstrates a very high optimization efficiency. efficiency. In contrast, DBO, WOA and PSO are the next best algorithms, which have higher convergence coverage, but still do not surpass IDBO. In the graph, DBO also shows strong optimization ability, with faster convergence and 77.96% coverage, which indicates that its performance in the WSN coverage problem is more stable and efficient. WOA is also competitive, with faster convergence but falls into the local optimum too early, and reaches the maximum coverage of 81.84% within 200 iterations. prematurely falls into the local optimum, and finally reaches a coverage rate of 75.60%. In contrast, the performance of GA, POA and GWO is significantly weaker, which not only have slow convergence speed, but also have coverage rates of 68.64%, 67.68% and 73.96%, respectively, indicating that they fail to improve the coverage rate sufficiently in the optimization process, and their ability to adapt to complex problems is poor. On the other hand, IDBO almost always breaks through the coverage peak quickly, and still keeps the rising trend even in the late iteration, which shows that it has a strong ability to jump out of the local optimum. The optimization results for 500 iterations are shown in Table 4. By analyzing the performance of each algorithm, IDBO shows excellent optimization ability, with an optimal coverage of 81.84%, which ranks first among all algorithms and significantly outperforms other competing algorithms.

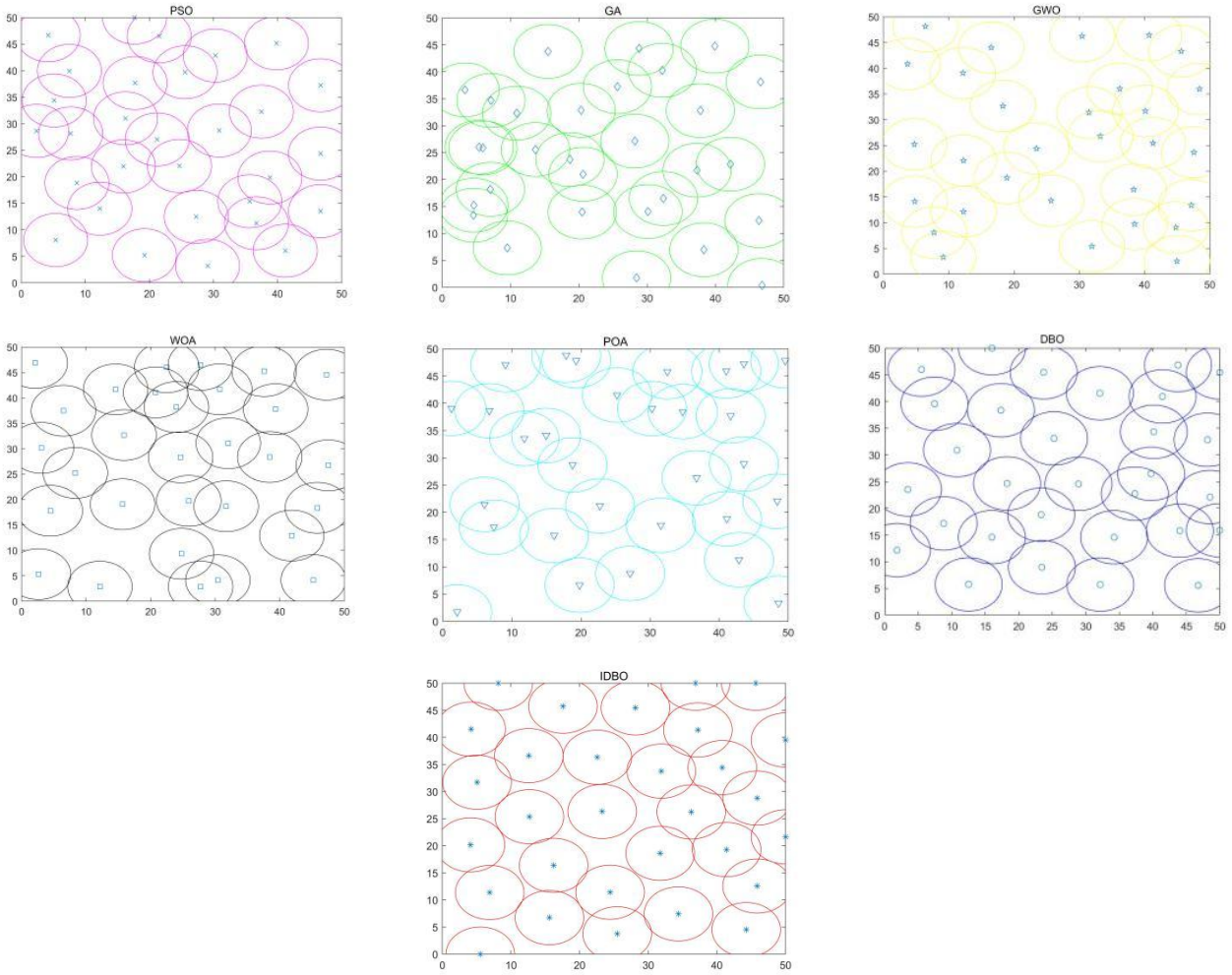


Figure 2 Optimization Node Distribution Diagram (Iteration 200 Times)

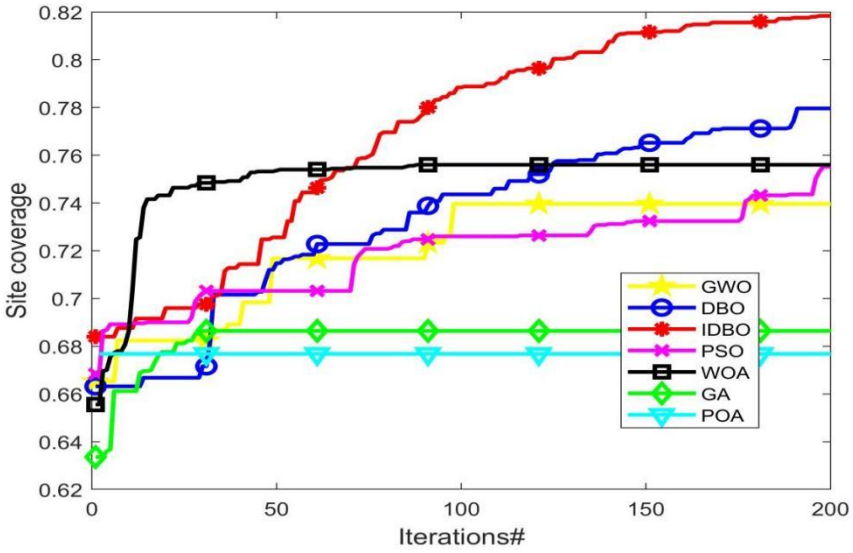


Figure 3 Coverage Optimization Line Chart (Iteration 500 Times)

Table 4 Coverage Results (iteration 500 times)

Algorithm	PSO	GA	GWO	WOA
fraction of coverage/%	75.52%	68.64%	73.96%	75.60%
Algorithm	POA	DBO	IDBO	
fraction of coverage/%	67.68%	77.96%	81.84%	

6 CONCLUSION

To tackle the challenges arising from random node deployment in the wireless sensor network (WSN) coverage problem, this study introduces an enhanced dung beetle optimization algorithm (IDBO). The algorithm combines three key strategies: first, a Logistic Chaos initialization strategy is used to generate a better initial solution; second, a Levy flight strategy is introduced to enhance the global search capability; and finally, a dynamic nonlinear convergence factor adaptively adjusts the search step size, thus further optimizing the search process. These improvements make IDBO not only have faster convergence speed, but also improve its robustness and adaptability in complex search space, effectively avoiding the local optimal solution problem.

Experimental results on the CEC2005 benchmark test set show that IDBO exhibits excellent optimization search performance in most of the test functions, with the first Friedman ranking, fully demonstrating its potential as a high-performance optimizer. To further verify the effectiveness of the proposed algorithm, IDBO is applied to solve the wireless sensor network (WSN) coverage problem. Experimental results indicate that, in comparison to the traditional DBO and other advanced algorithms such as GA, PSO, GWO, WOA, and POA, IDBO achieves notable improvements in reducing node redundancy, expanding coverage area, and enhancing coverage efficiency. Moreover, it demonstrates superior adaptability and faster convergence, highlighting its robustness in addressing complex WSN deployment scenarios. In addition, IDBO has better coverage efficiency performance compared to other intelligent optimization algorithms. Therefore, IDBO has excellent optimization performance in the wireless sensor network coverage problem, with the dual advantages of reducing network coverage cost and improving coverage rate. Future research will focus on further improving the convergence speed of the algorithm while ensuring high coverage.

COMPETING INTERESTS

The authors have no relevant financial or non-financial interests to disclose.

REFERENCES

[1] Li X, Li B, Li Y. Per-connection vs. service subscription: A study on service charging model of IoT platform empowering manufacturers to upgrade product intelligence. Nankai Management Review, 2024: 1–27.
[2] Zhang J. Research on data aggregation algorithm for wireless sensor networks based on node heterogeneity. Hebei: Hebei University, 2024.

[3] Chen G, Wan W. Trust assessment and node selection strategy based on distributed cloud node task collaboration. Guangdong Communication Technology, 2023, 43(12): 45–50.
[4] Zhu J. Research on node layout, localization, and mobile node path planning problems in WSN. Shenyang: Northeastern University, 2010.
[5] Huang Q, Lai C. Research on communication route planning algorithm for WSNs based on fuzzy affiliation optimization algorithm. Automation Technology and Application, 2024: 1–8.
[6] Song J, Hu Y M, Luo Y B. Comparative analysis of swarm intelligent optimization algorithms based on WSN network coverage optimization problem. Journal of Dali University, 2024, 9(12): 65–73.
[7] Dong H, Pan F. Constant tension fuzzy control of lithium battery winding based on genetic algorithm. Automation and Instrumentation, 2025, 40(2): 43–48.
[8] Wei S, Jia H, Cao J, et al. Prediction of heat transfer performance of buried pipe based on particle swarm optimization algorithm. District Heating, 2025(1): 107–116.
[9] Wang J, et al. An improved bat algorithm for node localization in wireless sensor networks. Applied Soft Computing, 2020, 89: 106156.
[10] Li X, et al. Energy-efficient cluster head selection in WSNs using differential evolution. Engineering Applications of Artificial Intelligence, 2021, 97: 104041.
[11] Kumar S, Singh P. TLBO-based energy balancing in wireless sensor networks for IoT applications. Ad Hoc Networks, 2022, 135: 102952.
[12] Chen H, et al. Multi-objective sparrow search algorithm for coverage and energy optimization in WSNs. IEEE Sensors Journal, 2023, 23(5): 5123–5135.
[13] Zhang L, et al. SCA-based Q-learning routing protocol for energy-efficient wireless sensor networks. Computer Networks, 2021, 194: 108133.
[14] Yao X, Liu Y, Lin G. Evolutionary programming made faster. IEEE Transactions on Evolutionary Computation, 1999, 3(2): 82–102.
[15] Kennedy J. Particle swarm optimization// Proceedings of the IEEE International Conference on Neural Networks (Perth, Australia). IEEE, 1995, 4: 1942–1948.
[16] Goldberg D, Holland J. Genetic algorithms and machine learning. Machine Learning, 1988, 3(2): 95–99.
[17] Mirjalili S, Mirjalili S M, Lewis A. Grey wolf optimizer. Advances in Engineering Software, 2014, 69: 46–61.
[18] Mirjalili S, Lewis A. The whale optimization algorithm. Advances in Engineering Software, 2016, 95: 51–67.
[19] Wang W, Tian W, Xu D, et al. Arctic puffin optimization: A bio-inspired metaheuristic algorithm for solving engineering design optimization. Advances in Engineering Software, 2024, 195: 103694.
[20] Xue J, Shen B. Dung beetle optimizer: A new meta-heuristic algorithm for global optimization. The Journal of Supercomputing, 2023, 79: 7305–7336.
[21] Liu J S, Li W X, Li Y. LWMEO: An efficient equilibrium optimizer for complex functions and engineering design problems. Expert Systems with Applications, 2022, 198: 116828.
[22] Wang H, Zhang X, Lu H. Sensor coverage optimization strategy based on geometric coverage algorithm. Computer Applications Research, 2017, 34(8): 2478–2482.