Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 8808575, 15 pages
https://doi.org/10.1155/2022/8808575

Research Article

Improved Ant Lion Optimizer for Coverage Optimization in Wireless Sensor Networks

Wei Chen,[1,2] **Panlong Yang**,[2] **Wei Zhao**,[3] **and Linna Wei**[3]

[1]*Department of Computer Information, Suzhou Vocational and Technical College, Suzhou, 234099 Anhui, China*
[2]*School of Computer Science and Technology, University of Science and Technology of China, Hefei, 230026 Anhui, China*
[3]*School of Computer Science and Technology, Anhui University of Technology, Maanshan, 243032 Anhui, China*

Correspondence should be addressed to Linna Wei; linnawei@ahut.edu.cn

Received 13 January 2022; Revised 18 July 2022; Accepted 25 July 2022; Published 16 August 2022

Academic Editor: Andrea Marin

Coverage optimization is an important research topic in wireless sensor networks (WSNs). By studying the coverage optimization problem, the coverage rate of the network can be improved, and the number of redundant sensors can be reduced. In order to improve the coverage performance of wireless sensor networks, we propose an improved ant lion optimizer (IALO) to solve the coverage optimization problem in a WSN. Firstly, we give a network coverage optimization model with the objective of maximizing network coverage rate. Secondly, we alternately execute Cuckoo Search (CS) and Cauchy mutation to update the positions of the ants to enhance population diversity and accelerate convergence speed. Then, we introduce differential evolution (DE) to mutate the population of antlions to improve the convergence accuracy of our algorithm. We compare IALO with the original ant lion optimizer (ALO) and other algorithms on 9 benchmark functions to verify its effectiveness. Finally, IALO is applied to the coverage optimization in wireless sensor networks. Simulation results show that, compared with previous works, IALO provides higher coverage rate, makes the sensor distribution more uniform, and effectively reduces the deployment cost.

1. Introduction

A wireless sensor network is a distributed system with a large number of sensor nodes [1]. In recent years, with the development of wireless communication technology, WSNs have been widely used in target tracking [2], environment monitoring [3], military applications [4], medical applications [5], etc. As network coverage affects the service quality of a WSN, sensor distribution becomes a key problem in the research fields of WSN. Because of the limitations of the physical environment, sensors are usually randomly deployed in the monitor area in practical applications. Consequently, the blank areas with no sensor and the areas with redundant sensors coexist in the monitor field. Therefore, it is necessary to adaptively adjust the positions of sensors in a WSN to improve its coverage rate, reduce the coverage blind area, and reduce the deployment cost.

The swarm intelligence optimization algorithm provided new ways to solve the coverage optimization problem in WSNs. In recent years, many researchers have applied it to the coverage control of WSN and studied its performance. The authors in [6] used particle swarm optimization (PSO) to optimize network coverage. This algorithm has strong global convergence ability, but it was easy to fall into the local optima. An optimization method based on artificial fish swarm algorithm was proposed in [7]. It effectively achieved wireless sensor network coverage optimization, but its redundancy is high. A method using bee algorithm was introduced in [8]. This algorithm provides higher coverage rate than the genetic algorithm and uses less system resources. In [9], the authors proposed a sensor deployment scheme based on glowworm swarm optimization (GSO) to enhance the network coverage after their initial random deployment of the mobile sensors. A novel sensor deployment scheme based on fruit fly algorithm (FOA) was proposed in [10] to improve the coverage rate. Compared with the classic standard PSO and GSO, this algorithm has faster convergence speed and higher coverage rate, but it needs to

be improved to achieve approximately complete coverage. The existing works show that the swarm intelligence optimization algorithm can improve the coverage rate and service performance of a WSN. But in order to meet the practical application requirements, the coverage rate and uniformity of WSN still need to be improved.

The ant lion optimizer (ALO) is a natural-inspired optimization algorithm proposed by Mirjalili in 2015. Researchers found that ALO surmounts other famous techniques like genetic algorithm (GA) and particle swarm optimization for various engineering problems [11]. At present, ALO algorithm is successfully applied to multiobjective transformer design optimization [12], power system optimization [13], WSN data gathering [14], route planning for unmanned aerial vehicle [15], etc. However, similar to other intelligent algorithms, the ALO algorithm also has problems such as slow convergence and easy to fall into local optima, and its optimization ability still needs to be improved. In this regard, scholars suggested many effective improved ALO algorithms. In [16], the authors proposed a novel quasiopositional chaotic antlion optimizer (QOCALO). This algorithm is designed by combining the quasiopposition-based learning (QOBL) and chaotic local search (CLS), which offers better results than the original ALO in terms of solution quality and convergence speed. In [17], the authors proposed an opposition-based Laplacian antlion optimizer (OB-L-ALO), which accelerates the convergence speed very efficiently and proficiently to avoid local optima. In [18], the authors proposed an improved ALO algorithm (OB-LF-ALO) integrating levy flight (LF) and opposition-based learning (OB), which enhances the exploration of the unvisited region of the search space and accelerated the convergence rate of basic ALO.

In order to maximize the coverage rate of WSN, aiming at the defects of ALO algorithm, we propose an improved ant lion algorithm (IALO) based on Cuckoo Search, Cauchy mutation, and differential evolution. The effectiveness of IALO is verified by comparing the benchmark functions. Then, IALO is applied to coverage optimization in WSN, and its application value is verified compared with other four algorithms in the same scene.

The rest of the paper is organized as follows. In Section 2, we give the coverage model. In Section 3, we introduce the standard ALO algorithm and present our IALO algorithm in detail. We give the experiment results and conclusions in Section 4 and 5, respectively.

2. Coverage Model

We assume that the monitor area A is a two-dimensional region with an area of $L \times L\,\mathrm{m}^2$. N mobile sensors are deployed randomly in this area. The coordinate of the i-th node is $\mathrm{node}_i = (x_i, y_i)$. The sensors are homogeneous; their sensing radius and communication radius are presented as r_s and r_c. Since the sensing range of each sensor node is a closed circular area with (x_i, y_i) as its center and r_s as a fixed radius, the monitor area is discretized into $m \times n$ pixels to be covered for the convenience of calculation. Each pixel is expressed as $p_j = (x_j, y_j)$, $j = 1, 2, \cdots m \times n$. The Euclidean distance between node_i and pixel $p_j = (x_j, y_j)$ is defined by the following equation:

$$d\left(\mathrm{node}_i, p_j\right) = \sqrt{\left(x_i - x_j\right)^2 - \left(y_i - y_j\right)^2}. \tag{1}$$

We use a probabilistic detection model in this paper. The sensing probability P_{cov} of the node node_i to the pixel p_j is described as follows [19]:

$$P_{\mathrm{cov}}\left(\mathrm{node}_i, p_j\right) = \begin{cases} 0 & r_s + r_e \le d\left(\mathrm{node}_i, p_j\right), \\ \exp\left(\dfrac{-\alpha_1 \lambda_1^{\beta_1}}{\lambda_2^{\beta_2} + \alpha_2}\right) & r_s - r_e < d\left(\mathrm{node}_i, p_j\right) < r_s + r_e, \\ 1 & r_s - r_e \ge d\left(\mathrm{node}_i, p_j\right), \end{cases} \tag{2}$$

where r_e $(0 < r_e < r_s)$ is a measure of uncertainty caused by environmental interference in detection. α_1, α_2, β_1, and β_2 are the detection probability parameters related to sensor node characteristics. λ_1, λ_2 are input parameters, $\lambda_1 = r_e - r_s + d(\mathrm{node}_i, p_j)$, and $\lambda_2 = r_e + r_s - d(\mathrm{node}_i, p_j)$.

In the monitor area, a pixel can be sensed by multiple sensors at the same time. Therefore, the joint sensing probability of the sensor set $\mathrm{Node}_{\mathrm{all}}$ to the pixel p_j shows if p_j is effectively covered [19].

$$C\left(\mathrm{Node}_{\mathrm{all}}, p_j\right) = 1 - \prod_{\mathrm{node}_i \in \mathrm{Node}_{\mathrm{all}}} \left(1 - P_{\mathrm{cov}}\left(\mathrm{node}_i, p_j\right)\right). \tag{3}$$

When $C(\mathrm{Node}_{\mathrm{all}}, p_j)$ is greater than or equal to a certain threshold C_{th}, it is considered that the pixel p_j can be detected by the sensor node.

In this paper, the coverage rate of a wireless sensor network is defined as the ratio of the number of the pixels that have been covered to the total number of all pixels in the monitor area. The coverage rate R_{cov} is calculated as follows [19]:

$$R_{\mathrm{cov}} = \frac{\sum_{j=1}^{m \times n} C\left(\mathrm{Node}_{\mathrm{all}}, p_j\right)}{m \times n}. \tag{4}$$

Equation (4) is used as the objective function to maximize the network coverage. We use IALO to obtain the optimal value of R_{cov} to improve the coverage performance of a WSN.

3. The Wireless Sensor Network Coverage Optimization Algorithm

3.1. Standard ALO Algorithm. ALO optimizes network coverage by mimicking the interaction between antlions and ants in a trap. The ants explore the search space by randomly

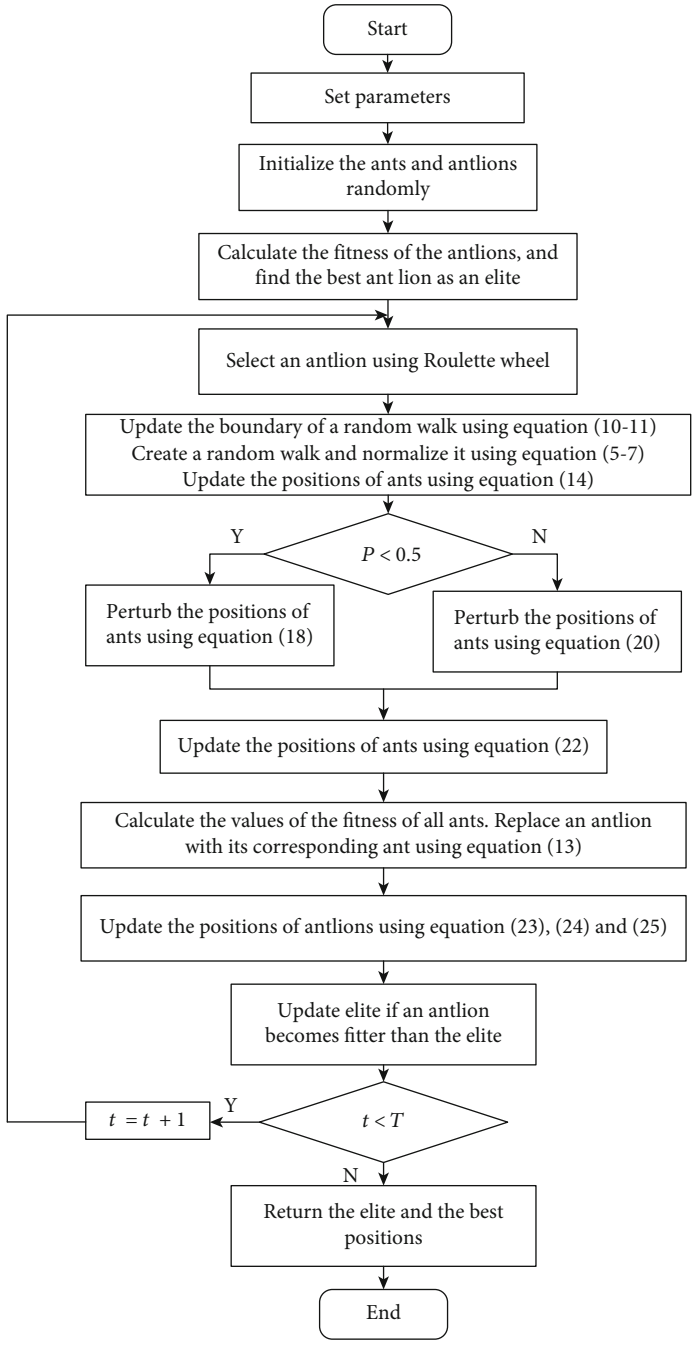


Figure 1: Coverage optimization in IALO.

walking around the antlions, and they learn from the selected antlion and the elite to ensure the diversity of the population and the optimization performance of the algorithm. An antlion is equivalent to a solution of the problem, and it updates and saves the approximate optimal solution by hunting the ants with high fitness. The five preying steps of the antlion can be represented by the following mathematical model (equations (5)–(14)) [11].

3.1.1. Random Walks of Ants. The ants walk randomly around the antlion to find food in the search space, and the antlions build traps to trap them. A random walk is chosen for modelling the ants' movement as follows:

$$X(t) = [0, \text{cumsum}(2r(t_1)) - 1, \cdots, \text{cumsum}(2r(t_n)) - 1], \tag{5}$$

$$r(t) = \begin{cases} 1, \boldsymbol{rand} > 0.5, \\ 0, \boldsymbol{rand} \leq 0.5, \end{cases} \tag{6}$$

1: Set the parameters of the WSN: the sensing radius of sensor r_s, the number of the nodes N, and the area of the monitoring area $L \times L$
2: Set the parameters of the algorithm: the population size Num, the dimension d, the maximum iteration number T, the lower bound lb, and the upper bound ub
3: Initialize the ants and antlions randomly
4: Calculate the fitness of the antlions, and find the best antlion as an elite
5: ***While***$(t < T)$
6: ***for1***$i = 1$ *to* Num
7: Select an antlion using roulette wheel
8: Update the boundary of a random walk using equations (10) and (11)
9: Create a random walk and normalize it using equation (5), (6), and (7)
10: Update the positions of ants using equations (14), (21), and (22)
11: ***End for1***
12: Calculate the values of the fitness of all ants
13: Replace an antlion with its corresponding ant using equation (13)
14: ***for2***$i = 1$ *to* Num
15: Update the positions of antlions using equations (23), (24), and (25)
16: ***End for2***
17: Update the elite if an antlion has a better fitness than it
18: $t = t + 1$
19: ***End while***
20: Output the elite (coverage rate) and the best positions (the deployment of the nodes).

Pseudocode 1: Improved ant lion optimizer.

where $X(t)$ is the set of the steps of the random walk, cumsum is the calculated cumulative sum, n is the maximum number of iterations, t shows the step of random walk, $r(t)$ is a stochastic function, and rand is a random number with uniform distribution in the interval of $[0, 1]$. The position of an ant is updated at every step of the optimization. In order to keep the random walks inside the search space, we use normalization by

$$X_i^t = \frac{\left(X_i^t - a_i\right)\left(d_i^t - c_i^t\right)}{b_i - a_i} + c_i^t, \tag{7}$$

where a_i and b_i are the minimum and maximum of the random walk of the i-th variable and c_i^t and d_i^t are the lower and upper bound of the i-th variable at the t-th iteration.

3.1.2. Trapping in Antlion's Pits. The ants select an antlion for random walk at every iteration, and their random walks are affected by antlions' traps. This mathematical model can be expressed as follows:

$$c_i^t = \text{Antlion}_j^t + c^t, \tag{8}$$

$$d_i^t = \text{Antlion}_j^t + d^t, \tag{9}$$

where Antlion_j^t is the position of the selected j-th antlion at the t-th iteration and c^t and d^t are the minimum and maximum values of all variables for the i-th ant at the t-th iteration.

3.1.3. Sliding the Ants towards an Antlion. The ALO algorithm uses a roulette wheel selection operator to select antlions based on their fitness during optimization. An antlion with better fitness has greater chances to be selected and to catch ants. Once an antlion realizes that an ant is in the trap, it shoots sands outward the center of the trap to prevent the ant from escaping. As a result, the trap decreases adaptively, and the range of the ant's random walk becomes smaller. Finally, the ant slowly slides toward the antlion. As the number of iterations increases, the change of the trap range can be expressed as follows:

$$c^t = \frac{c^t}{I}, \tag{10}$$

$$d^t = \frac{d^t}{I}, \tag{11}$$

$$I = 10^{\omega}\frac{t}{T}\begin{cases} \omega = 2, \text{if } t > 0.10T, \\ \omega = 3, \text{if } t > 0.50T, \\ \omega = 4, \text{if } t > 0.75T, \\ \omega = 5, \text{if } t > 0.90T, \\ \omega = 6, \text{if } t > 0.95T. \end{cases} \tag{12}$$

In these equations, t shows the current iteration, T is the maximum value of the iterations, I is a ratio, $I = 1$ when $t \leq 0.1\,T$, and ω is a constant based on the current iteration. The range of the traps shrinks sharply in the late iterations.

3.1.4. Catching Prey and Rebuilding the Trap. The fitness of the new position of an ant is calculated in this stage. If an ant has a better fitness than its corresponding antlion, the ant is caught by the antlion and then the antlion reconstructs

Table 1: Benchmark test functions.

Function	Formula	Dim	Bounds	Optimum
Sphere	$F_1(x) = \sum_{i=1}^{d} x_i^2$	30	[−100,100]	0
Schwefel 2.22	$F_2(x) = \sum_{i=1}^{d} \lvert x_i \rvert + \prod_{i=1}^{d} \lvert x_i \rvert$	30	[−10, 10]	0
Schwefel 1.2	$F_3(x) = \sum_{i=1}^{d} \left(\sum_{j=1}^{i} x_j \right)^2$	30	[−100,100]	0
Rosenbrock	$F_4(x) = \sum_{i=1}^{d-1} \left[100 \left(x_{i+1} - x_i^2 \right)^2 + (x_i - 1)^2 \right]$	30	[−30, 30]	0
Quartic	$F_5(x) = \sum_{i=1}^{d} i x_i^4 + \text{random}[0, 1)$	30	[−1.28,1.28]	0
Rastrigin	$F_6(x) = \sum_{i=1}^{d} \left[x_i^2 - 10 \cos(2\pi x_i) + 10 \right]$	30	[−5.12,5.12]	0
Ackley	$F_7(x) = -20 \exp\left(-0.2 \sqrt{1/d \sum_{i=1}^{d} x_i^2} \right) - \exp\left(1/d \sum_{i=1}^{d} \cos(2\pi x_i) \right) + 20 + e$	30	[−32, 32]	0
Griewank	$F_8(x) = 1/4000 \sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d} \cos\left(x_i / \sqrt{i} \right) + 1$	30	[−600,600]	0
Penalized	$F_9(x) = (\pi/d) \left\{ 10 \sin(\pi y_1) + \sum_{i=1}^{d-1} (y_i - 1)^2 \left[1 + 10 \sin^2(\pi y_{i+1}) \right] + (y_d - 1)^2 \right\} + \sum_{i=1}^{d} u(x_i, 10, 100, 4), y_i = 1 + (x_i + 1/4), u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i > a \end{cases}$	30	[−50, 50]	0

Table 2: Parameter setting.

Algorithm	Parameter setting
FOA	$s = 0.3$
PSO	$c_1 = c_2 = 2$, $\omega_{\max} = 0.9$, $\omega_{\min} = 0.4$
WOA [23]	$a_1 = [2, 0]$, $a_2 = [-2, 1]$, $b = 1$
GWO [24]	$a = [2, 0]$
IALO	CR = 0.95

the trap for the next hunt. The antlion is required to update its position to the latest position of the hunted ant to enhance its chance of catching new prey. The following equation is presented in this regard:

$$\text{Antlion}_j^t = \text{Ant}_i^t \text{ if } f\left(\text{Ant}_i^t\right) > f\left(\text{Antlion}_j^t\right). \quad (13)$$

Antlion_j^t is the position of the j-th antlion selected at the t-th iteration, Ant_i^t is the position of the i-th ant at the t-th iteration, and f shows the objective function.

3.1.5. Elitism. The best antlion is saved and considered as an elite antlion in each iteration, which affects the movements of all the ants during the iterations. Since every ant randomly walks around an antlion selected by the roulette wheel and the elite simultaneously, the position of the i-th ant at the t-th iteration can be expressed as follows:

$$\text{Ant}_i^t = \frac{R_A^t + R_E^t}{2}. \quad (14)$$

R_A^t is the random walk around the antlion selected at the t-th iteration, and R_E^t is the random walk around the elite at the t-th iteration.

3.2. Improved ALO Algorithm. We propose an improved ant lion optimizer integrating with Cuckoo Search, Cauchy mutation, and differential evolution (IALO). Compared to ALO, IALO has faster convergence speed and higher optimization accuracy.

3.2.1. Cuckoo Search. Cuckoo Search [20] is a nature-inspired optimization algorithm proposed by Xin-She Yang and Suash Deb. This algorithm mimics the aggressive reproduction strategy of cuckoo bird species with the Levy flight action of birds and fruit flies. The cuckoo birds lay their eggs in the nests of other birds so that they would not hatch their own eggs. In CS, three idealized rules are used:

(1) Each cuckoo lays one egg at a time and randomly chooses a nest to dump it

(2) The best nests with high quality eggs will be preserved for the next generation

(3) The number of available parasitic host nests is fixed, and a host bird detects a parasitic cuckoo egg with a probability $p_a \in [0, 1]$

According to the above rules, the update of nests is generated by the random walk of Levy flight as follows [20]:

$$x_i^{t+1} = x_i^t + \alpha \oplus \text{Levy}(\beta), \quad (15)$$

Table 3: Comparison results of function optimization.

Function	Metrics	IALO	ALO	FOA	WOA	GWO	PSO	OB-L-ALO [17]	OB-LF-ALO [18]
F1	Mean	**0.0000E + 00**	1.0654E − 05	1.0146E − 03	4.8198E − 151	1.0907E − 58	3.8243E − 01	1.0100E − 09	**0.0000E + 00**
	Std	**0.0000E + 00**	7.6999E − 06	1.3999E − 04	2.6394E − 150	3.0615E − 58	1.0367E − 01	4.6300E − 09	**0.0000E + 00**
F2	Mean	**0.0000E + 00**	3.6011E + 01	1.6207E + 00	1.6570E − 104	1.0141E − 34	3.4169E + 00	5.9800E − 05	**0.0000E + 00**
	Std	**0.0000E + 00**	4.6837E + 01	1.0403E − 01	5.8354E − 104	1.0483E − 34	1.4875E + 00	4.2600E − 05	**0.0000E + 00**
F3	Mean	**0.0000E + 00**	1.0400E + 03	2.7936E − 01	2.0499E + 04	2.2381E − 14	2.7138E + 01	1.1600E − 29	**0.0000E + 00**
	Std	**0.0000E + 00**	5.0487E + 02	4.5952E − 02	8.5427E + 03	1.0848E − 13	7.0735E + 00	5.9500E − 29	**0.0000E + 00**
F4	Mean	**1.7788E − 04**	1.2881E + 02	2.8844E + 01	2.7102E + 01	2.6849E + 01	2.7229E + 02	5.4600E − 03	9.5900E − 03
	Std	**2.8763E − 04**	2.2785E + 02	2.1101E − 01	5.2570E − 01	7.8341E − 01	2.6406E + 02	7.1700E − 03	1.2800E − 02
F5	Mean	**5.7664E − 05**	9.8351E − 02	2.4607E + 01	2.5284E − 03	7.2906E − 04	1.3874E + 00	2.7800E − 04	9.7300E − 05
	Std	**5.7176E − 05**	2.6388E − 02	7.1563E + 00	2.2613E − 03	3.1465E − 04	3.0946E + 00	2.5300E − 04	8.6900E − 05
F6	Mean	**0.0000E + 00**	8.6163E + 01	7.1921E + 01	**0.0000E + 00**	5.2374E − 01	8.9437E + 01	3.0100E − 09	**0.0000E + 00**
	Std	**0.0000E + 00**	3.1014E + 01	7.4816E + 00	**0.0000E + 00**	1.7055E + 00	1.7614E + 01	4.8200E − 09	**0.0000E + 00**
F7	Mean	**8.8818E − 16**	2.0733E + 00	8.8257E − 02	3.7303E − 15	1.7349E − 14	2.0203E + 00	2.1000E − 05	**8.8818E − 16**
	Std	**0.0000E + 00**	6.0300E − 01	7.9936E − 03	2.3603E − 15	3.0208E − 15	5.8398E − 01	1.4600E − 05	4.0100E − 31
F8	Mean	**0.0000E + 00**	1.0299E − 02	1.5752E − 06	**0.0000E + 00**	2.0188E − 03	6.6705E − 02	6.3200E − 09	**0.0000E + 00**
	Std	**0.0000E + 00**	6.4416E − 03	3.5681E − 07	**0.0000E + 00**	7.1923E − 03	2.4607E − 02	1.2700E − 08	**0.0000E + 00**
F9	Mean	**6.0988E − 09**	9.1844E + 00	1.7176E + 00	7.1550E − 03	3.4505E − 02	3.5577E + 00	6.4400E − 07	3.2000E − 05
	Std	**1.1969E − 08**	4.3550E + 00	4.0992E − 03	8.2659E − 03	1.5983E − 02	1.6070E + 00	5.8100E − 07	8.4100E − 05

where x_i^{t+1} shows the new solution and x_i^t shows the solution of the i-th cuckoo. The product $\oplus$ means entry-wise multiplication. α is the step size, which is usually selected as 1; it also can be set by

$$\alpha = \alpha_0\left(x_i^t - x_{\text{best}}^t\right). \tag{16}$$

In equation (16), α_0 shows the initial search step size, and x_{best}^t is the global optimal solution at the t-th iteration. In the random walk provided by Levy flight, the random step size is drawn from a Levy distribution.

$$L(\beta) \sim u = t^{-1-\beta}, \quad 0 < \beta \leqslant 2. \tag{17}$$

In this work, CS is embedded into IALO to perturb the positions of ants, in order to increase the chance of IALO to jump out of the local optima and enhance its ability to converge towards global optima. Using CS to perturb the positions of ants can be expressed as follows:

$$\text{Ant}_i^{t*} = \text{Ant}_i^t + \alpha_0\left(\text{Ant}_i^t - \text{elite}\right) \oplus \text{Levy}(\beta), \tag{18}$$

where Ant_i^{t*} shows the new position of an ant after perturbation and elite shows the best antlion of the t-th iteration.

3.2.2. Cauchy Mutation. Cauchy mutation is used in IALO to improve the diversity of the population of ants, increase the search space, accelerate the convergence speed, and improve the global search ability. The origin-centered 1D Cauchy density function is defined as follows [21]:

$$f(x) = \frac{1}{\pi}\left(\frac{t}{x^2 + t^2}\right). \tag{19}$$

After Cauchy mutation, the ants use less time to search the adjacent intervals and spend more time to search for the global optima. Therefore, Cauchy mutation is beneficial to increase the population diversity, and it also improves the global search ability of IALO. In this work, the Cauchy mutation carried out by an ant should be calculated as

$$\text{Ant}_i^{t*} = \text{Ant}_i^t + \text{Ant}_i^t \oplus \text{Cauchy}(0, 1), \tag{20}$$

where $\text{Cauchy}(0, 1)$ is a random number produced by the Cauchy distribution with $t = 1$.

In order to further improve the optimization performance of IALO, we adopt a dynamic selection strategy to mutate the position of an ant. Cuckoo Search and Cauchy mutation are used alternately under a certain probability of 50%.

$$\begin{cases} \text{Ant}_i^{t*} = \text{Ant}_i^t + \alpha_0\left(\text{Ant}_i^t - \text{elite}\right) \oplus \text{Levy}(\beta), & p < 0.5, \\ \text{Ant}_i^{t*} = \text{Ant}_i^t + \text{Ant}_i^t \oplus \text{Cauchy}(0, 1), & p \geq 0.5, \end{cases} \tag{21}$$

where p is a random number in $[0, 1]$.

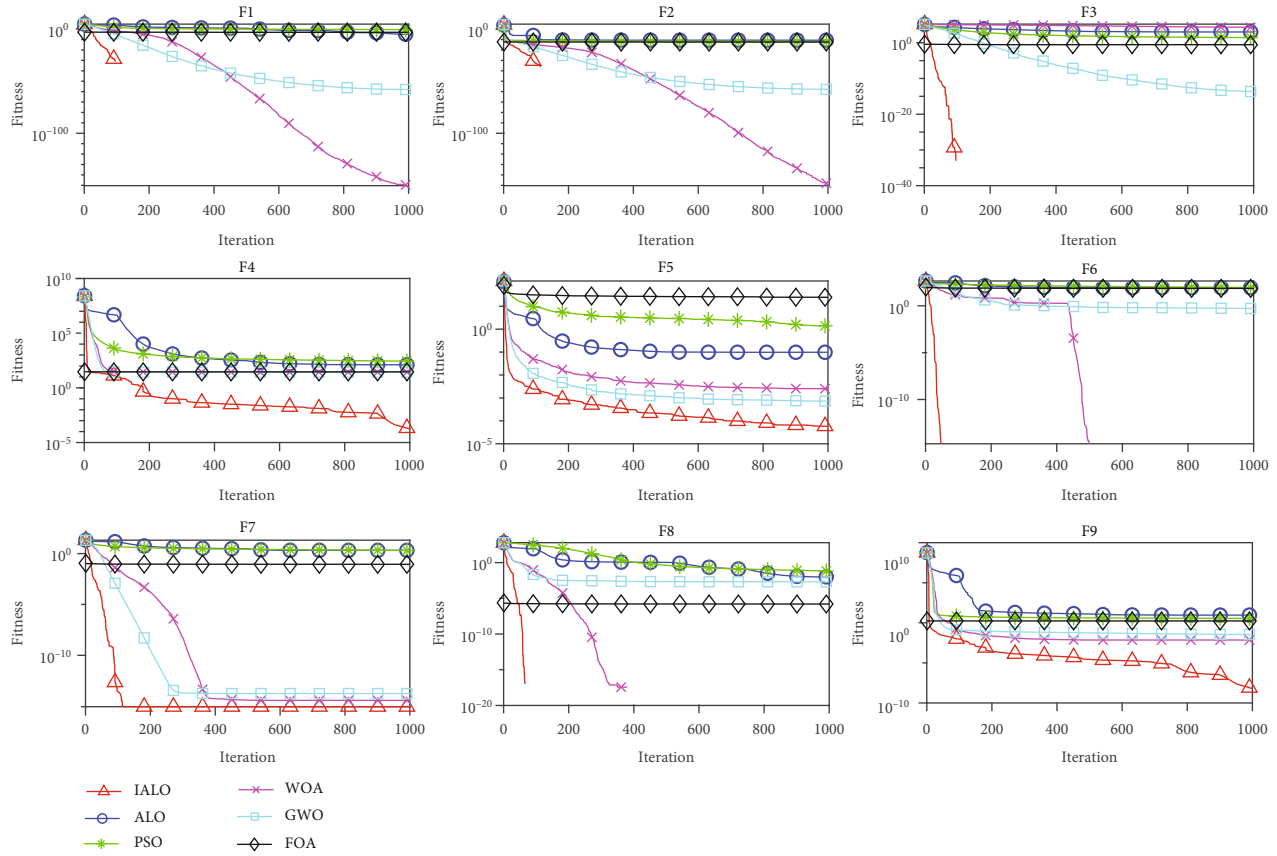


Figure 2: The convergence curves of six algorithms.

Table 4: Experiment setting.

Experiment	Area	Number of sensors (N)	Sensing radius (r_s)
1	50 m × 50 m	30,40,50,60,70	5 m
2	100 m × 100 m	50,60,70,80,90	7 m

After mutation, a greedy criterion is introduced to compare the fitness of the new position of an ant with its old one to determine whether to update the ant's position.

$$\begin{cases} \text{Ant}_i^{t+1} = \text{Ant}_i^{t*} & f\left(\text{Ant}_i^{t*}\right) > f\left(\text{Ant}_i^{t}\right), \\ \text{Ant}_i^{t+1} = \text{Ant}_i^{t} & f\left(\text{Ant}_i^{t*}\right) \le f\left(\text{Ant}_i^{t}\right). \end{cases} \tag{22}$$

3.2.3. Differential Evolution Algorithm. Differential evolution (DE) includes three operations: mutation, crossover, and selection [22]. DE first generates a new mutant vector by mutation strategy and then mixes the new vector with the target vector to generate a trial vector. The trial vector is compared to the target vector by a greedy criterion. If the trial vector outputs a smaller cost function value than the target vector, the target vector is replaced by the trial vector in the following iteration. Otherwise, the target vector is retained. The common differential mutation strategies are DE/rand/1, DE/rand/2, DE/best/1, DE/best/2, etc. In IALO, we use DE/rand/1 mutation strategy. We assume that there is a target vector $X_i(t)$, and we generate the mutant vector $V_i(t)$ according to

$$V_i(t) = X_{r1}(t) + F(X_{r2}(t) - X_{r3}(t)). \tag{23}$$

In equation (23), F is the mutation factor; it is a constant from $[0, 1]$. $r1$, $r2$, and $r3$ are random integers that are different with the running index i. $X_{r1}(t)$, $X_{r2}(t)$, and $X_{r3}(t)$ are distinct vectors randomly chosen from the current population at the t-th iteration. The mutant vector $V_i(t)$ is mixed with $X_i(t)$; they generate the trial vector $U_i(t)$ using

$$U_{i,j}(t) = \begin{cases} V_{i,j}(t), & \text{if } \text{rand}_{i,j}(0,1) \le \text{CR or } j = j_{\text{rand}}, \\ X_{i,j}(t), & \text{otherwise.} \end{cases} \tag{24}$$

In equation (24), j_{rand} is a randomly chosen index in the interval $[1, D]$, D shows the dimension of the vector, and $\text{rand}_{i,j}[0, 1]$ is a uniform random number in $[0, 1]$. CR is the crossover factor in $[0, 1]$.

The selection of the differential evolution algorithm determines whether the target vector or the trial vector is

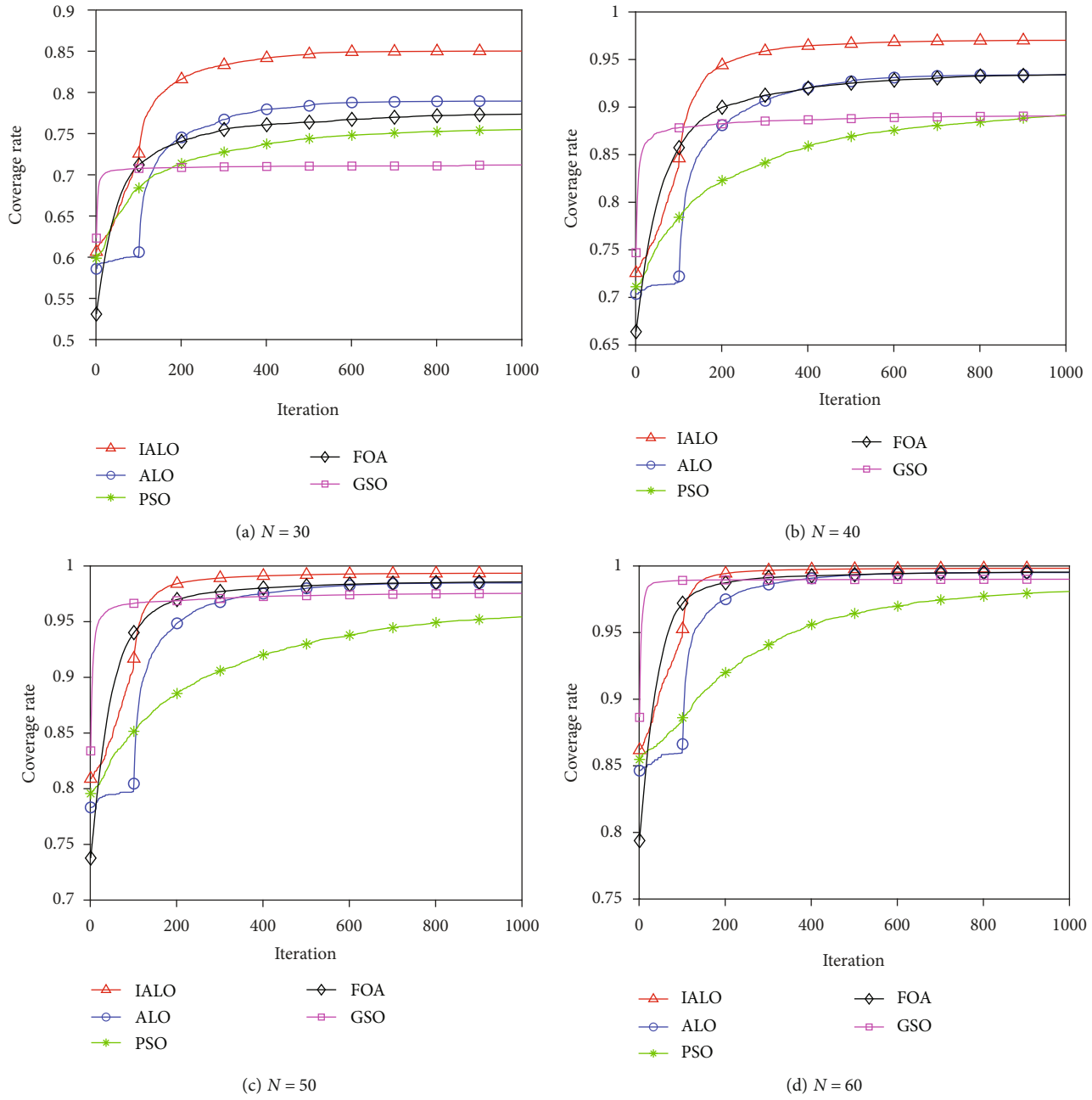


Figure 3: The coverage rate curves of the five algorithms in experiment 1.

retained in the next iteration. For the maximization problem, the selection is defined as follows:

$$X_i(t+1) = \begin{cases} U_i(t) & \text{if } f(U_i(t)) > f(X_i(t)), \\ X_i(t) & \text{if } f(U_i(t)) \le f(X_i(t)). \end{cases} \quad (25)$$

In this work, DE is introduced into IALO to perform differential mutation on the population of antlions to improve the convergence accuracy.

3.3. Coverage Optimization Based on IALO. In this paper, IALO is used to optimize the deployment of sensors. Its optimization objective is to maximize the coverage rate R_{cov} in the coverage model of a WSN. The coverage optimization problem is transformed into a high-dimensional vector optimization problem with equation (4) as its objective function. The process of node position optimization is transformed into a series of behaviors of antlions preying on ants. After the iterations, the position of the elite antlion is the optimal solution of the node distribution.

Each antlion in our algorithm represents a coverage distribution scheme of the sensors. The dimension of the antlion is twice the number of the sensors, where the $2i$ and $2i-1$ dimensions represent the abscissa and ordinate of the node, respectively. The flow of the coverage

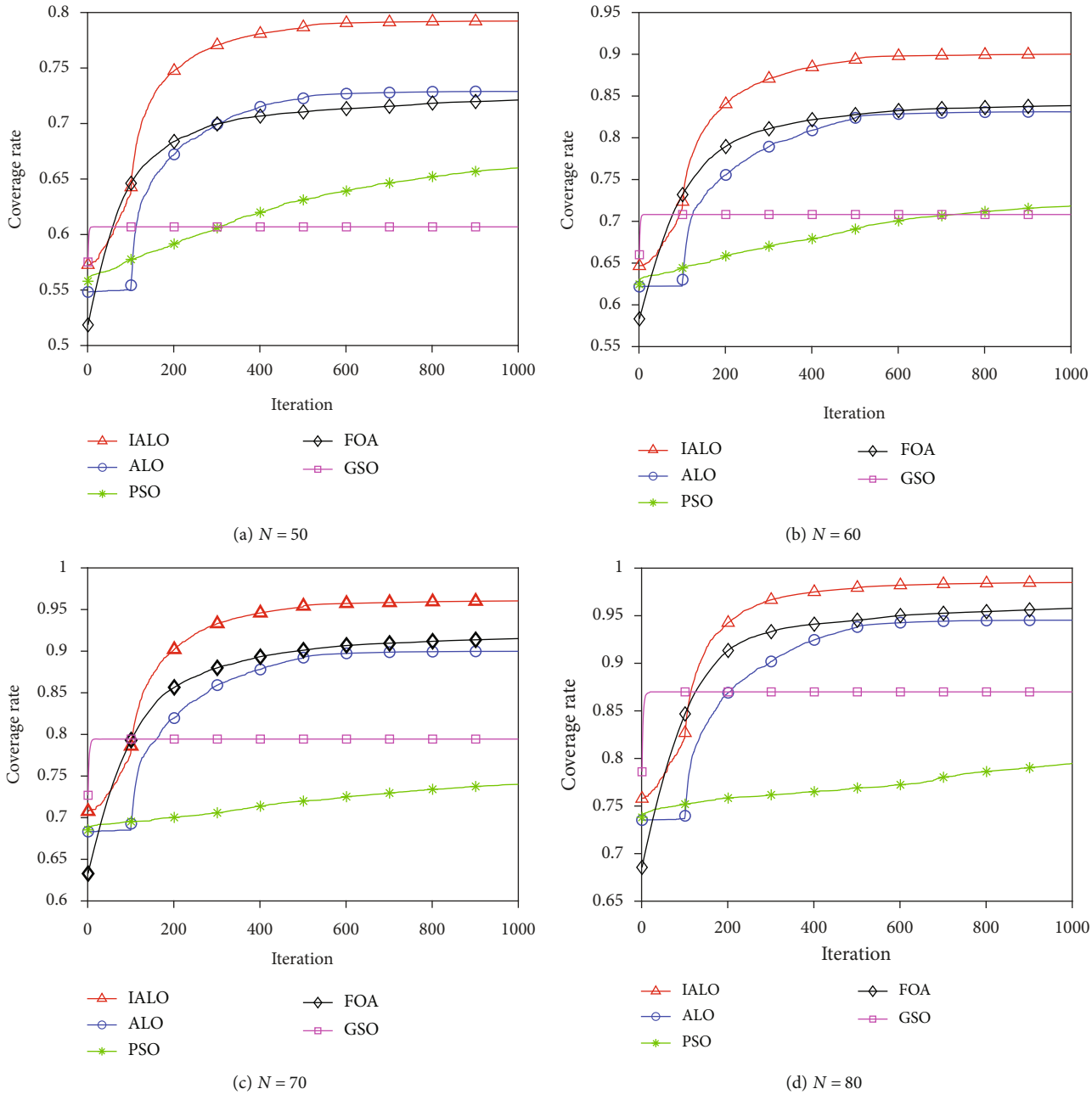


Figure 4: The coverage rate curves of the five algorithms in experiment 2.

optimization algorithm is shown in Figure 1, and its pseudocode is as follows.

4. Simulations

4.1. Function Optimization

4.1.1. Experiment Setting. In order to present the advantage of IALO, we select FOA, WOA [23], GWO [24], PSO, ALO, OB-L-ALO [17], and OB-LF-ALO [18] for algorithm experiment comparison on 9 benchmark test functions. The mathematical formula, dimension, boundary range, and global optimum of the test functions are shown in Table 1. Among the test functions, F1-F5 are unimodal functions, and F6-F9 are multimodal functions. The unimodal functions have only one global optimum which is used to evaluate the exploitation ability and convergence speed of an algorithm. Because the multimodal functions contain many local optima, they can be used to evaluate the ability to exploration and jump out of the local optima.

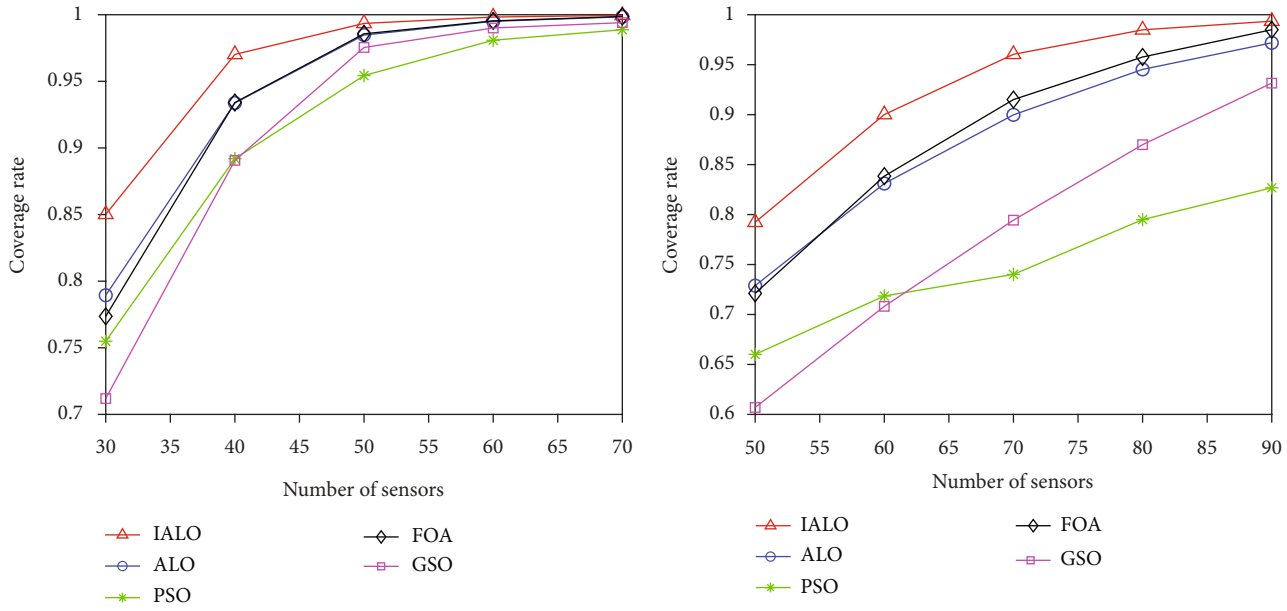
The experimental environment is Intel(R) Core (TM) i7-9750H dual-core CPU, 16 GB memory, Win10 64-bit operating system, and the simulation software is MATLAB R2019b. For the fairness of the experimental results, the population size of each algorithm is set to 30, the maximum number of iterations is set to 1000, and each algorithm is

Table 5: The coverage rates of the five algorithms in experiment 1.

N	Metrics	IALO	ALO	PSO	FOA	GSO
$N = 30$	Mean	**0.85018**	0.78943	0.75497	0.77363	0.71192
	Std	**5.763E − 03**	8.878E − 03	1.829E − 02	1.855E − 02	9.339E − 03
	Best	**0.85999**	0.80723	0.79885	0.80948	0.73153
	Worst	**0.83836**	0.77146	0.72086	0.74009	0.69399
$N = 40$	Mean	**0.97020**	0.93377	0.89192	0.93411	0.89067
	Std	**3.171E − 03**	1.153E − 02	2.077E − 02	1.395E − 02	8.572E − 03
	Best	**0.97378**	0.95225	0.92028	0.95417	0.90321
	Worst	**0.96157**	0.91060	0.85125	0.89470	0.87260
$N = 50$	Mean	**0.99348**	0.98475	0.95435	0.98569	0.97542
	Std	**3.792E − 04**	2.195E − 03	1.163E − 02	5.305E − 03	3.419E − 03
	Best	**0.99424**	0.98836	0.97348	0.99006	0.98244
	Worst	**0.99292**	0.97993	0.92907	0.97034	0.96966
$N = 60$	Mean	**0.99827**	0.99506	0.98086	0.99537	0.99004
	Std	**1.151E − 04**	8.077E − 04	6.670E − 03	3.270E − 03	2.187E − 04
	Best	**0.99842**	0.99626	0.98994	0.99682	0.99061
	Worst	**0.99797**	0.99338	0.96682	0.98168	0.98970
$N = 70$	Mean	**0.99954**	0.99842	0.98875	0.99869	0.99405
	Std	**3.237E − 05**	3.001E − 04	1.093E − 02	1.756E − 04	2.329E − 04
	Best	**0.99958**	0.99887	0.99612	0.99895	0.99449
	Worst	**0.99948**	0.99769	0.95062	0.99826	0.99367

Table 6: The coverage rates of the five algorithms in experiment 2.

N	Metrics	IALO	ALO	PSO	FOA	GSO
$N = 50$	Mean	**0.79227**	0.72887	0.66010	0.72116	0.60691
	Std	7.415E − 03	9.514E − 03	4.070E − 02	1.305E − 02	**6.160E − 03**
	Best	**0.80641**	0.74323	0.70827	0.73651	0.62004
	Worst	**0.78046**	0.70679	0.56853	0.68272	0.59624
$N = 60$	Mean	**0.90010**	0.83108	0.71850	0.83845	0.70815
	Std	**4.893E − 03**	1.000E − 02	5.866E − 02	1.201E − 02	6.198E − 03
	Best	**0.90962**	0.84976	0.79331	0.85503	0.71806
	Worst	**0.89157**	0.81256	0.61528	0.80525	0.69819
$N = 70$	Mean	**0.96036**	0.89981	0.74013	0.91508	0.79441
	Std	**3.989E − 03**	9.879E − 03	5.724E − 02	1.799E − 02	5.576E − 03
	Best	**0.96672**	0.91706	0.81953	0.93925	0.80562
	Worst	**0.95329**	0.87995	0.67545	0.86460	0.78722
$N = 80$	Mean	**0.98486**	0.94529	0.79494	0.95769	0.86995
	Std	**1.974E − 03**	7.423E − 03	4.869E − 02	8.188E − 03	4.215E − 03
	Best	**0.98709**	0.95682	0.87494	0.96939	0.88118
	Worst	**0.98109**	0.92239	0.73325	0.93539	0.86284
$N = 90$	Mean	**0.99356**	0.97189	0.82679	0.98498	0.93154
	Std	**6.255E − 04**	5.727E − 03	5.189E − 02	4.129E − 03	4.299E − 03
	Best	**0.99455**	0.98038	0.91795	0.98895	0.93739
	Worst	**0.99214**	0.95947	0.76388	0.97370	0.92491



(a) Experiment 1 (b) Experiment 2

Figure 5: Comparison of coverage rate under different numbers of sensors.

independently run 30 times for comprehensive testing. The parameter setting of each algorithm is shown in Table 2.

4.1.2. Comparison of Optimal Results. The comparison results of function optimization are shown in Table 3. Among them, ALO, FOA, WOA, GWO, PSO, and IALO are the results of 30 experiments of the test function, while OB-L-ALO and OB-LF-ALO are obtained by referring to other literature data.

The data in Table 3 demonstrate that IALO ranked first or tied first on average when solving F1-F5. In F1-F3, both IALO and OB-LF-ALO achieve global optima, and the standard deviation is 0. In F4-F5, IALO does not converge to the global optimum, but it has the best mean and standard deviation value among all the test algorithms. It is evident from the results that IALO has efficient exploitation ability. From the results of F6-F9, we can see that IALO is still competitive in multimodal function. In F6-F8, IALO can converge to the global optimum, which is the best or the best in parallel compared with other algorithms. In F9, the convergence accuracy of IALO is much higher than the other algorithms. It can be observed that IALO has also a very good exploration ability.

4.1.3. Comparison of Convergence Curves. The benchmark function curve can intuitively reflect the convergence speed and accuracy of each algorithm and also clearly show the ability of the algorithm to jump out of the local optima. The convergence curves of the nine benchmark test functions are shown in Figure 2. We can see that IALO can achieve or get close to the global optima in the shortest time, which shows that IALO has faster convergence speed and higher convergence accuracy. For F1-F3, F6-F8, the convergence curve of IALO decreases obviously from the beginning of iteration and the convergence speed is fast. For example, F1 and F3 achieve the global optima in the 96th and 47th generations, respectively. For F4-F5, F9, IALO does not converge to the global optima, but it can quickly jump out of the local optima, and its convergence accuracy is closer to the global optima than other algorithms. Therefore, IALO has good optimization performance and is effective and reliable in function optimization.

4.2. Coverage Optimization in WSNs. In order to verify the performance of IALO in the coverage optimization of a WSN, we assume that different numbers of sensors are randomly placed in the monitoring areas of different sizes. We compare IALO with GSO [9], FOA [10], ALO, and PSO. To eliminate the error caused by randomness, each algorithm was run 20 times in each experiment, and the final average results were compared.

4.2.1. Experiment Setting. As shown in Table 4, the monitoring areas of experiment 1 and experiment 2 are 50 m × 50 m and 100 m × 100 m, respectively. All sensors are homogeneous; they are with the same sensing radius in each experiment. The parameter settings of FOA, PSO, and IALO are the same as the ones in Section 4.1. The luciferin enhancement constant $\gamma = 0.1$, the luciferin decay constant $\rho = 0.9$, and the step size $s = (\sqrt{3}r_s - d_{ij})/2$ in GSO. All algorithms uniformly set the population to 30 and the maximum of iteration is 1000, and $C_{\text{th}} = 0.75$, $\alpha_1 = 1$, $\alpha_2 = 0$, $\beta_1 = 1$, $\beta_2 = 2$, $r_e = r_s/2$.

4.2.2. Comparison of Coverage Rate. Coverage rate is one of the important elements to indicate the performance of the distribution of the sensors. Figures 3 and 4 show the coverage rate of ALO, GSO, FOA, PSO, and IALO with the

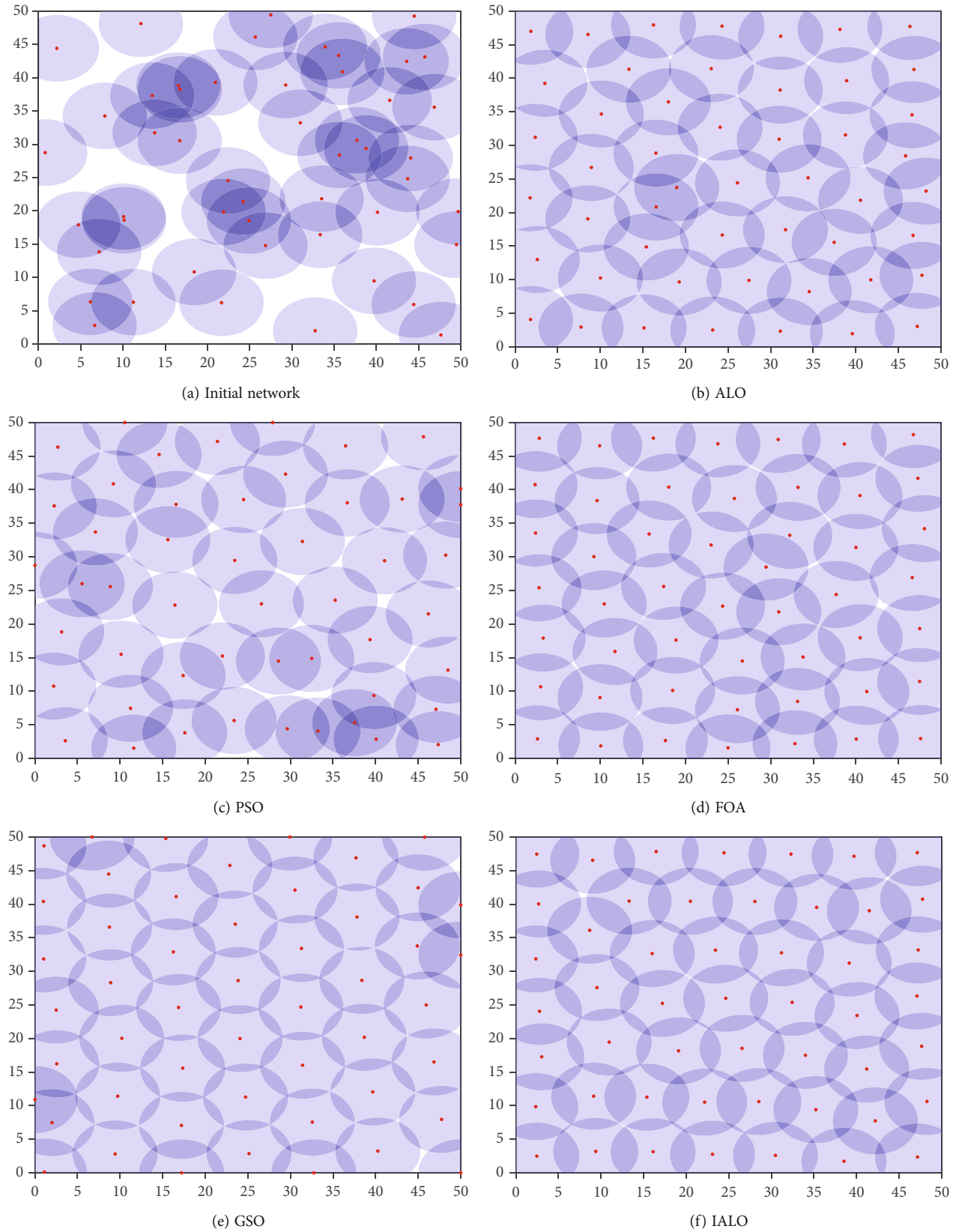


Figure 6: Sensor distribution in experiment 1.

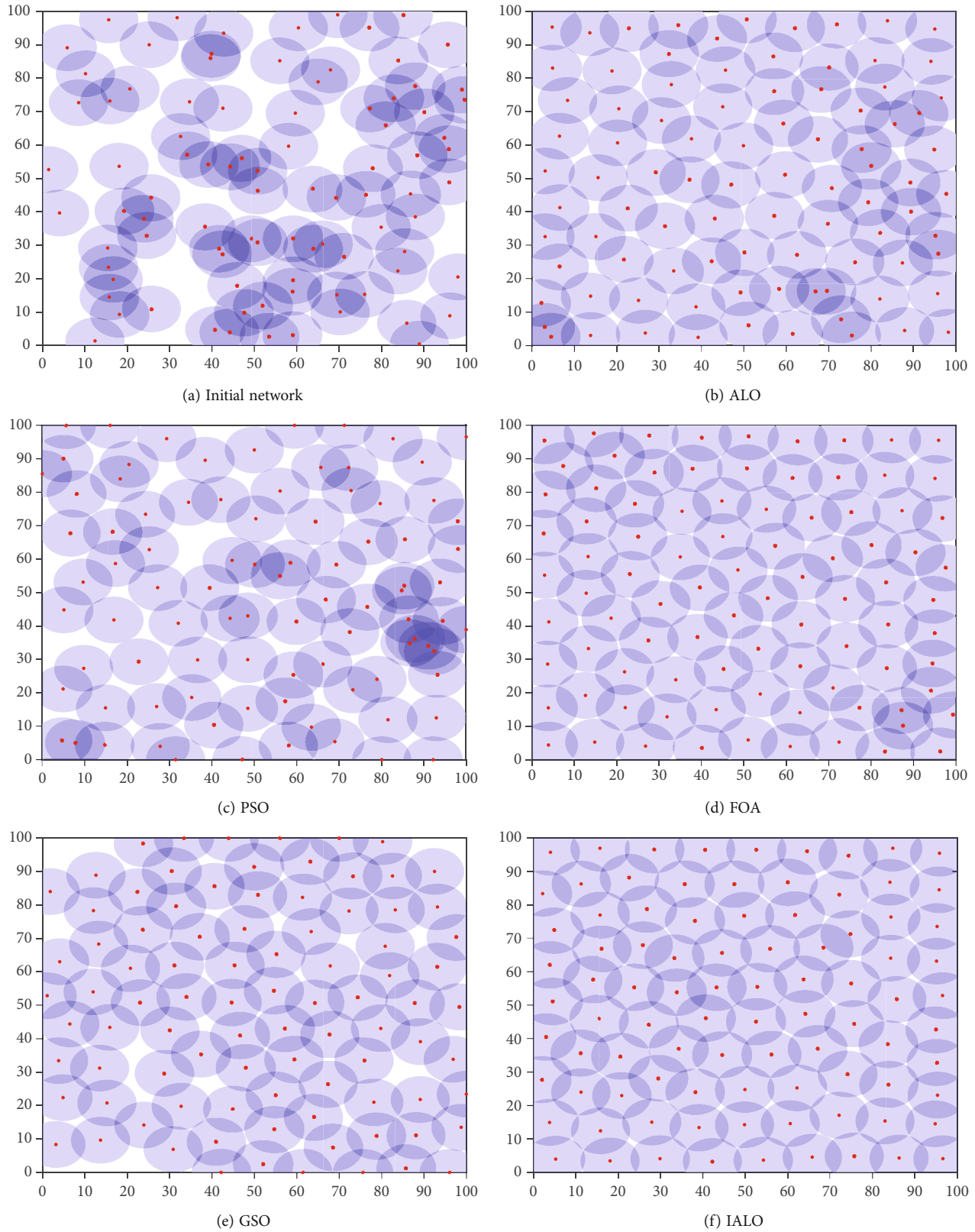


Figure 7: Sensor distribution in experiment 2.

increase of the number of iterations. Figure 3 shows the coverage rate of the five algorithms under different number of sensors in experiment 1. We can see from Figure 3 that when the number of iterations grows, the coverage rates of the five algorithms first rise sharply ad then tend to stable gradually. For example, as shown in Figure 3(b), the coverage rate of the five algorithms increases sharply in the first 1-100 iterations. In the 100-300 iterations, the rising speed of the coverage rates of the five algorithms slows down. When the iteration reaches 400 times, GSO reaches its maximum value and does no change, while the coverage rates of the other algorithms are still slowly increasing. The final coverage rate of IALO is 97.02%, ALO is 93.38%, PSO is 89.13%, FOA is 93.41%, and GSO is 89.07%. Compared with ALO, PSO, FOA, and GSO, the coverage rate of IALO increased by 3.64%, 7.83%, 3.61%, and 7.95%, respectively. Because CS, Cauchy mutation, and DE are used to enhance its global optimization ability, the coverage rate of IALO is better than the original ALO in each iteration.

In addition, the coverage rates of the five algorithms under different number of sensors in experiment 2 are shown in Figure 4. We can see that IALO has much higher coverage rate and faster convergence speed than ALO, PSO, FOA, and GSO. IALO is better than the other five algorithms in solving the coverage optimization problem.

Tables 5 and 6 show the final coverage rates of the five algorithms in experiment 1 and experiment 2, where Mean, Std, Best, and Worst represent the average value, standard deviation, the best value, and the worst value of the coverage rates, respectively. We can see from Table 5 that the coverage rate of each algorithm increases with the increase of the number of sensors. The four metrics (Mean, Std, Best, and Worst) of IALO are much better than those of the rest of the algorithms, and when the number of sensors is small, the coverage advantage of IALO is more obvious. For example, in experiment 1, when $N = 70$, compared with ALO, PSO, FOA, and GSO, IALO has improved the coverage rate by 0.11%, 1.08%, 0.08%, and 0.55%, respectively. However, when $N = 30$, the coverage rate of IALO was higher than them by 6.07%, 9.52%, 7.66%, and 13.83%, respectively. From Table 6, we can see that in experiment 2, with the increase of the number of sensors, the coverage rate of IALO, ALO, and FOA increases greatly. However, IALO still has the highest coverage rate and it is more stable as it has the smallest standard deviation.

Figure 5 shows the coverage rates of the five algorithms under different number of sensors. We can see that when the number of sensors is the same, the coverage rate of IALO is higher than the other algorithms. In our experiment 1, IALO obtains a coverage rate 99.561% when $N = 53$ in the $50\,\text{m} \times 50\,\text{m}$ monitoring area. This result is higher than the ones of ALO and FOA at $N = 60$ (99.506%, 99.537%), and it is also higher than the results of PSO and GSO at $N = 70$ (98.875%, 99.405%). IALO can provide the same coverage rate with fewer sensors than the other four algorithms.

4.2.3. Comparison of Coverage Effects. Figures 6 and 7 show the sensor distribution of $N = 50$ in experiment 1 and $N = 90$ in experiment 2, respectively. In these two figures, "•" represents a sensor, and the corresponding circle is the coverage area of the sensor.

Figure 6(a) shows the initial deployment of the sensors. There are many blank areas and overlap areas because of the random deployment of sensors. Figures 6(b)–6(e) are the final sensor distribution after executing ALO, PSO, FOA, and GSO, respectively. Figure 6(f) is the final sensor deployment of IALO. It can be seen that the sensor distribution of IALO is the most uniform one with the smallest sensor redundancy. It can also be seen from Figure 7 that the IALO has the best coverage rate, the smallest redundant area, and the smallest blank coverage.

5. Conclusions

In order to improve the network coverage in a WSN, we designed the IALO algorithm according to the standard ALO algorithm. In IALO, we used Cuckoo Search and Cauchy mutation to enhance the population diversity of ants, improve its global optimization ability, and accelerate its convergence speed. We mutated the population of antlions by differential evolution to improve the convergence accuracy of IALO. Experiments on 9 benchmark functions show that IALO has efficient exploitation and exploration ability. Compared with other algorithms, IALO has faster convergence speed and higher convergence accuracy, and it can effectively jump out of the local optima. We applied IALO to the coverage optimization problem in a WSN and compared it with the standard ALO algorithm and other related optimizations. Simulations results show that under the same test environment, IALO achieves higher network coverage. IALO also provides more uniform sensor distribution with effectively reduced number of nodes.

Data Availability

The data used to support the findings of this study are available from Wei Chen (weichen@szzy.edu.cn) upon request.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This research was funded by the Natural Science Research Project of Anhui Provincial Department of Education grant number KJ2019A1058, the Domestic Visiting Study and Training Project for Outstanding Young Core Teacher of Colleges and Universities in Anhui Province grant number gxgnfx2020152, the National Natural Science Foundation of China grant number 62172003, the Program for Synergy Innovation in the Anhui Higher Education Institutions of China grant number GXXT-2020-012, and the Natural Science Foundation of Anhui Province grant number 2108085MF217.

References

[1] Y. Xu, O. Ding, R. Qu, and K. Li, "Hybrid multi-objective evolutionary algorithms based on decomposition for wireless sensor network coverage optimization," *Applied Soft Computing*, vol. 68, pp. 268–282, 2018.

[2] X. Wang, J. Ma, S. Wang, and D. Bi, "Distributed energy optimization for target tracking in wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 9, no. 1, pp. 73–86, 2010.

[3] M. F. Othman and K. Shazali, "Wireless sensor network applications: a study in environment monitoring system," *Procedia Engineering*, vol. 41, pp. 1204–1210, 2012.

[4] M. P. Đurišić, Z. Tafa, G. Dimić, and V. Milutinović, "A survey of military applications of wireless sensor networks," in *2012 Mediterranean conference on embedded computing (MECO)*, pp. 196–199, Bar, Montenegro, 2012.

[5] H. Alemdar and C. Ersoy, "Wireless sensor networks for healthcare: a survey," *Computer Networks*, vol. 54, no. 15, pp. 2688–2710, 2010.

[6] J. Wang, C. Ju, Y. Gao, A. K. Sangaiah, and G. J. Kim, "A PSO based energy efficient coverage control algorithm for wireless sensor networks," *Computers, Materials and Continua*, vol. 56, no. 3, pp. 433–446, 2018.

[7] Y. Feng, S. Zhao, and H. Liu, "Analysis of network coverage optimization based on feedback K-means clustering and artificial fish swarm algorithm," *IEEE Access*, vol. 8, pp. 42864–42876, 2020.

[8] O. I. Khalaf, G. M. Abdulsahib, and B. M. Sabbar, "Optimization of wireless sensor network coverage using the bee algorithm," *Journal of Information Science and Engineering*, vol. 36, no. 2, pp. 377–386, 2020.

[9] W. H. Liao, Y. Kao, and Y. S. Li, "A sensor deployment approach using glowworm swarm optimization algorithm in wireless sensor networks," *Expert Systems with Applications*, vol. 38, no. 10, pp. 12180–12188, 2011.

[10] H. Zhao, Q. Zhang, L. Zhang, and Y. Wang, "A novel sensor deployment approach using fruit fly optimization algorithm in wireless sensor networks," in *2015 IEEE Trustcom/Big DataSE/ISPA*, pp. 1292–1297, IEEE Computer Society, 2015.

[11] S. Mirjalili, "The ant lion optimizer," *Advances in Engineering Software*, vol. 83, pp. 80–98, 2015.

[12] L. dos Santos Coelho, G. Maidl, J. Pierezan, V. C. Mariani, M. V. F. da Luz, and J. V. Leite, "Ant lion approach based on Lozi map for multiobjective transformer design optimization," in *2018 international symposium on power electronics, electrical drives, automation and motion (SPEEDAM)*, pp. 280–285, Amalfi, Italy, 2018, June.

[13] A. S. Vikhe and A. A. Kalage, "Power system optimisation using ant lion optimisation technique," in *2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA)*, pp. 355–361, Coimbatore, India, 2019, June.

[14] G. Yogarajan and T. Revathi, "Improved cluster based data gathering using ant lion optimization in wireless sensor networks," *Wireless Personal Communications*, vol. 98, no. 3, pp. 2711–2731, 2018.

[15] P. Yao and H. Wang, "Dynamic adaptive ant lion optimizer applied to route planning for unmanned aerial vehicle," *Soft Computing*, vol. 21, no. 18, pp. 5475–5488, 2017.

[16] S. Saha and V. Mukherjee, "A novel quasi-oppositional chaotic antlion optimizer for global optimization," *Applied Intelligence*, vol. 48, no. 9, pp. 2628–2660, 2018.

[17] S. K. Dinkar and K. Deep, "Opposition based Laplacian ant lion optimizer," *Journal of computational science*, vol. 23, pp. 71–90, 2017.

[18] S. K. Dinkar and K. Deep, "An efficient opposition based Levy flight antlion optimizer for optimization problems," *Journal of computational science*, vol. 29, pp. 119–141, 2018.

[19] L. Wang, W. Wu, J. Qi, and Z. Jia, "Wireless sensor network coverage optimization based on whale group algorithm," *Computer Science and Information Systems*, vol. 15, no. 3, pp. 569–583, 2018.

[20] X. S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *2009 World congress on nature & biologically inspired computing (NaBIC)*, pp. 210–214, Coimbatore, India, 2009, December.

[21] G. G. Wang, S. Deb, A. H. Gandomi, and A. H. Alavi, "Opposition-based krill herd algorithm with Cauchy mutation and position clamping," *Neurocomputing*, vol. 177, pp. 147–157, 2016.

[22] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[23] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, pp. 51–67, 2016.

[24] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.