Mathematical Biosciences and Engineering

MBE, 19(4): 3928–3952.
DOI: 10.3934/mbe.2022181
Received: 08 December 2021
Revised: 19 January 2022
Accepted: 25 January 2022
Published: 11 February 2022

http://www.aimspress.com/journal/MBE

Research article

A hybrid-strategy-improved butterfly optimization algorithm applied to the node coverage problem of wireless sensor networks

Donghui Ma and Qianqian Duan*

Department of Electric and Electronic Engineering, Shanghai University of Engineering Science, Shanghai 201620, China

* **Correspondence:** Email: dqq1019@163.com; Tel: +8615000688602.

Abstract: To increase the node coverage of wireless sensor networks (WSN) more effectively, in this paper, we propose a hybrid-strategy-improved butterfly optimization algorithm (H-BOA). First, we introduce Kent chaotic map to initialize the population to ensure a more uniform search space. Second, a new inertial weight modified from the Sigmoid function is introduced to balance the global and local search capacities. Third, we comprehensively use elite-fusion and elite-oriented local mutation strategies to raise the population diversity. Then, we introduce a perturbation based on the standard normal distribution to reduce the possibility of the algorithm falling into premature. Finally, the simulated annealing process is introduced to evaluate the solution's quality and improve the algorithm's ability, which is helpful to jump out of the local optimal value. Through numerous experiments of the international benchmark functions, the results show the performance of H-BOA has been significantly raised. We apply it to the WSN nodes coverage problem. The results show that H-BOA improves the WSN maximum coverage and it is far more than other optimization algorithms.

Keywords: butterfly optimization algorithm; wireless sensor networks; hybrid strategy; optimized coverage

1. Introduction

Wireless sensor networks (WSN) are widely used in the automation, production, transport, healthcare, and agricultural fields [1–3]. As one core technology in the area of the Internet, wireless sensor networks are composed of multiple sensors in different ways to complete signal collection,

signal transmission, data processing, and other operations [4–6]. WSN coverage problem is one of the basic problems in sensor networks [7,8]. Coverage capability is a criterion for evaluating the monitoring capability of WSN to a monitoring area, that is, how to deploy nodes to reach the maximum networks coverage, so as to improve the stability and effectiveness of information transmission.

Much work has been done to improve the maximum coverage of WSN. Yang et al. [9] improve the traditional centroid location algorithm, design a weighting strategy according to the reference anchor node and raise the location accuracy through multiple weights. From the view of repairing coverage gaps, based on graph theory and geometric calculation, Lu et al. [10] propose a Voronoi polygon strategy to repair local gaps for higher coverage. Based on game theory, Hajjej et al. [11] propose a new topology control method using reinforcement learning (RL) to repair coverage holes in a distributed way, so as to repair coverage vulnerabilities more effectively.

In recent years, an increasing number of scholars have consistently tried to apply intelligent algorithms to solve the WSN coverage problem [12] and have obtained specific results [13]. Kannan et al. [14] introduce the simulated annealing (SA) into the positioning research of wireless sensor networks. The algorithm has strong global optimization ability and robustness by jumping out of local optimal solution with a certain probability. However, the simulated annealing relies excessively on the parameters setting, which leads to the plight that better solution and search time cannot have both Kulkarni et al. [15] introduce particle swarm optimization (PSO) to improve node deployment and data fusion tasks. There is still a disadvantage that it is easy to fall into local optimum. Thus, the classical evolutionary computation cannot meet the needs of the problem. ZainEldin et al. [16] propose a novel improved dynamic deployment technique based-on genetic algorithm (IDDT-GA) to solve this problem. Similarly, aiming at how to minimize the number of nodes and maximize the coverage at the same time, Rebai et al. [17] construct an integer programming model and a special genetic algorithm (GA) is proposed for this model. Shivalingegowda et al. [18] introduce a new algorithm called hybrid gravitational search algorithm with social ski-driver (GSA-SSD). It proves that the hybrid algorithm model has certain advantages in the application of wireless sensor networks. Binh et al. [19] optimize several metaheuristic algorithms for WSN coverage in restricted areas with obstacles. Raj Priyadarshini et al. [20] extend the problem to underwater environment and proposed an energy prediction algorithm based on Markov chain Monte Carlo (MCMC) process for underwater acoustic WSN to reduce coverage vulnerabilities. Khalaf et al. [21] introduce a coverage optimization method for WSN based on the bee algorithm (BA) and compare with the genetic algorithm (GA), the results show BA can spend less time and use less resources to achieve higher coverage. Wang et al. [22] propose a Virtual Force Algorithm-Lévy-Embedded Grey Wolf Optimization (VFLGWO) to solve the WSN coverage problem. The results under different nodes and different monitoring area are discussed respectively in this paper. Feng et al. [23] combine K-means algorithm with artificial fish swarm algorithm, so that a WSN clustering method is proposed, which has better performance compared with traditional methods.

Arora et al. [24] propose a node location scheme using a natural heuristic meta-heuristic algorithm called butterfly optimization algorithm (BOA), which is simulated on sensor networks of different scales and compared to the performance of PSO and firefly algorithm (FA). The results show that BOA can obtain more accurate and stable node positioning. However, I think this work is still not perfect, because the butterfly algorithm has the disadvantages of low optimization accuracy and easy to fall into local optimization. Therefore, a series of work has down to improve the butterfly optimization algorithm and make BOA better with hybrid strategies. Arora et al. [24] combine the artificial bee

colony algorithm (ABC) with the butterfly optimization algorithm and add the Lévy flights strategy to the butterfly optimization algorithm, so that a BOA/ABC algorithm is proposed. When the search phase of BOA is completed, the ABC algorithm start to explore in the search space. To some extent, it speeds up the convergence speed and widens the search space. The core idea of hybrid strategy still lies in accelerating convergence speed and jumping out of local optimization. Utama et al. [25] propose a new hybrid butterfly optimization algorithm for solving green vehicle routing problem (G-VRP). This algorithm is discretized by LRV method and applied to solve discrete optimization problems. By combining the tabu search (TS) algorithm and local search swap and flip strategies, the algorithm gradually approaches the global optimum by continuously flipping and exchanging nodes. Although it can solve the G-VRP well, it still has high time complexity and needs a lot of computing time than BOA. To overcome the problems faced by classical evolutionary computation. Our contributions lie in two aspects:

1) We make a new attempt to integrate five different improvement strategies with BOA, which help to accelerate the convergence speed and improve the optimization accuracy while ensuring the calculation time. We introduce Kent chaotic map in the initialization stage, improve the local search and mutation strategy of butterflies, construct a new inertia weight factor, and propose a simple simulated annealing process based on Standard normal distribution disturbance and Metropolis criterion, which gives full play to the advantages of meta heuristic algorithm.

2) The time complexity, convergence and asymptotic of the algorithm are analyzed in this paper. Through simulation test, the results show the hybrid strategies perform well.

2. WSN coverage optimization model

In this paper, we use Boolean perception model to research the problem. There is a set of N sensor nodes $S = \{s_i, \mathrm{i} = 1,2,3,\cdots n\}$ in a 2D monitoring area of size $\mathrm{L} \times W$. Each node takes (x_i, y_i) as the center of a circle and takes R as the radius. Suppose the monitoring area which is to be covered is discretized into $\mathrm{L} \times W$ pixels, if the target pixel is within the sensor node coverage radius, it is successfully covered. The position of pixel p_j can be expressed as (x_j, y_j), so the distance can be expressed as

$$\mathrm{d}(s_i, p_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{2.1}$$

when the pixel p_j is within the radius of the nodes, p_j is covered successfully. So, we can get the probability which the pixel p_j is covered successfully, as shown in Eq (2.2).

$$p(s_i, p_j) = \begin{cases} 1 & d(s_i, p_j) \le R \\ 0 & d(s_i, p_j) > R \end{cases} \tag{2.2}$$

In particular, the target point only needs to be covered by any node in $S = \{s_i, i = 1,2,3,\cdots n\}$, then the information of this pixel can be detected. The joint perception probability of target point p_j is

$$p(S, p_j) = 1 - \prod_{i=1}^{n} [1 - p(s_i, p_j)] \tag{2.3}$$

In Eq (2.3), if p_j is out of the range of any node s_i in S, then $\prod_{i=1}^{n}[1-p(s_i,p_j)]=1$. So $p(S,p_j)=0$, that is, this pixel is out of WSN coverage range. If p_j is in the range of any node s_i in S, then $p(s_i,p_j)=1, \prod_{i=1}^{n}[1-p(s_i,p_j)]=0$, so $P(S,p_j)=1$, that is, the pixel p_j is perceived by S.

We use coverage rate to express the monitoring ability of sensor nodes. The formula of coverage rate is the ratio of sensor nodes coverage monitoring area A_c to the whole area A, which can be expressed by Eq (2.4).

$$f=\frac{A_c}{A}=\frac{\sum_{j=1}^{L\times W}p(S,p_j)}{L\times W} \tag{2.4}$$

We take Eq (2.4) as the objective function. We give a simple example to illustrate the problem vividly, as shown in Figure 1. Assume that the nodes deployment scheme is shown as the figure, the shaded areas represent covered areas and white areas represent uncovered areas. Only one pixel is not covered so the coverage area of sensor nodes A_c = 35. The number of sensor nodes is 8 and the monitoring area $A=6\times 6=36$. So the coverage rate is

$$coverage=\frac{A_c}{A}=\frac{35}{36}=0.972. \tag{2.5}$$

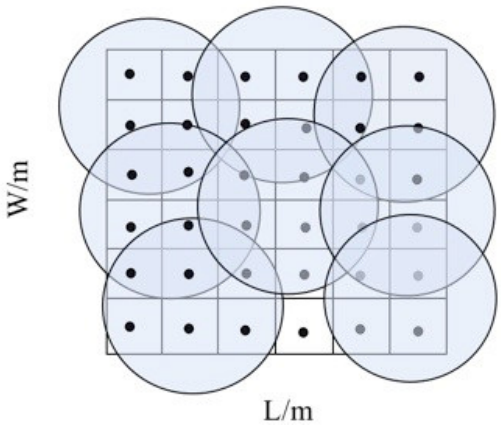


Figure 1. An example of WSN cover model.

3. Butterfly optimization algorithm

The butterfly optimization algorithm (BOA) [26] based on the butterfly foraging behavior, was first proposed by Professor Arora et al. in 2018. In nature, butterflies use multiple sensory organs to find food, such as vision, touch, etc. The most important one is the smell. In BOA, each butterfly can emit a certain concentration of fragrance and smell the fragrance nearby. The butterfly moves towards the most fragrant direction. This phase is known as the global search phase. Suppose one butterfly cannot smell the fragrance from other butterflies. It will move randomly. This phase is known as the local search phase. For the above behaviors, we make a range of assumptions as follows:

a) All the butterflies can disseminate a particular concentration of fragrance.

b) Each butterfly moves only at random or towards the direction of the highest concentration.
c) The intensity of the stimulus is only affected by the objective function.
d) The butterfly controls the local search and the global search by switching the probability p.

The fragrance concentration is closely related to the population's fitness, and related to the following three factors: stimulus intensity I, perceived form c, and power index α. The relationship among them can be expressed as

$$f = cI^{\alpha} \tag{3.1}$$

In the population iteration process, for butterflies moving through search space, the first thing to do is to calculate the fitness of all butterflies and then calculate the fragrance that the group produced at that location through Eq (3.2).

$$x_i{}^{t+1} = x_i{}^{t} + (r^2 \times g^* - x_i{}^{t}) \times f_i \tag{3.2}$$

where x_i^{t+1} is the location x_i for ith butterfly in iteration $t+1$, x_i^t is the location x_i for ith butterfly in iteration t.Here g^* represents the current best location among all the locations in the current stage. The fragrance of ith butterfly is represented by f_i. r, a random number in [0, 1]. The local search phase can be expressed as Eq (3.3).

$$x_i{}^{t+1} = x_i{}^{t} + (r^2 \times x_j{}^{t} - x_k{}^{t}) \times f_i \tag{3.3}$$

where x_j^t and x_k^t are the random butterflies in the solution space from the current population. r, is a random number in [0, 1]. Based on the above description, the process of BOA is as follows:

Step 1: Set each parameter and initialize the population position.
Step 2: Calculate fitness function and find the best population.
Step 3: If $rand > p$,, do global search using Eq (3.2).
Step 4: If $rand < p$ do local search using Eq (3.3).
Step 5: Judge whether butterfly exceeds the search boundary and do boundary treatment.
Step 6: Calculate the fitness of the new population in Step 3 or Step 4, and if it is better than the current solution, update the global optimal solution and the global optimal fitness.
Step 7: If the algorithm meets the maximum iteration, jump out of the program, else jump to Step3.

4. Hybrid butterfly optimization algorithm (H-BOA)

4.1. Kent chaotic map initialization mechanism

Like other classical evolutionary computation algorithms, BOA randomly initializes population, which may cause population unequally distribution at the initial stage. It will enormously affect the speed of searching. Chaotic map is one of the methods to solve this problem. In this paper, we choose Kent chaotic map, which can improve the search speed and increase population diversity.

Chaos phenomenon is one of the main manifestations of complex dynamics of non-linear dynamic mapping, which has great randomness, non-periodicity and other characteristics. So, it is widely used by scholars in the fields of information security, communication encryption and control theory [27–29].

In the field of evolutionary computation, chaotic mapping is often used to replace random number generator to initialize population [30–32].

Logistic chaotic map [33] is a commonly used chaotic map model, which is famous for its simple structure. However, it has a natural disadvantage, that is, it depends too much on the setting of parameters and initial values [34]. Because the chaotic interval is limited and the output chaotic sequence values are not uniformly distributed, the effect is still not ideal when the model is in a full mapping state. Kent chaotic map [35] is designed by

$$x_{k+1} = \begin{cases} \frac{x_k}{\alpha} & 0 \le x \le \alpha \\ \frac{1-x_k}{1-\alpha} & \alpha < x \le 1 \end{cases} \tag{4.1}$$

which combines the advantages of simple structure and uniform distribution of sequence values.

As shown in Figures 2 and 3, when iterating 200 times, there is still a part of the value aggregation extremes under the Logistic chaotic map model compared with the Kent map. To show the advantages of Kent map more clearly, we iterate 1000 times to get the value distribution of the Logistic map and the Kent map as shown in Figure 4, where the X-axis represents the range of values generated, the Y-axis represents the type of chaotic model, and the Z-axis represents the frequency of sequence values generated by the chaotic model in the relevant range.

It is clear from the graph that the Kent chaotic map has better randomness and chaotic characteristics than Logistic chaotic map. So, it can utilize as much information as possible in the solution space.

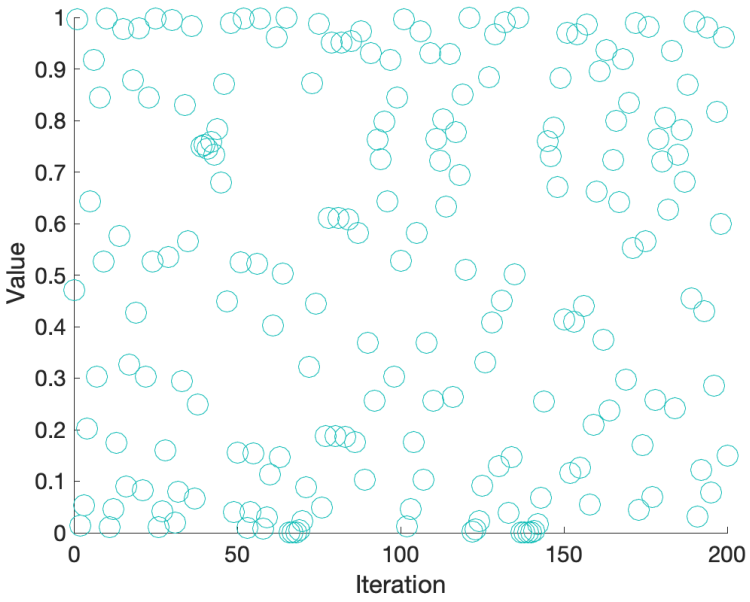


Figure 2. Logistic chaotic map.

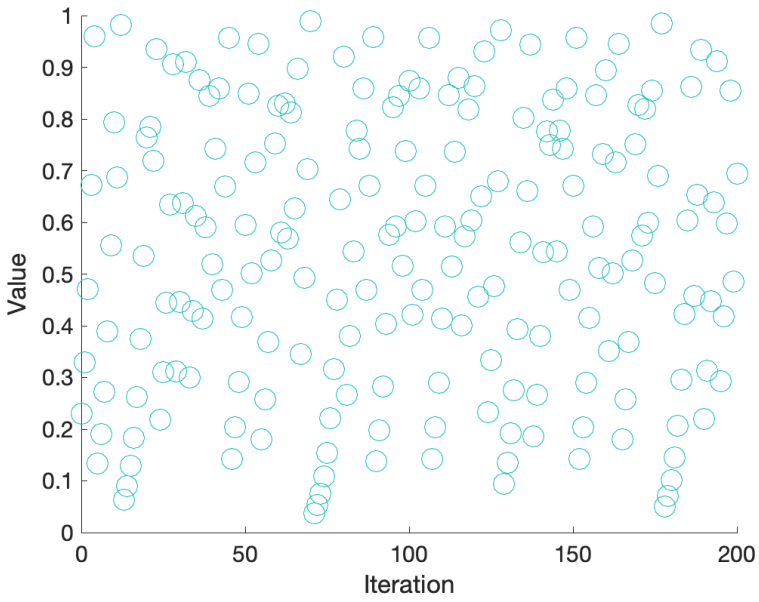


Figure 3. Kent chaotic map.

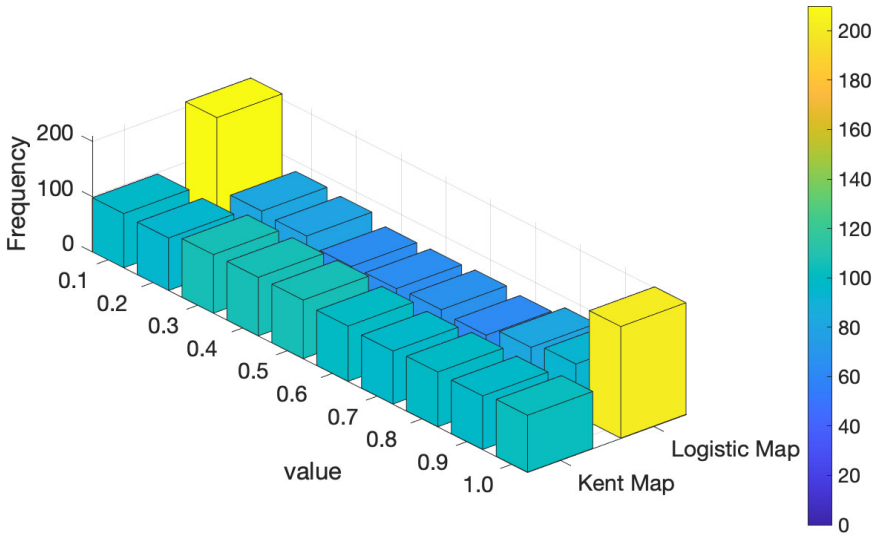


Figure 4. Frequency distribution of Kent map and Logistic map.

4.2. A new inertial weight strategy

BOA is similar to other heuristic swarm intelligence algorithms, facing the balance of the capacities between the global and the local search. A host of scholars use the inertial weight strategy, including linearity weight and nonlinearity weight to solve this problem. Both linearity and nonlinearity weight strategies have been widely used in the improvement of intelligent algorithms.

$$Sigmoid\ output = \frac{1}{1+e^{-x}} \tag{4.2}$$

We consider a better balance between linearity and nonlinearity of Sigmoid function [36], mathematical description of Sigmoid function is shown as Eq (4.2).

We design a new inertial weight based on the Sigmoid function. Inspired by the Paper [37], we add a self-adaptive property to make population search more flexible. The new function is shown as follows

$$w = \frac{1}{(1+\mu e^{\frac{10t}{M}-5})^2} \tag{4.3}$$

where $\mu \in [0, 1]$, t is the current iteration number, M is the maximum iteration number. At the beginning of the iteration, the weight is relatively large. The weight is slowly reduced as the number of iterations increases, which is expected to search in the most extensive possible range. In the middle of the iteration, the weight reduces at a relatively rapid rate to ensure a faster search speed. In the later iteration, the weight slowly decreases so that the local search ability increases, and it gradually approaches the global optimal solution within the solution space. The weight evolution curve depicts as Figure 5.

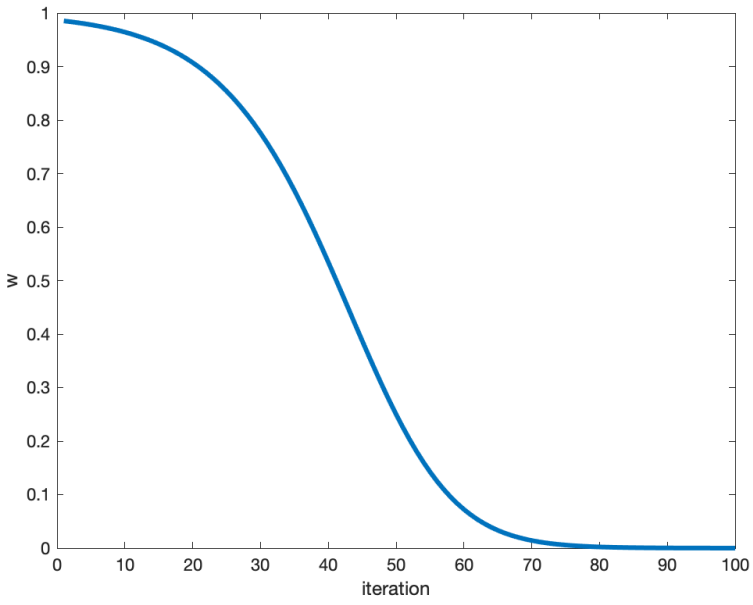


Figure 5. Graph of weight change with iteration number.

Equation (4.4) which is modified from Eq (3.2) described as follows:

$$x_i^{t+1} = w \times x_i^t + (r^2 \times g^* - x_i^t) \times f_i. \tag{4.4}$$

4.3. Elite-fusion and elite-oriented local mutation strategy

In the local search phase of BOA, when a butterfly cannot feel the fragrance of the best butterfly, it can only move towards the butterfly nearby. At this time, the blindness of population limits the search

range to some extent, which would lead individuals to fall into the local optimum. In order to enhance the ability of local search, in this paper, we draw on the idea of genetic algorithm (GA) [38] and differential evolution (DE) [39], propose elite-fusion mutation and elite-oriented mutation strategies, which enlarge the population on the one hand and reduce the possibility of algorithm falling into the local optimum at the end of the iteration on the other hand. Equation (4.5), which is modified from Eq (3.3) described as follows:

$$x_i^{t+1} = \beta \times [x_i^t + (r^2 \times x_j^t - x_k^t)] + (1 - \beta) \times g^* \tag{4.5}$$

$$x_i^{t+1} = g^* + \theta \times (x_j^t - x_k^t) + \gamma \times (x_m^t - x_n^t). \tag{4.6}$$

where x_i^t is the position x_i for tth butterfly in iteration number, x_i^{t+1} is the position x_i for $t+1$th butterfly in iteration number, x_j^t, x_k^t, x_m^t and x_n^t are the random butterflies in the solution space from the current population. θ, β, γ is a number in [0, 1]. Equation (4.6) retains part of the elite solution and merges it with the population generated from the local search phase. They compose a new solution through a certain weight, thus enriching the population diversity.

4.4. Disturbance with standard normal distribution and simulated annealing

In the search process, H-BOA relies on the information of elite butterflies, so the direction of elite butterflies, that is, the direction with the strongest fragrance, is very important. Normal distribution disturbance is a way of random disturbance term, which is usually used in optimization problem [40,41]. In nature, normal distribution is one of the objective laws of nature. Introducing positive distribution to disturb the population is in line with the objective law of swarm intelligence algorithm. Therefore, adding a normal distribution disturbance term to the new location individuals generated in the search stage will not destroy the search law and expand the search space to a certain extent.

Simulated annealing (SA) is a heuristic algorithm proposed by Professor Metropolis in 1953. It is based on the principle of solid annealing [42]. The most commonly used is the Metropolis criterion, which says that the algorithm dynamically generates a probability to determine whether to accept an inferior solution at each iteration.so it will help the algorithm jump out of the local optimum effectively. This shows excellent global optimization.

If the complete simulated annealing algorithm is combined with BOA, it will increase the time complexity of the algorithm and reduce the convergence speed of the algorithm. Therefore, we adopt the core idea of Metropolis criterion to help the algorithm jump out of the local optimal solution while ensuring the convergence speed. In this paper, the initial solution of the simulated annealing process is the population after the search stage. We add the population subject to the disturbance of standard normal distribution to replace the process of generating random solutions in the simulated annealing process, as well as combining it with the simulated annealing process. The new solution is explored around the optimal individual through Metropolis criterion, so the population can make full use of the characteristic information reflected by the best butterfly and guide the whole group to evolve towards the optimal solution. At the same time, it has the ability to jump out of the local optimum and great convergence speed. It can constantly modify and refine the evolution direction.

Assume that T_t is the current temperature, T_0 is the initial temperature, the maximum number of iterations is M, and the current iteration is t. We parallel it to the search phase. The flow chart of this stage is shown in Figure 6.

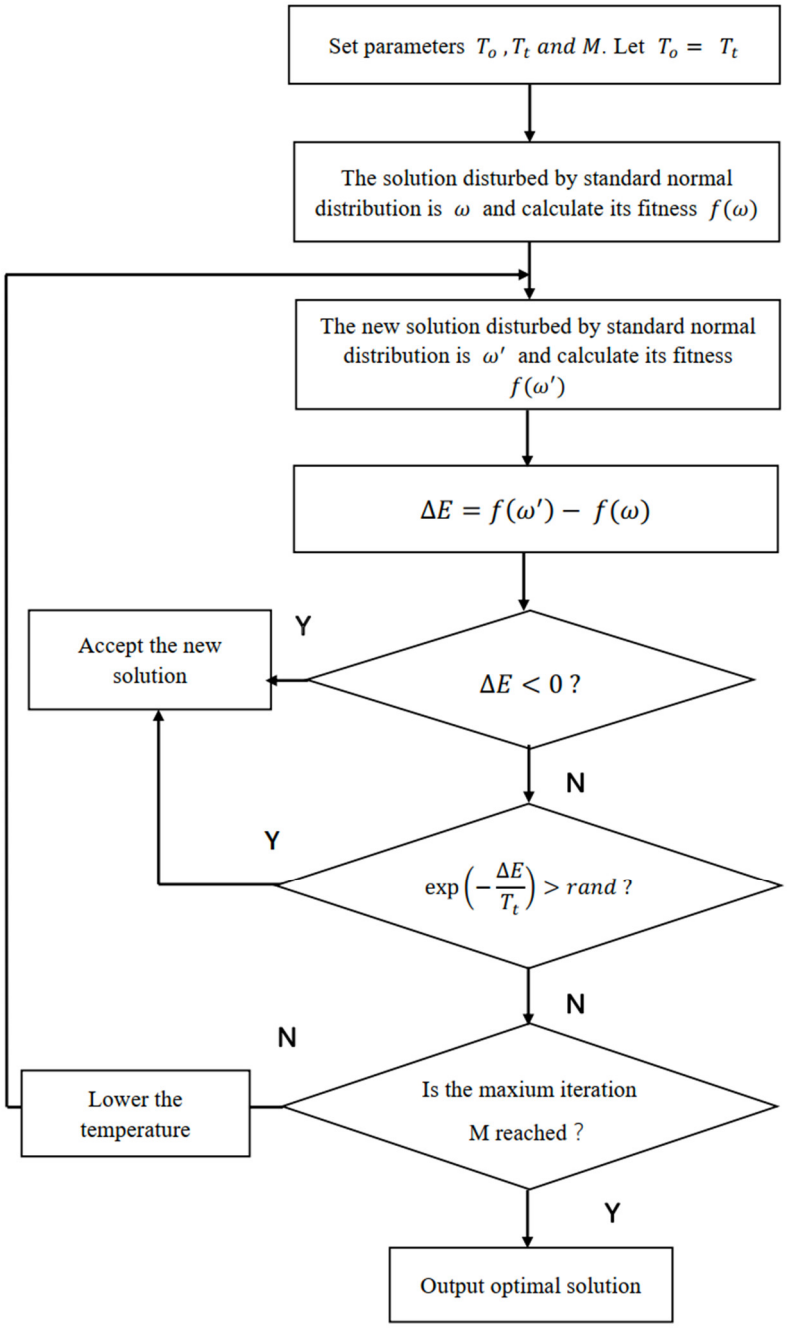


Figure 6. Flow chart of normal distribution disturbance and Metropolis criterion.

4.5. The Pseudocode of H-BOA

Based on the above description, the Pseudocode of H-BOA can be described as follows:

Algorithm 1 A Hybrid-Strategy-Improved Butterfly Optimization Algorithm

Input: Population size N,switch probability p, p_c, sensor modality c, power exponent α, maximum iterations M and parameters $\beta, \gamma, \theta, \mu, T_0$

Output: Best solution X^*

```
Initialize the butterfly population F with Kent chaotic map
for t ← 1 to M do
    for i ← 1 to N do
        Calculate fitness of F
    end for
    Find the best solution as X^best
    for i ← 1 to N do
        Generate a random number r from [0, 1]
        if r ≤ p then
            Do global search using Eq (4.4)
        else
            If r > p_c then
                Do elite-fusion local mutation strategy using Eq (4.5)
            else
                Do elite-oriented local mutation strategy using Eq (4.6)
            end if
        end if
    end for
    Perform disturbance with Standard Normal Distribution
    Perform simulated annealing process according to Metropolis criterion
    Update the best solution
end for
```

4.6. Algorithm complexity analysis

Time complexity of an algorithm is one of the criteria for evaluating the performance of an algorithm. In BOA, assume that the number of populations is N, the objective function is f(x), and the dimension is n. According to the analysis of BOA, the time complexity of BOA is *O(n + f(n))*.

For H-BOA, in the initial population phase, assume the time for initializing each parameter is t_1, the time for initializing the population position using Kent chaotic map is t_2, the time for calculating fitness according to the population position is $f(n)$, and the time to save the current optimal solution is t_3, so the total time complexity in this phase is

$$O(N(nt_2 + t_3 + f(n) + t_1) = O(n + f(n)).$$

When entering the iteration, assume the time for updating new inertia weight is t_4. The time to calculate the individual fragrance concentration is t_5.

1) Global search phase: Assume that the time used to update each dimension of the butterfly position is t_6 according to Eq (4.4), the time to calculate the fitness of the new population is $f(n)$, the time to compare the fitness of the latest and old butterfly is t_7. Then the total time complexity of the global search phase is

$$O(N(t_4 + t_5 + nt_6 + t_7 + f(n)) = O(n + f(n)).$$

2) Local search phase: The population uses two mutation methods to perform local wandering. The time to generate a random number p c is t_8, the time used to update each dimension of the butterfly is t_9 and t_{10} according to Eqs (4.5) and (4.6), the time to calculate new population fitness is f(n). The time for comparing and replacing the fitness of the old and new population is the same as that of the global search phase, which is t_7. Then the total time complexity of this stage is

$$O(N(t_4 + t_7 + n(t_8 + t_9 + t_{10}) + f(n))) = O(n + f(n)).$$

Standard normal disturbance and simulated annealing stage: Assume that the time to generate a normal distribution of random numbers is t_{12} , the time used to calculate each dimension of the butterfly after disturbance is t_{13}, the fitness of the new population is calculated as $f(n)$ and the time complexity of the simulated annealing stage is $O(Nn)$, then the total time complexity of this stage is

$$O(N(t_{12} + t_{13}) + f(n)) + O(Nn) = O(n + f(n)).$$

Record the optimal solution stage: Assume that the comparison time of each butterfly's fitness with the current optimal solution is t_{14} and the time to replace the new solution in the simulated annealing phase is t_{15} .so the total time complexity of this stage is

$$O(N(t_{14} + t_{15})) = O(n).$$

Therefore, the total time complexity of H-BOA is $O(n + f(n))$, which does not increase the extra time complexity compared with BOA.

The space complexity is mainly affected by dimensionality and population size, so the space complexity of the two algorithms is $O(Nn)$.

4.7. Asymptotic analysis and convergence analysis of the algorithm

In this section, we analyze the asymptotic and convergence of H-BOA. In addition, the definitions and theorems are provided.

Definition 1. The objective function $f: R^n \to R$ is a continuous function on the non-empty feasible region S and there is an optimal solution set $\Omega = \{x^* | x^* \in S, f(x^*) < \delta\}$ in S, δ is an acceptable fitness.

Theorem 1. H-BOA has progressive fitness. Assume that the optimal butterfly position found in the tth is x_t^*, the non-negative random process generated by H-BOA is $\{d_t \mid = f(x_t^*) - f(x^*), 1 \leq t \leq T\}$. When $f(x_t^*) > f(x^*)$, there is a normal number τ, which makes $E(d_{t+1}) \leq E(d_t) - \tau$. Then we can say the optimization algorithm has progressive fitness [43].

Proof. According to the analysis in Section 4.4, the new solution generated by normal distribution disturbance and simulated annealing process is not necessarily closer to the global optimal solution, but the previous optimal position will be recorded during optimization, that is, the original better position will not be covered. Therefore, the distance between the new solution and the global optimal solution will not increase after annealing, that is, the direction of population evolution is monotonous.

Because the core purpose of H-BOA is to find the most fragrant butterfly so that $P(f(x_{t+1}^*) - f(x_t^*) > 0) = 0$. The butterfly population updates individual positions in three ways and all of them have random factors. So $P(f(x_{t+1}^*) = f(x_t^*)) \neq 1$. From the above derivations we can get

$$P(f(x_{t+1}^*) - f(x_t^*) < 0) > 0.$$

Let $E[f(x_{t+1}^*) - f(x_t^*)] = -\tau_{t+1},\ where\ \tau_{t+1} > 0.$ So

$$E(d_{t+1} - d_t) = E\left[\left(f(x_{t+1}^*) - f(x^*)\right) - \left(f(x_t^*) - f(x^*)\right)\right]$$

$$= E(f(x_{t+1}^*) - f(x_t^*))$$

$$= -\tau_{t+1}.$$

That is $E(d_{t+1}) = E(d_t) - \tau_{t+1}$. Finally, let $\tau = min\{\tau_1, \tau_2, \cdots, \tau_T\}$, so we can get

$$E(d_{t+1}) \le E(d_t) - \tau.$$

So, the algorithm has progressive fitness.

In the next step, we will analyze the convergence of H-BOA and the definitions and theorems are provided [44,45].

Definition 2. There is a set of random sequences $\xi_t\ (t = 1, 2, \cdots)$ in the probability space, if there is a random variable ξ when any $\varepsilon > 0$ is satisfied, it makes

$$\lim_{t\to\infty} P\{|\xi_t - \xi| < \varepsilon\} = 1$$

or the following equation is qualified, namely

$$P\{\bigcup\nolimits_{n=1}^{\infty} \bigcap\nolimits_{k\ge n}^{\infty} [|\xi_t - \xi| \ge \varepsilon]\} = 0.$$

Then the random sequence $\{\varepsilon_t\}$ is said to converge to Random variable ε with probability 1.

Lemma 1: (Borel-Cantelli Lemmas): $Let\ A_1, A_2, \cdots F$

1) If $\sum_{n=1}^{\infty} \mathrm{P}(A_n) < \infty$,then $\mathrm{P}(limsup_n A_n) = 0$.

2) $If\ \sum_{n=1}^{\infty} \mathrm{P}(A_n) = \infty$ and $\{A_n\}$ are independent, then $\mathrm{P}(limsup_n A_n) = 1$.

Theorem 2. Let $\{X(t)\}$ be the position sequence of H-BOA. $\{X_g(t)\} \in X(t)$ be the best solution sequence. Assume that ΔE generated in the simulated annealing stage satisfies $\Delta E \sim N(\mu, \sigma^2)$ If Definition 2 is satisfied, namely

$$P\{\lim_{t\to\infty} f(x_g(t)) = X^*\} = 1.$$

We can call that H-BOA converges to the global optimum with probability 1.

Proof. Let $p(k) = \prod_{t=1}^{k} P\{|f(X_g(t)) - X^*| \ge \varepsilon\}$ where $\varepsilon > 0$. Here X^* is the global optimal solution. Suppose $X^* = min\{f(x), x \in \Omega\}$, then

$$p(k) = \prod_{t=1}^{k} P\{\left|f\left(X_g(t)\right) - X^*\right| \ge \varepsilon\}$$

$$= \prod_{t=1}^{k} P\{f\left(X_g(t)\right) - X^* \ge \varepsilon\}$$

$$= \prod_{t=1}^{k} P\{\Delta f(X_g(t)) \geq \varepsilon + X^* - f(X_g(t-1))\}.$$

So, we can get an equation from $\Delta f(X_g(t)) \sim N(\mu_1, \sigma_1^2)$ shown as

$$P(k) = \prod_{t=1}^{k} \int_{\varepsilon + X^* - f(X_g(t-1))}^{\infty} \frac{1}{\sqrt{2\pi}\sigma_1} e^{\frac{-x^2}{2\pi\sigma_1^2}} \mathrm{d}t,$$

then let

$$b = max\{\int_{\varepsilon + X^* - f(X_g(t-1))}^{\infty} \frac{1}{\sqrt{2\pi}\sigma_1} e^{\frac{-x^2}{2\pi\sigma_1^2}} \mathrm{d}t\}.$$

According to the relevant knowledge of the infinite series, we can get

$$\sum_{k=1}^{\infty} p(k) \leq \sum_{k=1}^{\infty} b^k = \frac{1}{1-b} < \infty,$$

From Lemma 1 we can get

$$P\{\bigcup_{n=1}^{\infty} \bigcap_{k \geq n}^{\infty} |f(X_g(t)) - X^*| \geq \varepsilon\} = 0.$$

Finally, it can be seen that $f(X_g(t)$ converges to X^* with probability 1 from Definition 2.

5. Hybrid butterfly optimization algorithm (H-BOA)

5.1. Simulation experiment environment

The experimental environment is Windows10, 64-bit operating system, CPU is Intel Core i510400H, main frequency is 3.2 GHz, memory is 8 G, and the algorithm is based on MATLAB2020b.

5.2. International benchmark function test

To verify the performance of H-BOA, we select ten international benchmark functions shown in Table 2, where f_1, f_2, f_3, f_4, f_5, f_6 are unimodal functions, f_7, f_8, f_9, f_{10} are multimodal functions. We compared H-BOA with particle swarm optimization (PSO), BOA, a new inertia weight proposed in Section 4.2 improved butterfly optimization algorithm (IBOA) and butterfly optimization algorithm of Cauchy mutation and adaptive weight optimization (CWBOA) [46]. To keep the fairness and objectivity, the initial population size is all uniformly set to 30, the number of iterations is set to 500, and the part of parameters are the same as those in CWBOA. To reduce the randomness, we conducted 30 experiments and took the average and standard. The specific algorithm parameter settings are shown in Table 1. The data of CWBOA is directly derived from the paper [46].

Table 1. Caption of the table.

Algorithm	parameter settings
H-BOA	$p = p_c = 0.8, \beta = 0.3, \theta = \gamma = 0.1, \mu = 1, T_0 = 1000$
IBOA	$p = 0.8, I = 0.01, \alpha = 0.1$
BOA	$p = 0.8, I = 0.01, \alpha = 0.1$
CWBOA	$p = 0.8, I = 0.01, \alpha = 0.1$
PSO	$c_1 = c_2 = 2, Vmax = 1, Vmin = -1$

Table 2. International text function.

Function	dim	range	best value
$f_1 = \sum_{i=1}^{n} x_i^2$	30	(-100,100)	0
$f_2 = max\ (\lvert x_i \rvert, 1 < i < n)$	30	(-100,100)	0
$f_3 = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	(-10,10)	0
$f_4 = \sum_{i=1}^{n} (\sum_{j=1}^{n} x_j)^2$	30	(-100,100)	0
$f_5 = \sum_{i=1}^{n} ix^4 + random(0,1)$	30	(-1.28,1.28)	0
$f_6 = 0.26(x_1^2 + x_2^2) - 0.48x_1x_2$	30	(-10,10)	0
$f_7 = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30	(-5.12,5.12)	0
$f_8 = \sum_{i=1}^{d} x_i^2 + \left(\sum_{i=1}^{d} 0.5ix_i\right)^2 + \left(\sum_{i=1}^{d} 0.5ix_i\right)^4$	30	(-5,10)	0
$f_9 = \frac{1}{4000}\sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	(-600,600)	0
$f_{10} = -20\,exp\left(-0.2 \times \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i)\right) + 20 + e$	30	(-32,32)	0

5.3. Experimental results and analysis

To measure the convergence and accuracy of H-BOA, Table 3 shows the optimization results of the five algorithms on the test functions. "-" represents the test function is not included in the original paper. Figure 7 shows the iterative convergence curves of the four algorithms where the X-axis represents the number of iterations and the Y-axis represents the fitness.

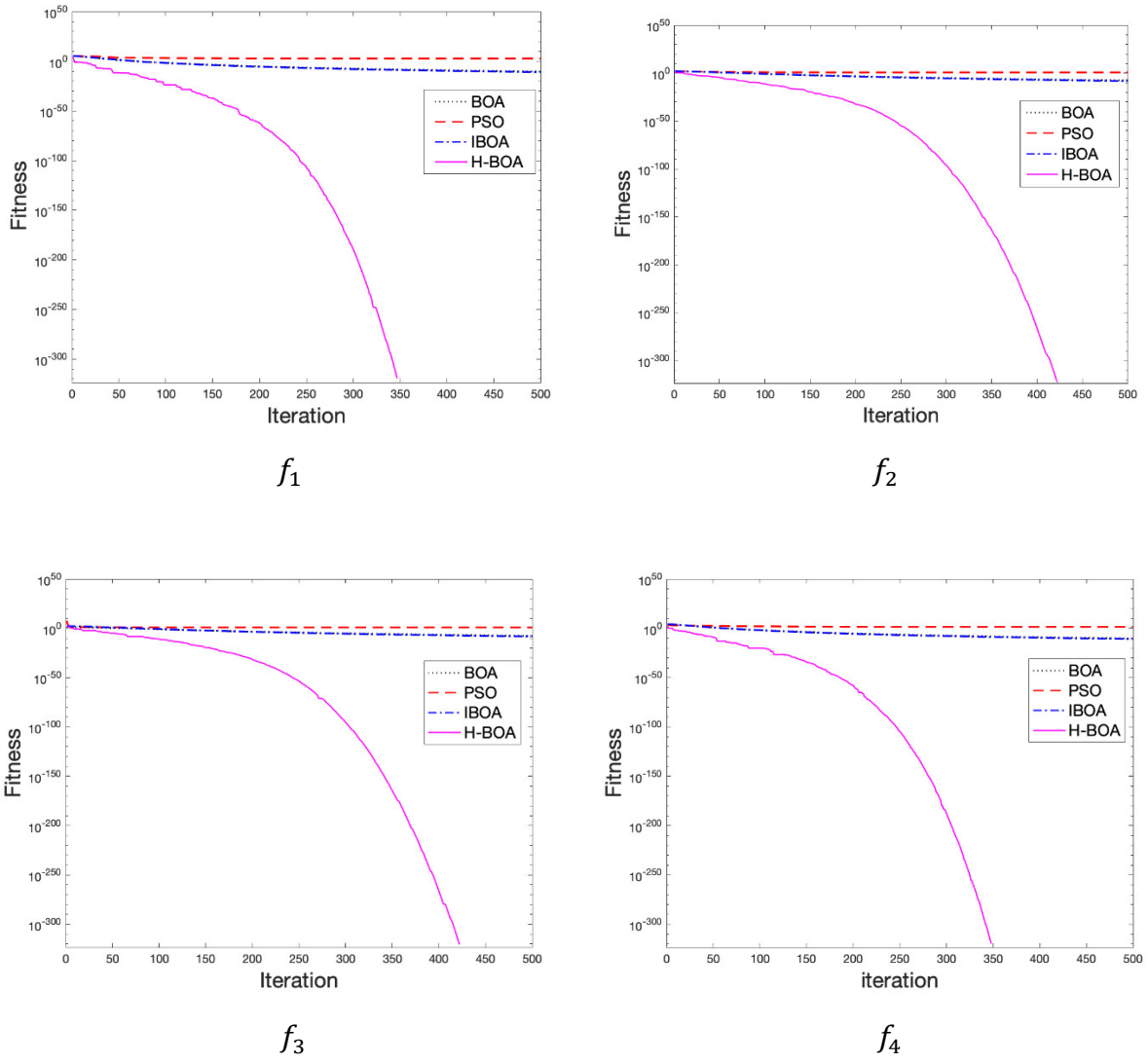
Table 3. Function optimization results.

Function		HBOA	IBOA	CWBOA	BOA	PSO
f_1	Mean	0	1.36E-11	0	5.70E-11	8.81E+02
	Std	0	5.90E-13	0	6.59E-12	1.72E+02
f_2	Mean	0	1.17E-11	3.29E-134	5.53E-11	1.08E+03
	Std	0	8.53E-13	1.80E-133	5.30E-12	6.54E+02
f_3	Mean	0	5.99E-09	3.86E-134	1.94E-08	5.49E+00
	Std	0	4.70E-10	1.52E-133	2.52E-09	8.36E-01
f_4	Mean	0	2.91E-09	-	1.75E-08	1.05E+01
	Std	0	1.69E-09	-	2.59E-09	2.87E+00
f_5	Mean	8.71E-05	1.15E-03	1.57E-04	1.63E-03	2.47E-01
	Std	7.24E-05	2.89E-04	1.35E-04	7.81E-04	1.40E-01
f_6	Mean	0	9.79E-13	0	1.78E-12	5.43E-84
	Std	0	3.18E-13	0	2.50E-13	9.22E-84
f_7	Mean	0	2.27E-14	0	1.30E+02	1.04E+02
	Std	0	3.11E-14	0	7.49E+01	1.62E+01
f_8	Mean	0	1.12E-11	0	5.39E-11	1.25E+04
	Std	0	1.27E-12	0	7.27E-12	9.39E+03
f_9	Mean	0	8.04E-12	0	3.28E-11	9.14E+00
	Std	0	6.77E-13	0	1.14E-11	4.32E+00
f_{10}	Mean	8.88E-16	5.72E-09	8.88E-16	1.93E-08	1.02E+01
	Std	0	7.95E-11	0	1.12E-09	8.39E-01

It can be seen from Table 3 that H-BOA has found the optimal function value 0 except f_5 and f_{10} and the optimization effect reaches 100%. For function f_5, the average accuracy of H-BOA is two orders of magnitude better than BOA, and one order of magnitude better than CWBOA. As for the

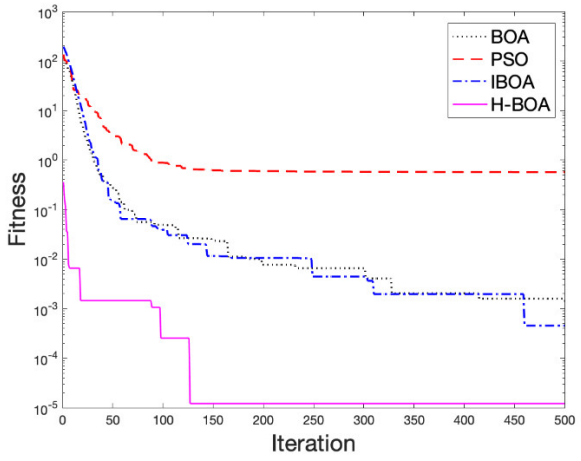
function f_{10}, the average accuracy of H-BOA is seven orders of magnitude better than that of BOA, which is the same as the result of CWBOA and the variance is 0. The algorithm has good stability and robustness. In addition, as can be seen from the results that the performance of IBOA is generally better than that of BOA, which shows the new inertia weight strategy is feasible.

As can be seen from Figure 7, when solving different functions, the curve of the H-BOA is smoother, the convergence speed is faster, the optimization accuracy is higher and the curve inflection point appears firstly, which indicating that H-BOA has a stronger ability to overcome the local optimum. Further prove the effectiveness of hybrid strategies.

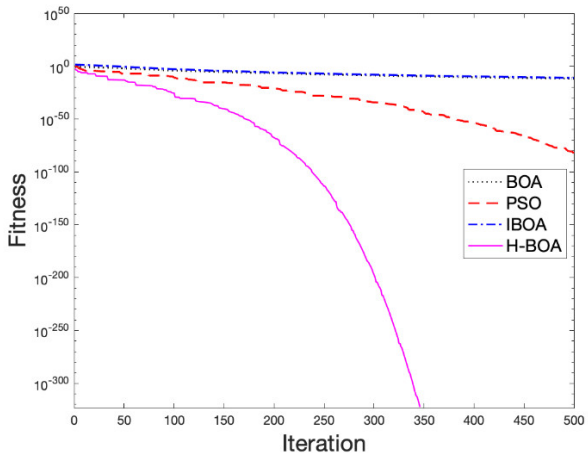


Continued on next page

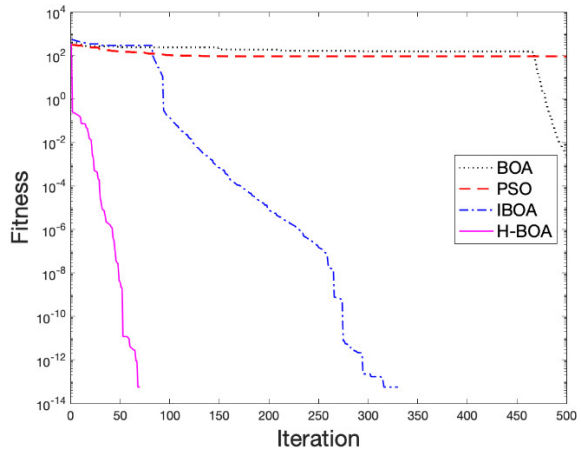
Figure 7. Convergence curve for four algorithms.



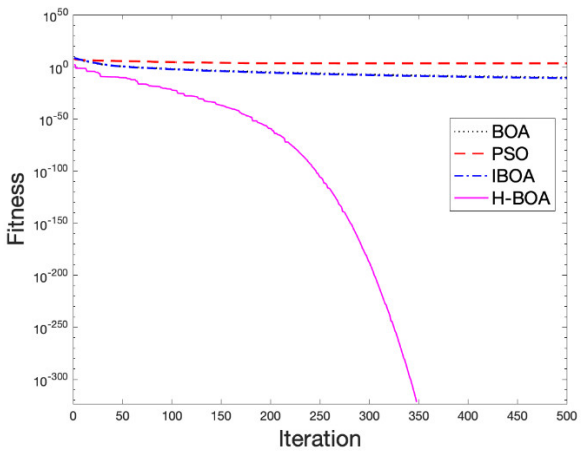
f_5



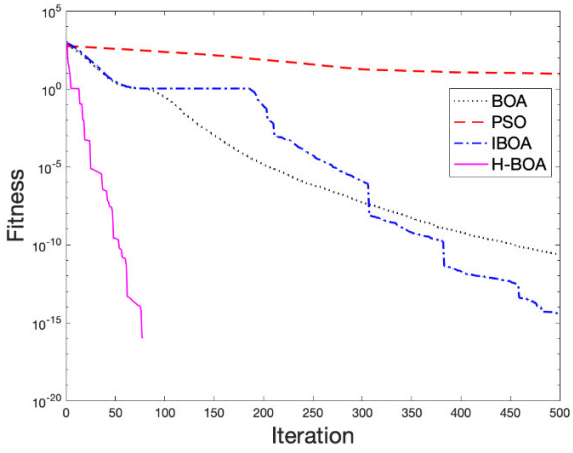
f_6



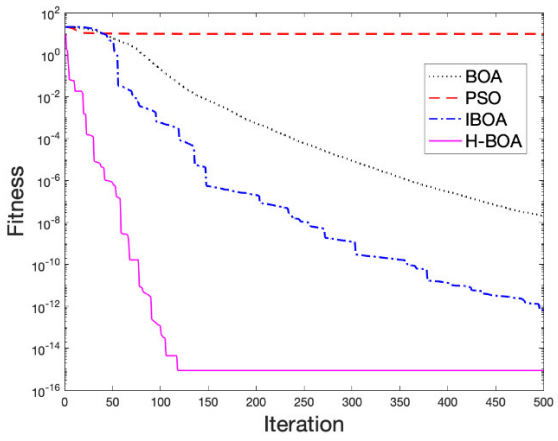
f_7



f_8



f_9



f_{10}

Figure 7. Convergence curve for four algorithms.

5.4. WSN coverage optimization simulation experiment

The simulation experiment uses MATLAB, and the environment settings are the same with international function test. We choose H-BOA, BOA, the single-strategy improved butterfly optimization algorithm (IBOA) and the sparrow search algorithm (SSA) for comparison. The algorithm parameters are shown as Table 4, where the maximum number of iterations is $M = 100$, and the population size is $N = 30$.

To show the effectiveness of H-BOA, in this paper, we tried to do multiple experiments with different numbers of nodes. The experimental parameter settings are shown in Table 5. Figure 8 shows the simulation coverage figure. Figure 9 describes the maximum coverage percent that various algorithm can achieve and Table 6 shows the specific maximum coverage value. Figure 10 shows the evolution curve of the maximum coverage of WSN with the number of population iterations when the number of nodes is 25, as can be seen from the figure that H-BOA has more uniform distribution and less gaps. This fully shows higher convergence and better robustness. In particular, for some wireless sensor node deployment tasks that require significantly higher coverage, the optimization strategy of the HBOA algorithm can achieve more excellent optimization with fewer nodes in less time.

In summary, H-BOA has better performance in WSN node coverage application and can improve energy effectiveness and real-time schedule.

Table 4. Algorithm parameter setting.

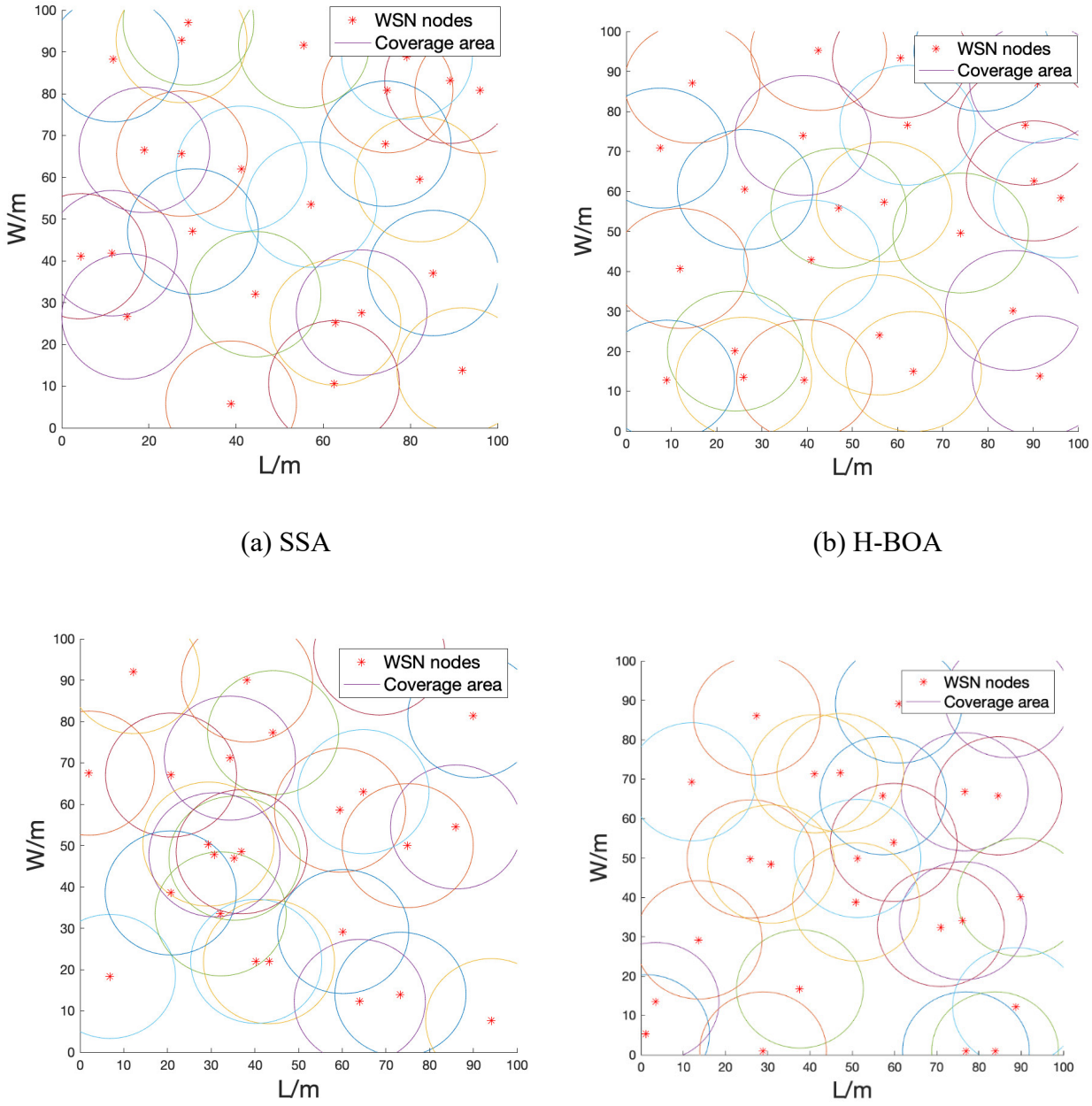
Algorithm	parameter settings
H-BOA	$p = p_c = 0.8, \beta = 0.3, \theta = \gamma = 0.1, \mu = 1, T_0 = 1000$
IBOA	$p = 0.8, I = 0.01, \alpha = 0.1$
BOA	$p = 0.8, I = 0.01, \alpha = 0.1$
SSA	$ST = 0.6, PD = 0.7, SD = 0.2$

Table 5. Parameter setting.

Parameter settings	value
Region Area	100m × 100m
Pixels	100 × 100
Nodes	N = 10,15,20,25,30
Perceived Radius	R = 15m

Table 6. Comparison of maximum coverage rate with different number of nodes.

Algorithm	Number of nodes			Maximum coverage (%)	
	N = 10	N = 15	N = 20	N = 25	N = 30
SSA	62.14	78.51	83.42	92.60	94.50
H-BOA	**64.61**	**80.05**	**91.21**	**95.78**	**98.34**
BOA	56.82	71.81	82.47	90.91	94.13
IBOA	56.68	74.42	83.07	91.60	94.37



(a) SSA (b) H-BOA

(c) BOA (d) IBOA

Figure 8. WSN nodes coverage graph for four algorithms.

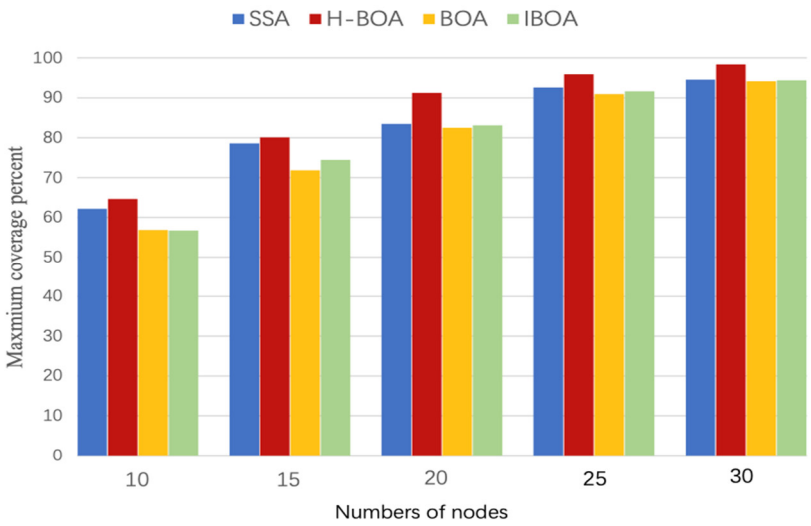


Figure 9. Comparison of different node coverage.

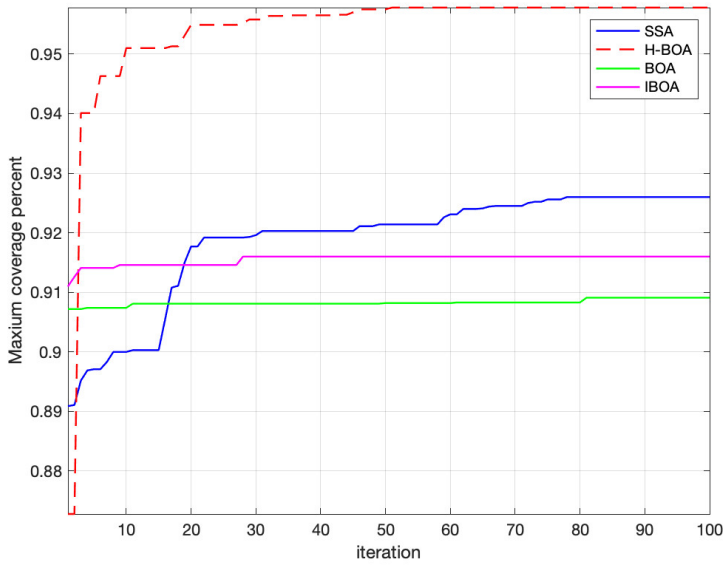


Figure 10. Evolution curve.

6. Conclusions

Based on BOA, we propose a hybrid-strategy-improved butterfly optimization algorithm (HBOA). First, H-BOA uses Kent chaotic map to initialize the population to keep a more balanced search space. Next, we introduce a new inertial weight modified from the Sigmoid function to increase the fixability of global search and local search. Then the introduction of elite-fusion and elite-oriented strategy aims to improve population diversity. Finally, we adopt the disturbance based on standard normal distribution to prevent the algorithm from falling into precociousness and use simulated annealing process to ensure the quality of the solution. The above improvement points observably improve the performance of the algorithm. We selected the international test functions for performance testing and

compared them with BOA, a single-strategy-improved butterfly optimization algorithm (IBOA) and particle swarm optimization (PSO). The results show that H-BOA is better than other algorithms, together with higher accuracy and robustness. We apply H-BOA in WSN coverage research, which proves that this algorithm can effectively solve this type of problem and broaden the application field. However, the algorithm still has some shortcomings. Our next work is how to optimize the strategy to make the algorithm more expressive and propose more effective hybrid heuristic algorithms for different and more complex problems.

Acknowledgments

All sources of funding of the study must be disclosed.

Conflict of interest

The authors declare there is no conflict of interest.

References

1. I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, A survey on sensor networks, *IEEE Commun. Mag.*, **40** (2002), 102–114. https://doi.org/10.1109/MCOM.2002.1024422
2. I. F. Akyildiz, M. C. Vuran, *Wireless sensor networks*, John Wiley & Sons, 2010. https://doi.org/10.1002/9780470515181
3. J. Ko, C. Lu, M. B. Srivastava, J. A. Stankovic, A. Terzis, M. Welsh, Wireless sensor networks for healthcare, *Proc. IEEE*, **98** (2010), 1947–1960. https://doi.org/10.1109/JPROC.2010.2065210
4. S. Havedanloo, H. R. Karimi, Improving the performance metric of wireless sensor networks with clustering Markov chain model and multilevel fusion, *Math. Probl. Eng.*, **2013** (2013). https://doi.org/10.1155/2013/783543
5. Z. A. A. Aziz, S. Y. A. Ameen, Air pollution monitoring using wireless sensor networks, *J. Inf. Technol. Inf.*, **1** (2021), 20–25. Available from: https://www.qabasjournals.com/index.php/jiti/article/view/26.
6. A. C. Djedouboum, A. A. Abba Ari, A. M. Gueroui, A. Mohamadou, Z. Aliouat, Big data collection in large-scale wireless sensor networks, *Sensors*, **18** (2018), 4474. https://doi.org/10.3390/s18124474
7. I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, Wireless sensor networks: a survey, *Comput. Networks*, **38** (2002), 393–422. https://doi.org/10.1016/S1389-1286(01)00302-4
8. Y. Tian, Y. Ou, H. R. Karimi, Y. T. Liu, J. Q. Han, Distributed multitarget probabilistic coverage control algorithm for wireless sensor networks, *Math. Probl. Eng.*, **2014** (2014). https://doi.org/10.1155/2014/421753
9. X. Yang, X. Wang, W. Wang, An improved centroid localization algorithm for WSN, in *2018 IEEE 4th Information Technology and Mechatronics Engineering Conference (ITOEC)*, (2018), 1120–1123. https://doi.org/10.1109/ITOEC.2018.8740471
10. X. Lu, Y. Su, Q. Wu, Y. Wei, J. Wang, An improved coverage gap fixing method for heterogenous wireless sensor network based on voronoi polygons, *Alexandria Eng. J.*, **60** (2021), 4307–4313. https://doi.org/10.1016/j.aej.2021.03.007

11. F. Hajjej, M. Hamdi, R. Ejbali, M. Zaied, A distributed coverage hole recovery approach based on reinforcement learning for wireless sensor networks, *Ad Hoc Networks*, **101** (2020), 102082. https://doi.org/10.1016/j.adhoc.2020.102082
12. K. Tarnaris, I. Preka, D. Kandris, A. Alexandridis, Coverage and k-coverage optimization in wireless sensor networks using computational intelligence methods: a comparative study, *Electronics*, **9** (2020), 675. https://doi.org/10.3390/electronics9040675
13. A. Singh, S. Sharma, J. Singh, Nature-inspired algorithms for wireless sensor networks: a comprehensive survey, *Comput. Sci. Rev.*, **39** (2021), 100342. https://doi.org/10.1016/j.cosrev.2020.100342
14. A. A. Kannan, G. Mao, B. Vucetic, Simulated annealing based localization in wireless sensor network, in *The IEEE Conference on Local Computer Networks 30th Anniversary (LCN'05)l*, (2005), 2–514. https://doi.org/10.1109/LCN.2005.125
15. R. V. Kulkarni, G. K. Venayagamoorthy, Particle swarm optimization in wireless-sensor networks: a brief survey, *IEEE Trans. Syst.*, **41** (2010), 262–267. https://doi.org/10.1109/TSMCC.2010.2054080
16. H. ZainEldin, M. Badawy, M. Elhosseini, H. Arafat, A. Abraham, An improved dynamic deployment technique basedon genetic algorithm (iddt-ga) for maximizing coverage in wireless sensor networks, *J. Ambient Intell. Humanized Comput.*, **11** (2020), 4177–4194. https://doi.org/10.1007/s12652-020-01698-5
17. M. Rebai, M. L. Berre, H. Snoussi, F. Hnaien, L. Khoukhi, Sensor deployment optimization methods to achieve both coverage and connectivity in wireless sensor networks, *Comput. Oper. Res.*, **59** (2015), 11–21. https://doi.org/10.1016/j.cor.2014.11.002
18. C. Shivalingegowda, P. V. Y. Jayasree, Hybrid gravitational search algorithm based model for optimizing coverage and connectivity in wireless sensor networks, *J. Ambient Intell. Humanized Comput.*, **12** (2021), 2835–2848. https://doi.org/10.1007/s12652-020-02442-9
19. H. T. T. Binh, N. T. Hanh, L. V. Quan, N. D. Nghia, N. Dey, Metaheuristics for maximization of obstacles constrained area coverage in heterogeneous wireless sensor networks, *Appl. Soft Comput.*, **86** (2020), 105939. https://doi.org/10.1016/j.asoc.2019.105939
20. R. R. Priyadarshini, N. Sivakumar, Enhancing coverage and connectivity using energy prediction method in underwater acoustic WSN, *J. Ambient Intell. Humanized Comput.*, **11** (2020), 2751–2760. https://doi.org/10.1007/s12652-019-01334-x
21. O. I. Khalaf, G. M. Abdulsahib, B. M. Sabbar, Optimization of wireless sensor network coverage using the bee algorithm, *J. Inf. Sci. Eng.*, **36** (2020), 377–386.
22. S. Wang, X. Yang, X. Wang, Z. Qian, A virtual force algorithm-lévy-embedded grey wolf optimization algorithm for wireless sensor network coverage optimization, *Sensors*, **19** (2019), 2735. https://doi.org/10.3390/s19122735
23. Y. Feng, S. Zhao, H. Liu, Analysis of network coverage optimization based on feedback k-means clustering and artificial fish swarm algorithm, *IEEE Access*, **8** (2020), 42864–42876. https://doi.org/10.1109/ACCESS.2020.2970208
24. S. Arora, S. Singh, Node localization in wireless sensor networks using butterfly optimization algorithm, *Arabian J. Sci. Eng.*, **42** (2017), 3325–3335. https://doi.org/10.1007/s13369-017-2471-9
25. D. M. Utama, D. S. Widodo, M. F. Ibrahim, S. K. Dewi, A new hybrid butterfly optimization algorithm for green vehicle routing problem, *J. Adv. Transp.*, **2020** (2020). https://doi.org/10.1155/2020/8834502

26. S. Arora, S. Singh, Butterfly optimization algorithm: a novel approach for global optimization, *Soft Comput.*, **23** (2019), 715–734. https://doi.org/10.1007/s00500-018-3102-4
27. T. F. Lee, Enhancing the security of password authenticated key agreement protocols based on chaotic maps, *Inf. Sci.*, **290** (2015), 63–71. https://doi.org/10.1016/j.ins.2014.08.041
28. A. T. Azar, S. Vaidyanathan, *Chaos Modeling and Control Systems Design*, Springer, 2015.
29. L. Zhang, X. Xin, B. Liu, Y. Wang, Secure OFDM-PON based on chaos scrambling, *IEEE Photonics Technol. Lett.*, **23** (2011), 998–1000. https://doi.org/10.1109/LPT.2011.2149512
30. L. D. S. Coelho, A. A. R. Coelho, Model-free adaptive control optimization using a chaotic particle swarm approach, *Chaos, Solitons Fractals*, **41** (2009), 2001–2009. https://doi.org/10.1016/j.chaos.2008.08.004
31. Y. Li, M. Han, Q. Guo, Modified whale optimization algorithm based on tent chaotic mapping and its application in structural optimization, *KSCE J. Civil Eng.*, **24** (2020), 3703–3713. https://doi.org/10.1007/s12205-020-0504-5
32. Z. Ma, X. Yuan, S. Han, D. Sun, Y. Ma, Improved chaotic particle swarm optimization algorithm with more symmetric distribution for numerical function optimization, *Symmetry*, **11** (2019), 876. https://doi.org/10.3390/sym11070876
33. Y. Wang, K. W. Wong, X. Liao, G. Chen, A new chaos-based fast image encryption algorithm, *Appl. Soft Comput.*, **11** (2011), 514–522. https://doi.org/10.1016/j.asoc.2009.12.011
34. M. Wang, The diffusive logistic equation with a free boundary and sign-changing coefficient, *J. Differential Equations*, **258** (2015), 1252–1266. https://doi.org/10.1016/j.jde.2014.10.022
35. M. S. Tavazoei, M. Haeri, Comparison of different one-dimensional maps as chaotic search pattern in chaos optimization algorithms, *Appl. Math. Comput.*, **187** (2007), 1076–1085. https://doi.org/10.1016/j.amc.2006.09.087
36. S. Saremi, S. Mirjalili, A. Lewis, How important is a transfer function in discrete heuristic algorithms, *Neural Comput. Appl.*, **26** (2015), 625–640. https://doi.org/10.1007/s00521-014-1743-5
37. Z. Wang, j. Wang, D. Li, Research on WSN optimization coverage of an enhanced sparrow search algorithm, *Chin. J. Sens. Actuators*, **34** (2021), 818–828.
38. D. Whitley, A genetic algorithm tutorial, *Stat. Comput.*, **4** (1994), 65–85. https://doi.org/10.1007/BF00175354
39. K. V. Price, Differential evolution: a fast and simple numerical optimizer, in *Proceedings of North American Fuzzy Information Processing*, (1996), 524–527. https://doi.org/10.1109/NAFIPS.1996.534790
40. Y. Feng, G. G. Wang, J. Dong, L. Wang, Opposition-based learning monarch butterfly optimization with gaussian perturbation for large-scale 0-1 knapsack problem, *Comput. Electr. Eng.*, **67** (2018), 454–468. https://doi.org/10.1016/j.compeleceng.2017.12.014
41. Q. Xu, Z. Wang, Z. Zhen, Information fusion estimation-based path following control of quadrotor uavs subjected to gaussian random disturbance, *ISA Trans.*, **99** (2020), 84–94. https://doi.org/10.1016/j.isatra.2019.10.003
42. S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, Optimization by simulated annealing, *Science*, **220** (1983), 671–680. https://doi.org/10.1126/science.220.4598.671
43. D. Chu, H. Chen, X. Wang, Whale optimization algorithm based on adaptive weight and simulated annealing, *Chin. J. Electr.*, **47** (2019), 992–999. https://doi.org/10.3969/j.issn.0372-2112.2019.05.003

44. L. Jin, H. Tang, B. Lin, X. Zhu, A simulated annealing algorithm for continuous functions and its convergence properties, *Math. Numer. Sin.*, **27** (2005), 19–30. https://doi.org/10.12286/jssx.2005.1.19
45. Z. Liu, H. Lu, J. Ren, A diversity-enhanced hybrid firefly algorithm, *J. Shanxi Univ., Nat. Sci. Ed.*, **44** (2021), 249–256. https://doi.org/10.13451/j.sxu.ns.2020034
46. W. Gao, S. Liu, Z. Xiao, J. Yu, Butterfly optimization algorithm based on Cauchy variation and adaptive weight, *Comput. Eng. Appl.*, **56** (2020), 43–50. https://doi.org/10.3778/j.issn.1002-8331.1907-0048

AIMS Press