A Fusion Crossover Mutation Sparrow Search Algorithm

Yanqiang Tang, Chenghai Li, Song Li, Bo Cao, and Chen Chen

Air and Missile Defense College of Air Force Engineering University, Xi’an 710051, China

Correspondence should be addressed to Yanqiang Tang; yanqiang_tang@163.com and Song Li; li_song77@163.com

Received 7 May 2021; Revised 1 July 2021; Accepted 13 July 2021; Published 23 July 2021

1. Introduction

Swarm intelligence optimization algorithm is a kind of biodynamics algorithm that simulates the behavior of some creatures in nature or is inspired by some physical phenomenon. Its central idea is to balance the global search and local search in a solution space to find the optimal solution. The swarm intelligence algorithm has attracted the attention of many scholars because of its simple operation and high efficiency. At present, these algorithms are widely used in image processing [1], training neural networks [2], signal processing [3], and feature selection [4]. The common swarm intelligence optimization algorithms mainly include particle swarm optimization (PSO) [5], grey wolf optimizer (GWO) [6], whale optimization algorithm (WOA) [7], salp swarm algorithm (SSO) [8], sine cosine algorithm (SCA) [9], and sparrow search algorithm (SSA) [10].

Swarm intelligence optimization algorithms have the problem of easily falling into a local optimum and reducing the diversity of the population in later iterations. In response to this problem, many researchers have proposed methods to introduce various learning techniques into swarm intelligence optimization algorithms. For example, Li et al. [11] conducted a comprehensive study and analysis on how traditional intelligent optimization algorithms combine with learning operators and gave an outlook on its future development direction. To solve the numerical optimization problem, Li and Wang [12] proposed an improved elephant herd optimization using dynamic topology and learning-based biogeography optimization (BLEHO), while ensuring a better evolutionary process for the population through an elite strategy. Meanwhile, the global velocity strategy and the new learning strategy were introduced in EHO to update the velocity and position of individuals [13], and good results were obtained. The cuckoo search algorithm is an effective evolutionary method for global optimization, but it has the drawback of premature convergence problem and poor balance between exploitation and exploration. To address these problems, a balanced learning function was introduced into the cuckoo search algorithm by which a balance between exploitation and exploration was achieved [14]. Li et al. [15] proposed a new extension of CS with Q-learning steps and genetic operators, namely, the dynamic step cuckoo search algorithm (DMQL-CS). The step control strategy was considered as the action, which was used to check the effect of a single multistep evolution and to learn a single optimal step by calculating the Q-function value, and the crossover and variation operations expand the search range of the population and increase the diversity of the population. Li et al. [16] introduced a learning model.
combining individual historical knowledge and group knowledge into the CS algorithm while using a threshold statistical learning strategy to select the optimal learning model to exploit the potential of individual knowledge learning and group knowledge learning, providing a good trade-off between exploration and exploitation. To address the premature convergence suffered by the CS and the poor balance between exploration and exploitation, a dynamic CS with Taguchi opposition-based search (TOB-DCS) was proposed [17]. It employed two new strategies: Taguchi opposition-based search and dynamic evaluation. The Taguchi search strategy provided random generalized learning based on opposing relationships to enhance the exploration ability of the algorithm. The dynamic evaluation strategy reduced the number of function evaluations and accelerated the convergence property. Wang et al. [18] and Feng et al. [19] introduced opposition-based learning into the krill herd algorithm and monarch butterfly optimization, respectively, providing us with a strategy to optimize the population of individuals. Wang et al. [20] introduced chaos theory into CS to further improve the optimization performance of CS. Lu et al. [21] studied three chaotic strategies with 11 different chaotic map functions on GWO and successfully applied the most suitable chaotic strategy as the chaotic GWO to real-world engineering problems. Inspired by the SCA algorithm, Nabil et al. [22] updated the positions of the followers in the SSO algorithm through a sine function, which helped to improve the exploration stage and avoid stagnation in local areas. Liu et al. [23] introduced chaos strategy into SSA and used adaptive inertia weight to balance the convergence speed and exploration ability of the algorithm, but between the speed of convergence and the ability of exploration, a certain part must be sacrificed. Yuan et al. [24] used gravity center reverse learning mechanism to initialize the population so that the population has better spatial solution distribution. Secondly, the learning coefficient was introduced into the location updating part of the discoverer to improve the global search ability of SSA. Lei et al. [25] introduced the Levy flight strategy into SSA, which improved the global optimization ability of SSA, but increased the complexity. Zhang et al. [26] used the sine cosine algorithm as a hybrid algorithm to help SSA jump out of local optimum. Although all the above various improvements contribute to optimization performance, there are still certain shortcomings, and thus optimization performance can be further improved.

In this paper, on the basis of predecessors’ ideas, the flight behavior of the bird swarm algorithm and the cross mutation idea of the genetic algorithm are introduced into SSA, and the improved tent chaos is used to assist. The experimental results show that the ISSA proposed in this paper is effective and feasible in convergence accuracy, stability, convergence speed, and comprehensive performance.

The organizational structure of this paper is as follows. Section 1 introduces SSA and its improved methods. Section 2 uses test function analysis to verify the effectiveness of the improved algorithm. Section 3 gives the conclusion and describes the future of the proposed algorithm.

2. Sparrow Search Algorithm and Its Improvement

2.1. Sparrow Search Algorithm. Xue and Shen [10] proposed the sparrow search algorithm method. The idea of this algorithm comes from the foraging and antipredation behaviors of sparrows, which can be abstracted as an explorer-follower-forwarder model. The explorer has a high energy reserve and high fitness value. It provides a foraging area and direction for the followers. The followers follow the explorer with the best fitness value to find food to increase their own energy reserves and fitness values. Some followers may also constantly monitor the explorer to compete for food. A forewarner will sound an alarm when it is aware of the danger, and at the same time, moves quickly to a safe area to get a better position. Sparrows in the middle of the flock will randomly walk close to other sparrows, which is an anti-predation behavior. At the same time, if the alarm value is greater than the safety threshold, the explorer needs to take all the followers away from the dangerous area.

In SSA, assuming that there are \( N \) sparrows in a \( D \)-dimensional search space, the position of each sparrow is as follows:

\[
X = \begin{bmatrix}
X_{1,1} & X_{1,2} & \cdots & X_{1,d} & \cdots & X_{1,D} \\
X_{2,1} & X_{2,2} & \cdots & X_{2,d} & \cdots & X_{2,D} \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
X_{N,1} & X_{N,2} & \cdots & X_{N,d} & \cdots & X_{N,D}
\end{bmatrix},
\]

(1)

where \( i = 1, 2, \ldots, N, d = 1, 2, \ldots, D \), and \( x_{id} \) represents the position of the \( ith \) sparrow in the \( dth \) dimension.

As the explorer guides the movement of the entire sparrow group, search for food can occur anywhere. Therefore, the location of the explorer is updated as follows:

\[
x_{i,d}^{t+1} = \begin{cases} 
X_{i,d}^{t} \cdot \exp\left(\frac{-i}{\alpha \cdot \text{iter}_{\text{max}}}\right), & R_{2} < \text{ST}, \\
x_{i,d}^{t} + Q \times L, & R_{2} \geq \text{ST}, 
\end{cases}
\]

(2)

where \( t \) represents the current iteration, \( \text{iter}_{\text{max}} \) is the maximum iteration, \( \alpha \) is a random number in \((0,1]\), \( \text{iter}_{\text{max}} \) is a random number following a normal distribution, and \( L \) represents a \( 1 \times d \) matrix, where each element is 1. \( R_{2} \in [0,1] \) represents an alarm value, and \( \text{ST} \in [0.5, 1] \) represents the safety threshold. If \( R_{2} < \text{ST} \), there are no predators around and the explorer will enter the wide search mode. If \( R_{2} \geq \text{ST} \), some sparrows have found natural predators, and all sparrows need to fly to another safe area quickly.
The followers follow the explorer to find food and may compete with the explorer for food to increase their food intake. The position update equation is as follows:

\[
x_{id}^{t+1} = \begin{cases} 
Q \cdot \exp \left( \frac{x_{worst}^{t+1} - x_{id}^{t+1}}{r} \right), & i > \frac{n}{2} \\
\frac{1}{D} \sum_{d=1}^{D} \left( \left| x_{id}^{t+1} - x_{best}^{t+1} \right| \cdot \text{rand}[-1, 1] \right), & \text{otherwise}
\end{cases}
\]

(3)

where \(x_{worst}^{t+1}\) represents the global worst position of the sparrows in the \(d\) dimension at the \(t+1\)th iteration and \(x_{best}^{t+1}\) represents the best position of the explorer in the \(d\) dimension at the \((t+1)\)th iteration. When \(i > (n/2)\), the \(i\)th follower with poor fitness is most likely to starve to death; otherwise, the \(i\)th follower randomly finds a location for food near the best position of the explorer.

Assuming that the forewarner sparrows account for about 10% to 20% of the sparrow population, and their initial positions are randomly determined, the mathematical model can be expressed as follows:

\[
x_{id}^{t+1} = \begin{cases} 
x_{best}^{t+1} + \beta \cdot |x_{id}^{t} - x_{best}^{t}|, & f_i \neq f_g, \\
x_{id}^{t} + K \cdot \left( \frac{x_{id}^{t} - x_{worst}^{t}}{f_i - f_{worst}} + \varepsilon \right), & f_i = f_g,
\end{cases}
\]

(4)

where \(\beta\) is a step size control parameter, which is a random normal distribution with the mean of 0 and variance of 1; \(K\) is a random number between \([-1, 1]\); \(f_i, f_g, f_{worst}\), respectively, represent the current fitness values of the sparrow, the current global best fitness value, and the current global worst fitness value; \(\varepsilon\) is the smallest constant to avoid zero-division-error. \(f_i \neq f_g\) represents the sparrow at the edge of the group. \(f_i = f_g\) indicates that the sparrow in the middle of the group is aware of the danger and needs to move to another place.

2.2. Initial Population. The quality and diversity of the initial population in the swarm intelligence optimization algorithm play a great role in the optimization performance of the algorithm. A high-quality initial population can increase the convergence speed of the algorithm. Therefore, the initialization of the sparrow population has a great effect on the search accuracy of the SSA algorithm. In order to maintain the diversity of the population, this paper adopts the fusion of an improved tent chaotic map and elite opposition-based learning strategy to initialize the population and replace the method of randomly generating the population in the SSA algorithm. We first use the improved tent chaotic map to make the initial population distribution more uniform, and then we use the feature that elite individuals contain more effective information in the elite opposition-based learning strategy to construct superior individuals and further improve the quality of the population.

2.2.1. Improved Tent Chaotic Sequence. Tent chaos, as a type of chaos, is a more complicated nonlinear dynamic behavior than logistic chaos. The study proposed by Shan et al. [27] has shown that the tent chaotic map has better traversal uniformity and faster search speed than logistic chaotic. Using the randomness, ergodicity, and regularity of the tent chaotic sequence to optimize the SSA algorithm can effectively maintain the diversity of the algorithm population, thereby improving the global search capability.

The expression of the tent chaotic map is as follows:

\[
y_{i+1} = \begin{cases} 
2y_i, & 0 \leq y \leq \frac{1}{2} \\
2(1 - y_i), & \frac{1}{2} < y \leq 1.
\end{cases}
\]

(5)

Analysis of equation (5) shows that the tent chaotic sequence is not perfect, and there are points with small periods and those with unstable periods. To avoid falling into the points in the iterative process, a random variable, \(\text{rand}(0,1) \times (1/N)\), is introduced into the tent chaotic expression. The improved tent chaotic map expression is as follows:

\[
y_{i+1} = \begin{cases} 
2y_i + \text{rand}(0,1) \times \frac{1}{N}, & 0 \leq y \leq \frac{1}{2} \\
2(1 - y_i) + \text{rand}(0,1) \times \frac{1}{N}, & \frac{1}{2} < y \leq 1.
\end{cases}
\]

(6)

The improved tent chaotic map after Bernoulli shift transformation is expressed as follows:

\[
y_{i+1} = (2y_i) \text{mod} 1 + \text{rand}(0,1) \times \frac{1}{N}.
\]

(7)

According to the characteristics of the tent chaotic map, the method for initializing the sparrow population in the feasible region is as follows:

\[
x_{id} = \text{lb}_{id} + q(\text{ub}_{id} - \text{lb}_{id}) \times y_{id},
\]

(8)

where \(q\) is the uniform distribution in \([0, 1]\), and \(\text{lb}_{id}\) and \(\text{ub}_{id}\) represent the upper and lower bounds of the feasible solution, respectively.

2.2.2. Elite Opposition-Based Learning Strategy. Opposition-based learning (OBL) is a new strategy in computational intelligence. A great number of studies have shown that the opposition-based solution in the opposition-based learning strategy is closer to the global optimal solution than the existing solutions. Therefore, using this strategy can effectively enhance the diversity of the population, and to a certain extent, prevent the algorithm from falling into a local optimum. The elite opposition-based learning strategy is based on the reverse learning strategy in
order to overcome the problem in the reverse learning strategy that the current solution and the resulting reverse solution are not necessarily easier than the current search space in search for the global optimal solution. The strategy selects the elite individuals in the current solution to construct the reverse solution, thus avoiding the invalid reverse solutions generated by nonelite individuals. Assuming that \( s_i = (s_{i1}, s_{i2}, s_{i3}, \ldots, s_{iN}) \) \( (i = 1, 2, 3, \ldots, N) \) is an elite individual in the \( D \)-dimensional search space, then its opposition-based solution, \( \bar{s}_i = (\bar{s}_{i1}, \bar{s}_{i2}, \bar{s}_{i3}, \ldots, \bar{s}_{iD}) \), is defined as follows:

\[
\bar{s}_i = \delta(\text{lb}_j + \text{ub}_j) - s_{ij},
\]  

(9)

where \( s_{ij} \) is the \( j \)-dimension vector of the elite solution \( s_i \), \( j = 1, 2, \ldots, D \), \( \delta \) is the random value in \([0, 1]\), and \( \text{lb}_j = \min_j(s_{ij}) \) and \( \text{ub}_j = \max_j(s_{ij}) \) are the lower and upper bounds of the \( j \)-th dimension search space, respectively. After the opposition-based solution \( \bar{s}_i \) is obtained, it needs to be treated across the bounds:

\[
s_{ij} = \text{rand}(\text{lb}_j, \text{ub}_j), \quad \text{if } s_{ij} < \text{lb}_j \text{ or } s_{ij} > \text{ub}_j,
\]  

(10)

where \( \text{rand}(\text{lb}_j, \text{ub}_j) \) is a random value in \([\text{lb}_j, \text{ub}_j] \).

2.2.3. Population Initialization. According to the two methods described in Sections 2.2.1 and 2.2.2, the pseudocode of the population initialization of the SSA algorithm in this paper is shown in Algorithm 1.

2.3. Birds Swarm Algorithm Strategy. The bird swarm algorithm (BSA) was proposed by Meng et al. [28] in 2015 based on the flight, foraging, and vigilance behaviors of birds. Similar to SSA, the BSA also has explorers and followers. The equations for updating their locations are as follows:

\[
x_{ij}^{t+1} = x_{ij}^t + x_{ij}^t \times \text{rand}(0, 1),
\]

\[
x_{ij}^{t+1} = x_{ij}^t + (x_{k}^t - x_{ij}^t) \times \text{FL} \times \text{rand}(0, 1),
\]  

(11)

where \( \text{rand}(0, 1) \) represents a Gaussian random distribution with a mean of 0 and a standard deviation of 1; \( k \in \{1, 2, 3, \ldots, N\} \), \( k \neq i \), FL is \([0, 2]\) represents the probability that a follower follows the explorer to find food.

In SSA, when \( R_2 < \text{ST} \), the number of the individual explorer sparrows is decreasing in every dimension, while the individual explorer sparrows in the BSA overcome this defect, as shown in Figure 1. Therefore, with the help of the idea of the explorer in BSA, the equation for the position of the explorer in SSA is modified as follows:

\[
x_{i,d}^{t+1} = \begin{cases} 
    x_{i,d}^t + x_{i,d}^t \times \text{rand}(0, 1), & R_2 < \text{ST}, \\
    x_{i,d}^t + Q, & R_2 \geq \text{ST}.
\end{cases}
\]  

(12)

At the same time, in SSA, the follower approaching the best positions in all dimensions can lead to rapid convergence but also reduces the diversity of the population, thereby making it easy for the algorithm to fall into a local optimum. In the BSA, the follower approaches the explorer with a certain probability. While ensuring global convergence and the diversity of the population, the local optimum can be avoided effectively. Therefore, the idea of the follower in BSA is introduced into SSA, and the location of the improved SSA follower is updated as follows:

\[
x_{i,d}^{t+1} = \begin{cases} 
    Q \cdot \exp \left( \frac{x_{\text{worst},d}^t - x_{i,d}^t}{Q} \right), & i > \frac{n}{2} \\
    x_{ij}^t + (x_{k}^t - x_{ij}^t) \times \text{FL} \times \text{rand}(0, 1), & \text{otherwise}.
\end{cases}
\]  

(13)

2.4. Cauchy Mutation. The Cauchy mutation comes from the Cauchy distribution of the continuous probability distribution. The probability density of the one-dimension Cauchy distribution is expressed by the following:

\[
f(x) = \frac{1}{\pi \cdot \frac{a}{x^2 + a^2}}, \quad x \in (-\infty, +\infty).
\]  

(14)

When \( a = 1 \), as the distribution is the standard Cauchy distribution. Figure 2 shows the curves of the probability density functions of the Cauchy distribution and Gaussian distribution. It can be seen from the figure that the main feature of the Cauchy distribution is that, compared to the Gaussian distribution, its peak value at zero is smaller, its descent from the peak to the zero value is less steep and its descent is slower. Therefore, the Cauchy mutation is stronger than Gaussian mutation in perturbation ability, and the range of mutation is more uniform. Introducing the Cauchy mutation into the SSA algorithm can fully take advantage of the perturbation ability of the Cauchy operator and improve the algorithm’s global optimization ability.

The Cauchy mutation equation is as follows:

\[
\text{mutation} (x) = x \left(1 + \tan \left(\pi (u - 0.5)\right)\right),
\]  

(15)

where \( x \) is the original individual position, \( \text{mutation} (x) \) is the individual position after Cauchy mutation, and \( u \) is a random number in \((0, 1)\).

2.5. Chaotic Perturbation. The purpose of introducing chaotic perturbation in the algorithm is to prevent it from falling into a local optimum and to improve the global search ability and optimization accuracy. The steps are as follows:

(1) Generate chaotic variables \( Y_d \) through equation (7).

(2) Map the chaotic variables to the solution space of the problem to be solved

\[
\text{new} \ X_d = \min_d + (\max_d - \min_d) \cdot Y_d,
\]  

(16)

where \( \min_d \) and \( \max_d \) are the minimum and maximum values of the \( d \)-th dimension variable \( \text{new} X_d \).

(3) Perform chaotic perturbation on the individual according to the following equation:
where $X'$ is the individual who needs chaotic perturbance, new $X$ is the amount of chaotic perturbance generated, and new $X'_d$ is the individual after the chaotic perturbance.

2.6. Introduction of Genetic Algorithm. The genetic algorithm is introduced into the SSA algorithm [29]. Due to its position update equations, the idea of SSA algorithm is position transfer, that is to improve the population quality by sharing useful information among sparrow individuals and self-learning between individuals. After the introduction of the genetic algorithm, the improved individual obtains a better next-generation population through crossover mutation. Moreover, the SSA algorithm now not only has the powerful global search ability of the genetic algorithm, but also integrates the position transfer idea of the SSA algorithm. It makes full use of the information of the population and the individuals that are ignored by the genetic algorithm, and its optimization performance is better.

At the same time, the selection of crossover probability $p_c$ and mutation probability $p_m$ in the genetic algorithm is one of the important factors affecting the optimization ability of the algorithm. If $p_c$ is too small, the generation speed of new individuals during iteration will slow down, leading to early termination of the calculation process. If $p_c$ is too large, there are too many newly generated individuals in the group, which can damage the excellent individuals that

\[ \text{new } X'_d = \frac{X' + \text{new } X}{2}, \]  

(17)
have been generated in the group. If \( p_m \) is too small, the ability to generate new individuals by mutation operations will be weak, and many excellent genes will be lost prematurely and not enter the next generation, which is not conducive to maintaining the diversity of the population. Lastly, if \( p_m \) is too large, it will be similar to a random search algorithm.

Therefore, this paper improves the crossover rate and the mutation rate and proposes an adaptive crossover and an adaptive mutation strategy based on the golden ratio index function to make the good individuals at a certain moment of evolution also change. The improved change equations are shown below:

\[
P_c = \begin{cases} 
  p_{c1} \exp(-0.618) \frac{(f' - f_{avg})}{f_{max} - f_{avg}}, & f' \geq f_{avg} \\
p_{c1}, & f' \leq f_{avg} 
\end{cases}
\]

\[
P_m = \begin{cases} 
  p_{m1} \exp(-0.382) \frac{(f_{max} - f)}{f_{max} - f_{avg}}, & f' \geq f_{avg} \\
p_{m1}, & f' \leq f_{avg} 
\end{cases}
\]

where \( p_{c1} \) is the maximum crossover probability set by the algorithm, and its value is set to 0.7; \( p_{m1} \) is the maximum mutation probability, and its value is set to 0.01. Meanwhile, the golden ratio is introduced into the equations, and the optimal solutions are approached one by one according to the principle of equal ratio, symmetric contraction, and optimal selection; this improves the speed of the global optimal solution search.

A flow chart of the genetic algorithm introduced into SSA is shown in Figure 3.

In Figure 3, the individual sparrow in SSA is regarded as a chromosome in GA. In the \( N \)th generation group, individual sparrows enter the \((N + 1)\)-th generation after being improved, crossed over, and mutated. The steps to introduce GA into SSA are as follows:

1. Improve the selection operator: in each generation, first calculate the fitness value of each individual sparrow, sort by fitness value, and choose the top half of the best individuals as excellent samples to be improved through the SSA algorithm (these samples thus enter the next generation).

2. Crossover mutation operator: in order to obtain the remaining \( p/2 \) next-generation individuals, select outstanding individuals from the SSA population as fathers and mothers, and generate new offspring individuals for the next generation through dynamic crossover operators and mutation operators.

### 2.7. Improved Sparrow Search Algorithm

The pseudocode of the improved sparrow search algorithm is expressed in detail in Algorithm 2.

### 3. Experimental Results and Analysis

In order to verify the optimization performance of the ISSA algorithm, the particle swarm algorithm, grey wolf optimization algorithm, whale optimization algorithm, sine cosine optimization algorithm, salvia swarm algorithm, basic sparrow search algorithm, and ISSA were used in simulation experiments on 15 benchmark functions. The convergence accuracy, stability, convergence speed, improved advantages, and comprehensive performance of the ISSA algorithm were analyzed. The simulation experiment in this paper was performed on the Windows 10 64-bit operating system with Intel Core i7 CPU, 2.60 GHz, 16 GB, and MATLAB R2016b. The parameters of the algorithms are shown in Table 1.

#### 3.1. Comparative Experiment with Benchmark Functions

The benchmark functions are shown in Table 2. \( f_1 \text{--} f_6 \) are unimodal functions, \( f_7 \text{--} f_{10} \) are multimodal functions, and \( f_{11} \text{--} f_{15} \) are fixed-dimension functions. In order to make the experimental results fair and objective, the number of individuals in each algorithm is set to 30, and the maximum number of iterations is 500. Fifty independent simulation experiments were performed on each benchmark function for each algorithm, and the average value, optimal value, and standard deviation obtained from 50 experiments were calculated; the results are shown in Table 3.

For the unimodal functions \( f_1 \text{--} f_4 \), ISSA can accurately find the optimal value of zero, and the average value and standard deviation are also zero. Although SSA can find the optimal value of zero in function \( f_1 \) and function \( f_2 \), the average values and standard deviations are not zero. The optimization performance of the other five algorithms can only approach zero infinitesimally. Among them, WOA performs better in \( f_1 \) and \( f_2 \), with its optimization performance several orders of magnitude higher than those of the other four algorithms. In the unimodal functions \( f_5 \) and \( f_6 \), the optimization performance of ISSA is only one or two orders of magnitude higher than that of SSA; meanwhile, the optimization performance of the remaining five algorithms is not much different from that of the SSA algorithm. For the multimodal functions \( f_7 \text{--} f_{10} \), ISSA, SSA, and WOA can all find the optimal value zero in \( f_7 \), and the average value and the standard deviation are all zero. In \( f_8 \), ISSA and SSA stop searching after finding a point that is infinitesimally close to the optimal value and have the same stability; WOA is slightly inferior and is not as stable as ISSA and SSA. In \( f_9 \), ISSA, SSA, GWO, and WOA can also find the optimal value zero, but GWO and WOA are less stable. In \( f_{10} \), the performance of the ISSA algorithm is not improved much, with the optimal value being only three orders of magnitude higher than that of SSA; moreover, ISSA and SSA have stabilities in the same order of magnitude. In the fixed-dimension functions, seven algorithms are able to find the optimal value in both \( f_{11} \) and \( f_{13} \). The standard deviation of SSO in \( f_{11} \) is the lowest, and the standard deviation of PSO in \( f_{13} \) is the lowest, indicating they are the most stable. In \( f_{12} \), only ISSA and SSA are able to find the optimal value. The
Initialize the parameters of SSA, for instance, the initial values of the upper and lower bound $lb$ and $ub$ and the maximum number of iterations $T$;

Initialize the sparrow population $X_i$ ($i = 1, 2, \ldots, N$) by Algorithm 1;

Calculate the fitness value $f_i$ of the individual sparrow;

Obtain the current optimal position, the worst position, and the corresponding optimal fitness value and worst fitness value;

While ($t < T$)

For $i = 1$ to $N$

Update the positions of the explorers using equation (12)

Update the positions of the followers using equation (13)

Update the positions of the forewarners using equation (4)

End for

For $i = 1$ to $N$

Update $f_i$ and calculate the average fitness value $f_{avg}$ of the sparrow population;

if $f_i < f_{avg}$

Carry out Cauchy mutation according to equation (15)

else

Perform tent chaotic perturbation according to equation (17)

end if

Update $f_i$ and sort the individuals by $f_i$;

Retain the best first half of the individuals to promoted by SSA and directly enter the next generation;

Select parents and offspring from the individuals that have been promoted to crossover and mutation using equations (18) and (19);

Update the current optimal position, the worst position, and the corresponding optimal fitness value and worst fitness value;

End for

$t = t + 1$;

End while

Return $X$

Algorithm 2: Pseudocode of the ISSA.
### Table 1: Parameter setting of each algorithm.

| Algorithm | Parameter |
|-----------|-----------|
| PSO       | $C_1 = C_2 = 2$, $w_{\text{min}} = 0.2$, $w_{\text{max}} = 0.9$ |
| GWO       | $a = [2, 0]$ |
| WOA       | $a_1 = [2, -1]$; $b = 1$ |
| SCA       | $A = 2$ |
| SSO       | $c_1 = 2 \times \exp((-4 \times x/L); c_2 = \text{rand}(0, 1)$ |
| SSA       | $P_{\text{percent}} = 0.2$; $r_2 = \text{rand}(0, 1)$ |

### Table 2: Benchmark function.

| Function   | Formula                                                                 | Dim  | Domain          | Best     |
|------------|-------------------------------------------------------------------------|------|-----------------|----------|
| Sphere     | $f_1(x) = \sum_{i=1}^{n} x_i^2$                                         | 30   | $[-100, 100]$   | 0        |
| Schwefel's | $f_2(x) = \sum_{i=1}^{n} |x_i| + \prod_{i=1}^{n} |x_i|$                                              | 30   | $[-10, 10]$     | 0        |
| Quadric    | $f_3(x) = \sum_{i=1}^{n} |x_i|^2$                                                               | 30   | $[-100, 100]$   | 0        |
| Rosenbrock | $f_4(x) = \sum_{i=1}^{n-1} [100 (x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$ | 30   | $[-30, 30]$     | 0        |
| Step       | $f_5(x) = \sum_{i=1}^{n} (|x_i + 0.5|)^2$                                | 30   | $[-100, 100]$   | 0        |
| Quartic    | $f_6(x) = \sum_{i=1}^{n} x_i^4 + \text{random}[0, 1]$                | 30   | $[-1.28, 1.28]$ | 0        |
| Rastrigrin  | $f_7(x) = \sum_{i=1}^{n} x_i^2 - 10 \cos(2 \pi x_i) + 10$             | 30   | $[-5.12, 5.12]$ | 0        |
| Ackley     | $f_8(x) = -20 \exp(-0.2 \sqrt{(1/n) \sum_{i=1}^{n} x_i^2}) - \exp((1/n) \sum_{i=1}^{n} \cos(2 \pi x_i)) + 20 + e$ | 30   | $[-32, 32]$     | 0        |
| Griewong   | $f_9(x) = (1/4000) \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos(x_i/\sqrt{10}) + 1$ | 30   | $[-600, 600]$   | 0        |
| Generalized Penalized | $f_{10}(x) = (\pi/n) \left[ 10 \sin^2(\pi x_{j_1}) + \sum_{i=1}^{n} (y_{i-1} - 1)^2 \left[ 1 + 10 \sin^2(\pi y_{i-1}) \right] + (y_n - 1)^2 \right]$ | 30   | $[-50, 50]$     | 0        |
| Foxholes   | $f_{11}(x) = ((1/1500) + \sum_{i,j} \left( \frac{1}{(j + \sum_{i} (x_i - a_i)^2}) \right)^4$ | 2    | $[-65, 65]$     | 1        |
| Kowalik    | $f_{12}(x) = \sum_{i=1}^{n} (|a_i - (x_i b_1^2 + b_2 x_i) - (b_2^2 + b_3 x_i + x_i^2))|^2$ | 4    | $[-5, 5]$       | 0.000307 |
| Six-Hump Camel | $f_{13}(x) = 4x_1^2 - 2.1x_1^4 + (1/3)x_1^6 + x_2 - 4x_2^2 - 2x_2^4$ | 2    | $[-5, 5]$       | -1.0316  |
| Hartmann 6-D | $f_{14}(x) = -\sum_{i=1}^{n} (\exp[-\sum_{j=1}^{n} a_j (x_j - p_j)^2]$ | 6    | $[0, 1]$        | -3.32237 |
| Schel     | $f_{15}(x) = -\sum_{i=1}^{n} |x - a_i (x - a_i)^2 + c_i|^2$          | 4    | $[0, 10]$       | -10.5363 |

### Table 3: Comparison of benchmark function results.

| Algorithm | $f_1$ Mean | $f_1$ Std | $f_2$ Mean | $f_2$ Std |
|-----------|------------|----------|------------|----------|
| PSO       | 4.1424E-06 | 0.0236E-01 | 2.7700E-02 | 0.0002E-00 |
| GWO       | 2.3907E-29 | 0.2413E-27 | 8.3437E-17 | 0.0000E-00 |
| WOA       | 5.0425E-09 | 0.2760E-07 | 5.9323E-10 | 0.0000E-00 |
| SCA       | 9.0000E-03 | 0.0171E-01 | 3.0800E-02 | 0.0000E-00 |
| SSO       | 2.7431E-08 | 0.0000E-00 | 2.1975E-02 | 0.0000E-00 |
| SSA       | 0.0000E+00 | 0.0000E+00 | 2.8881E-28 | 0.0000E-00 |
| ISSA      | 0.0000E+00 | 0.0000E+00 | 1.9201E-27 | 0.0000E-00 |

| $f_3$ Mean | $f_3$ Std | $f_4$ Mean | $f_4$ Std |
|------------|----------|------------|----------|
| PSO        | 2.6534E+01 | 0.0000E+00 | 9.6482E+01 | 0.0000E+00 |
| GWO        | 6.5094E-09 | 0.0000E-00 | 6.2987E-01 | 0.0000E-00 |
| WOA        | 1.3812E+04 | 0.0000E+00 | 2.1975E-02 | 0.0000E-00 |
| SCA        | 1.4069E+03 | 0.0000E+00 | 2.1975E-02 | 0.0000E-00 |
| SSO        | 3.8572E+02 | 0.0000E+00 | 2.1975E-02 | 0.0000E-00 |
| SSA        | 3.7614E-26 | 0.0000E+00 | 2.1975E-02 | 0.0000E-00 |
| ISSA       | 0.0000E+00 | 0.0000E+00 | 2.1975E-02 | 0.0000E-00 |

| $f_5$ Mean | $f_5$ Std | $f_6$ Mean | $f_6$ Std |
|------------|----------|------------|----------|
| PSO        | 7.7336E-06 | 0.0000E+00 | 6.2400E-02 | 0.0000E-00 |
| GWO        | 4.3825E-05 | 0.0000E+00 | 4.9189E-04 | 0.0000E-00 |
| WOA        | 5.7600E-02 | 0.0000E+00 | 4.8200E-02 | 0.0000E-00 |
| SCA        | 3.4801E+00 | 0.0000E+00 | 7.8900E-02 | 0.0000E-00 |
| SSO        | 1.8335E-08 | 0.0000E+00 | 1.7420E-01 | 0.0000E-00 |
| SSA        | 3.2830E-14 | 0.0000E+00 | 1.2378E-06 | 0.0000E-00 |
| ISSA       | 4.2487E-16 | 0.0000E+00 | 7.3944E-04 | 0.0000E-00 |

| $f_7$ Mean | $f_7$ Std | $f_8$ Mean | $f_8$ Std |
|------------|----------|------------|----------|
| PSO        | 3.0135E+01 | 0.0000E+00 | 5.6300E-01 | 0.0000E-00 |
| GWO        | 5.6843E-14 | 0.0000E+00 | 1.8386E-14 | 0.0000E-00 |
| WOA        | 0.0000E+00 | 0.0000E+00 | 4.6541E-15 | 0.0000E-00 |
stability of ISSA is three orders of magnitude higher than that of SSA, and the remaining algorithms are infinitesimally close to the optimal value. In f14, only SCA and WOA fail to find the optimal value, and the optimization capabilities of the other algorithms are not much different. In f15, only SCA fails to find the optimal value, whereas ISSA has the best stability, which is 14 orders of magnitude higher than the stability of the other algorithms. Therefore, the performance of ISSA is greatly improved in the unimodal functions and the multimodal functions. All the algorithms perform well for the fixed-dimension functions. It is relatively easy to find their optimal values. ISSA’s performance improvement is weak in these functions.

Figure 4 shows the convergence curves of different algorithms on the benchmark functions. It can be seen that, for the unimodal functions f1, f2, and f3 and multimodal functions f7 and f9, the ISSA algorithm makes these functions find the optimal value before reaching the maximum number of iterations. From the overall performance of all functions, the convergence speed of ISSA is faster than that of other algorithms. For the fixed-dimension functions f12, f13, f14, and f15, each algorithm is able to find the optimal value or comes infinitesimally close to the optimal value. Although the convergence speed of the ISSA algorithm has not improved much, its convergence speed can still reflect its advantages.

3.2. Rank-Sum Test. Derrac et al. [30] suggest that a statistical test should be carried out to evaluate the performance of improved algorithms. It is inadequate to evaluate an algorithm only based on the average and standard deviation. A statistical test is needed to demonstrate that the proposed improved algorithm has significant improvements over existing algorithms. Each result is compared independently to represent the stability and fairness of the algorithm. In this paper, the Wilcoxon rank-sum test is used to determine whether each result of ISSA is statistically significantly
Figure 4: Continued.
different from the best results of the other five algorithms at $p$ less than 0.05. Table 4 shows the $p$ values obtained by the rank-sum tests for ISSA and the other five algorithms on the 15 benchmark functions. When two algorithms under comparison reach the optimal values, they cannot be compared; thus, NaN in the table means “not applicable;” that is, a significance assessment cannot be made. $R$ is the result of the significance assessment: “+,” “−,” and “=,” respectively, represent ISSA’s performance as being superior, inferior, and equivalent to the algorithms under comparison.

Table 4 shows that only the $p$ values of ISSA and WOA in $f_9$, ISSA and PSO in $f_4$, and ISSA and SSA in $f_2$ and $f_3$ are slightly greater than 0.05; the other $p$-values are much less than 0.05. This indicates that the superior performance of ISSA is statistically significant. In the comparison between ISSA and SSA, the $R$ in the $f_2$ and $f_3$ functions is “−.” This is because the optimization performance of SSA itself is better. Both ISSA and SSA can find the optimal values, but their average values are different. Although the optimization performance of ISSA on fixed-dimension functions has been improved somewhat, there was not much room for improvement to begin with.

In order to evaluate the performance of the algorithms in many aspects, the mean absolute error (MAE) is used as an evaluation index. The MAE ranking shown in Table 5 corresponds to the algorithms listed in Table 2.

As can be seen, ISSA ranks first. Compared with the other six algorithms, ISSA has the smallest MAE, which further proves the effectiveness of ISSA.

3.3. Box Plot. In Section 2.1, each algorithm is used to perform 50 experiments on each of the 15 benchmark functions. The results are compared in Table 3. However, it is difficult to observe the various data and the relationship between them from Table 3. To verify the stability and convergence of ISSA, we also conduct a box plot analysis [31]. The box plot summarizes six types of data: the maximum, upper quartile, median, lower quartile, minimum, and outliers. Figure 5 shows the plots for the 15 benchmark functions in the experiment. In the figure, “+” in each box subgraph represents the outlier, “−” represents the median, the two ends of the rectangular box are the upper and lower quartiles, and “=” represents the maximum or minimum. The difference across the seven algorithms is large. In order
Table 4: p value for Wilcoxon’s rank-sum test.

| Function | PSO | GWO | WOA | SCA | SSO | SSA |
|----------|-----|-----|-----|-----|-----|-----|
| f1       | 3.3111E−20 + | 3.3111E−20 + | 3.3111E−20 + | 3.3111E−20 + | 3.3111E−20 + | 1.2593E−19 + |
| f2       | 3.3111E−20 + | 3.3111E−20 + | 3.3111E−20 + | 3.3111E−20 + | 3.3111E−20 + | 1.2593E−19 + |
| f3       | 3.3111E−20 + | 3.3111E−20 + | 3.3111E−20 + | 3.3111E−20 + | 3.3111E−20 + | 1.2593E−19 + |
| f4       | 4.7330E−20 + | 4.7330E−20 + | 4.7330E−20 + | 4.7330E−20 + | 4.7330E−20 + | 1.332E−18 + |
| f5       | 7.0661E−18 + | 7.0661E−18 + | 7.0661E−18 + | 7.0661E−18 + | 7.0661E−18 + | 2.2000E−03 + |
| f6       | 7.0661E−18 + | 4.0077E−09 + | 1.6684E−05 + | 7.0661E−18 + | 7.0661E−18 + | 2.1000E−03 + |
| f7       | 3.3111E−20 + | 3.0516E−20 + | NaN = 3.3111E−20 + | 3.3111E−20 + | NaN = 3.3111E−20 + |
| f8       | 3.3111E−20 + | 9.5336E−04 + | 8.2200E−02 − | 3.3111E−20 + | 3.3111E−20 + | NaN = 3.3111E−20 + |
| f9       | 7.0661E−18 + | 7.0661E−18 + | 7.0661E−18 + | 7.0661E−18 + | 7.0661E−18 + | 1.8692E−06 + |
| f10      | 7.6000E−03 + | 2.1353E−04 + | 6.1731E−05 + | 4.9524E−07 + | 9.3163E−04 + |
| f11      | 7.6000E−03 + | 9.5403E−18 + | 7.5041E−18 + | 7.0661E−18 + | 8.0500E−02 − |
| f12      | 1.5698E−18 + | 1.5698E−18 + | 1.5698E−18 + | 1.5698E−18 + | 1.5698E−18 + | 1.2590E−01 − |
| f13      | 8.5920E−01 − | 6.3501E−13 − | 2.1497E−12 − | 1.8868E−18 + | 3.7419E−16 + | 1.2000E−03 + |
| f14      | 3.9541E−04 + | 5.3043E−19 + | 5.3043E−19 + | 5.3043E−19 + | 5.3043E−19 + | 1.1848E−04 + |

Table 5: Algorithm MAE ranking.

| Algorithm | MAE  | Rank |
|-----------|------|------|
| ISSA      | 0.5228 | 1    |
| SSA       | 1.9799 | 2    |
| GWO       | 2.6226 | 3    |
| PSO       | 17.8021 | 4    |
| SSO       | 124.377 | 5    |
| WOA       | 2.8926 +E + 03 | 6    |
| SSA       | 8.0045 +E + 03 | 7    |
Figure 5: Continued.
Figure 5: Continued.
to make this box plot comparison more obvious, three subgraphs are generated for each benchmark function, which are the comparison box plots of the seven algorithms, the comparison box plots of the remaining six algorithms after the SCA algorithm is removed, and the comparison box plots of the ISSA and SSA algorithms.

It can be seen from Figure 5 that the median of ISSA is always close to the optimal value of each function, and the variation in optimal values across 50 iterations is the smallest. Furthermore, the distribution of the convergence value is more concentrated than in the other six algorithms, indicating that the ISSA algorithm has stronger robustness. Comparing ISSA and SSA separately, SSA and ISSA can find the optimal value in each iteration of functions $f_7$, $f_8$, $f_9$, and $f_{13}$, so the performance of ISSA cannot be improved anymore. Additionally, except for $f_{11}$ and $f_{14}$, in each benchmark function SSA has more outliers than ISSA, indicating that the ISSA algorithm performs better in terms of stability. Although no SSA outliers appeared in $f_{11}$ and $f_{14}$, the variation range in optimal values of SSA is relatively large, which shows that the SSA algorithm is less stable than ISSA.

### 3.4. Radar Chart

In order to analyze the overall optimization capabilities of the seven algorithms, the results of the algorithms in each benchmark function are compared and ranked according to their mean values (see Table 3). If the means are equal, the standard deviations are compared; and if the standard deviations are also equal, the two are ranked identically. The ranking results are shown in Table 6. The last row in Table 6 is the average rank of each algorithm. ISSA ranks first and thus has the best optimization performance among the seven algorithms; it is also significantly better than SSA. The ranking results of the remaining algorithms are SSA, GWO/WOA, PSO, SSO, and SCA. In order to display the ranking of the seven algorithms more intuitively on different benchmark functions, a radar chart is used to plot the results of Table 6. This chart is shown in Figure 6. The smaller the area enclosed by the algorithm performance curve, the smaller the ranking values of the algorithm and the better the optimization performance. It can be seen from the figure that the area enclosed by the black curve representing ISSA is the smallest, indicating that ISSA has the best optimization performance overall among the seven algorithms.
4. Conclusion

In this paper, the idea of flight behavior of BSA and the idea of crossover and mutation of GA are introduced into SSA. Thus ISSA is proposed. Through the improvement, the ISSA can keep the diversity of the population at the end of iteration and keep the population in a better state. The improved method has good global convergence performance and robustness and shows better optimization ability and better comprehensive performance than the original algorithm. In the next step, it will be compared with other novel algorithms such as Monarch Butterfly Optimization (MBO) [32], Earthworm Optimization Algorithm (EWA) [33], Elephant Herding Optimization (EHO) [34], Moth Search (MS) algorithm [35, 36], Slime Mould Algorithm (SMA) [37], and Harris Hawk Optimization (HHO) [38] on specific problems, and it will be applied in practical engineering to verify the advantages of ISSA.

Data Availability

All data are included within the article.

Additional Points

Highlights. (1) Bird swarm algorithm and genetic algorithm are integrated into SSA; (2) Cauchy mutation and Tent chaotic sequence are introduced into SSA; (3) The greater performance of ISSA is verified over some classic algorithms.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors’ Contributions

Chenghai Li made a major contribution to this work.

References

[1] L. N. Yang, X. Sun, and Z. L. Li, “An efficient framework for remote sensing parallel processing: integrating the artificial bee colony algorithm and multiagent technology,” Remote Sensing, vol. 11, no. 152, pp. 1–21, 2019.
[2] S. H. Sina, P. Behrouz, and S. H. Sami, “Prioritizing and analyzing the role of climate and urban parameters in the confirmed cases of COVID-19 based on artificial intelligence applications,” International Journal of Environmental Research and Public Health, vol. 17, no. 10, Article ID 3730, 2020.
[3] J. Dash, B. Dam, and R. Swain, “Implementation of narrow-width automatic digital tuner using popular swarm intelligence technique,” Engineering Applications of Artificial Intelligence, vol. 79, no. 3, pp. 87–99, 2019.
[4] P. Hu, J. H. Pan, and S. H. Chu, "Improved binary grey wolf optimizer and its application for feature selection," Knowledge-Based Systems, vol. 195, no. 11, Article ID 105746, 2020.
[5] J. Kennedy and R. Eberhart, "Particle swarm optimization," in Proceedings of the International Conference on Neural Networks, pp. 1942–1948, IEEE, Perth, Australia, 1995.
[6] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," Advances in Engineering Software, vol. 69, no. 3, pp. 46–61, 2014.
[7] S. Mirjalili and A. Lewis, "The whale optimization algorithm," Advances in Engineering Software, vol. 95, no. 5, pp. 51–67, 2016.
[8] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili, "Salp swarm algorithm: a bio-inspired optimizer for engineering design problems," Advances in Engineering Software, vol. 114, no. 6, pp. 163–191, 2017.
[9] S. Mirjalili, "SCA: a sine cosine algorithm for solving optimization problems," Knowledge-Based Systems, vol. 96, no. 96, pp. 120–133, 2016.
[10] J. Xue and B. Shen, "A novel swarm intelligence optimization approach: sparrow search algorithm," Systems Science & Control Engineering, vol. 8, no. 1, pp. 22–34, 2020.
[11] W. Li, G.-G. Wang, and A. H. Gandomi, "A survey of learning-based intelligent optimization algorithms," Archives of Computational Methods in Engineering, vol. 21, no. 1, 2021.
[12] W. Li and G.-G. Wang, "Elephant herding optimization using dynamic topology and biogeography-based optimization based on learning for numerical optimization," Engineering with Computers, vol. 21, no. 20, 2021.
[13] W. Li, G. G. Wang, and A. H. Alavi, "Learning-based elephant herding optimization algorithm for solving numerical optimization problems," Knowledge-Based Systems, vol. 195, no. 11, Article ID 105675, 2020.
[14] J. Li, Y.-H. Yang, H. Lei, and G.-G. Wang, "Solving logistics distribution center location with improved cuckoo search algorithm," International Journal of Computational Intelligence Systems, vol. 14, no. 1, pp. 676–692, 2021.
[15] J. Li, D.-D. Xiao, H. Lei, Z. Ting, and T. Tian, "Using cuckoo search algorithm with Q-learning and genetic operation to solve the problem of logistics distribution center location," Mathematics, vol. 8, no. 2, Article ID 149, 2020.
[16] J. Li, Y.-X. Li, S.-S. Tian, and J.-L. Xia, "An improved cuckoo search algorithm with self-adaptive knowledge learning," Neural Computing and Applications, vol. 32, no. 16, pp. 11967–11997, 2020.
[17] J. Li, Y.-X. Li, S.-S. Tian, and J. Zou, "Dynamic cuckoo search algorithm based on Taguchi opposition-based search," International Journal of Bio-Inspired Computation, vol. 13, no. 1, Article ID 59, 2019.
[18] 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, no. 12, pp. 147–157, 2016.
[19] Y. Peng, G.-G. Wang, J. Dong, and L. Wang, "Opposition-based learning monarch butterfly optimization with Gaussian perturbation for large-scale 0-1 knapsack problem," Computers & Electrical Engineering, vol. 67, no. 1, pp. 454–468, 2018.
[20] G.-G. Wang, S. Deb, A. H. Gandomi, Z. Zhang, and H Amir, "Chaotic cuckoo search," Soft Computing, vol. 20, 2015.
[21] C. Lu, L. Gao, X. Li, C. Hu, X. Yan, and W. Gong, "Chaotic-based grey wolf optimizer for numerical and engineering optimization problems," Memetic Computing, vol. 12, no. 4, pp. 371–398, 2020.
[22] N. Nabil, A. E. Ahmed, A. E. Mohamed, and M. Majdi, "Boosting salp swarm algorithm by sine cosine algorithm and disrupt operator for feature selection," Expert Systems with Applications, vol. 145, no. 1, Article ID 113103, 2020.
[23] G. Y. Liu, C. Shu, Z. W. Liang, B. H. Peng, and L. F. Cheng, "A modified sparrow search algorithm with application in 3d route planning for UAV," Sensors, vol. 21, no. 4, Article ID 1224, 2021.
[24] J. Yuan, Z. Zhao, Y. Liu et al., "DMPPCT control of photovoltaic microgrid based on improved sparrow search algorithm," IEEE Access, vol. 9, pp. 16623–16629, 2021.
[25] Y. Lei, G. De, and L. Fei, "Improved sparrow search algorithm based DV-hop localization in WSN," in Proceedings of the 2020 Chinese Automation Congress (CAC), pp. 2240–2244, Shanghai, China, November 2020.
[26] J. N. Zhang, K. W. Xia, Z. P. He, Z. X. Yin, S. J. Wang, and A. Ahmadian, "Semi-supervised ensemble classifier with improved sparrow search algorithm and its application in pulmonary nodule detection," Mathematical Problems in Engineering, 2021, Article ID 662935, 18 pages, 2021.
[27] L. Shan, H. Qiang, J. Li, and Z. Q. Wang, "Chaotic optimization algorithm based on tent map," Control and Decision, vol. 2005, no. 2, pp. 179–182, 2005.
[28] X. B. Meng, X. Z. Gao, L. H. Lu, Y. Liu, and H. Z. Zhang, "A new bio-inspired optimisation algorithm: bird Swarm algorithm," Journal of Experimental & Theoretical Artificial Intelligence, vol. 28, no. 4, pp. 1–15, 2016.
[29] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in Proceedings of the MHS’95 Sixth International Symposium on Micro Machine & Human Science, IEEE, Nagoya, Japan, October 2002.
[30] J. Derrac, G. Salvador, M. Daniel, and H. Francisco, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," Swarm and Evolutionary Computation, vol. 1, no. 1, pp. 3–18, 2011.
[31] S. Rohit, S. Urvinder, S. Supreet, S. Gurdeep, and M. Nitin, "Self-adaptive salp swarm algorithm for engineering optimization problems," Applied Mathematical Modelling, vol. 89, no. 1, pp. 188–207, 2021.
[32] G. G. Wang, S. Deb, and Z. Cui, "Monarch butterfly optimization," Neural Computing Applications, vol. 31, no. 7, pp. 1995–2014, 2019.
[33] G. G. Wang, S. Deb, and L. D. S. Coelho, "Earthworm optimization: a bio-inspired metaheuristic algorithm for global optimization problems," International Journal of Bio-Inspired Computation, vol. 12, no. 1, pp. 1–22, 2018.
[34] G. G. Wang, S. Deb, X. Z. Gao, and L. D. S. Coelho, "A new metaheuristic optimization algorithm motivated by elephant herding behavior," International Journal of Bio-Inspired Computation, vol. 8, no. 6, pp. 394–409, 2016.
[35] Y. Feng and G. G. Wang, "Binary moth search algorithm for discounted 0–1 knapsack problem," IEEE Access, vol. 6, pp. 10708–10719, 2018.
[36] S. Mirjalili, "Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm," Knowledge Based Systems, vol. 89, pp. 228–249, 2015.
[37] S. Li, H. Chen, M. Wang, A. A. Heidari, and S. Mirjalili, "Slime mould algorithm: a new method for stochastic optimization," Future Generation Computer Systems, vol. 111, pp. 300–323, 2020.
[38] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, "Harris hawks optimization: algorithm and applications," Future Generation Computer Systems, vol. 28, pp. 1–23, 2018.