A Cooperative Coevolution Wingsuit Flying Search Algorithm with Spherical Evolution

Jiaru Yang¹ · Yu Zhang¹ · Ziqian Wang¹ · Yuki Todo² · Bo Lu³ · Shangce Gao¹

Received: 6 June 2021 / Accepted: 20 September 2021
© The Author(s) 2021

Abstract

The algorithm wingsuit flying search (WFS) mimics the procedure of landing the vehicle. The outstanding feature of WFS is parameterless and of rapid convergence. However, WFS also has its shortcomings, sometimes it will inevitably be trapped into local optima, thereby yield inferior solutions owing to its relatively weak exploration ability. Spherical evolution (SE) adopts a novel spherical search pattern that takes aim at splendid search ability. Cooperative coevolution is a useful parallel structure for reconciling algorithmic performance. Considering the complementary strengths of both algorithms, we herein propose a new hybrid algorithm that is comprised of SE and WFS using cooperative coevolution. During the search for optimal solutions in WFS, we replaced the original search matrix and introduced the spherical mechanism of SE, in parallel with coevolution to enhance the competitiveness of the population. The two distinct search dynamics were combined in a parallel and coevolutionary way, thereby getting a good search performance. The resultant hybrid algorithm, CCWFSSE, was tested on the CEC2017 benchmark set and 22 CEC 2011 real-world problems. The experimental data obtained can verify that CCWFSSE outperforms other algorithms in aspects of effectiveness and robustness.

Keywords Nature-inspired algorithms · Wingsuit flying search · Cooperative coevolution · Metaheuristic · Spherical evolution

1 Introduction

In terms of human development, several problem-solving processes are often heuristic [1–3]. However, it was only since the 1940s that heuristics were used as a scientific method for various applications [4]. The initial cornerstone in the domain of heuristics was the emergence of evolutionary algorithms, which greatly advanced the theorization and practicalization of heuristics [5, 6]. As population-based computational intelligence [7], evolutionary algorithms

(EAs) [8–11] are applicable to optimization problems and, exhibit immense potential for use in the field of optimization. The metaheuristic algorithm is an extension of the heuristic algorithm on which the stochastic algorithm and local search are fused [12, 13]. In definition, metaheuristic algorithm is proposed relative to the optimization algorithm [14–16].

Recently, nature-inspired metaheuristic (NMH) [17, 18] algorithms have yielded promising results in combinatorial optimization [19–21]. Inspired by the foraging behavior of bee swarm, Pham et al. developed a novel bee algorithm [22–24]. The gravitational search algorithm (GSA) [25, 26] derives from the law of gravitational and mass interactions. Particle swarm optimization (PSO) [27, 28] is an evolutionary computation technique originated from the study of bird feeding behavior. The basic idea of PSO is to find the optimal solution through collaboration and information sharing among individuals in a population. Based on the mechanism that the trajectories of the particles vary depending on the optimal location of an individual, this algorithm can effectively search for better solutions. Subsequently, Tian et al. integrated PSO with differential evolution (DE) [29–31] to

✉ Bo Lu
lb2364@sth.edu.cn

✉ Shangce Gao
gaosc@eng.u-toyama.ac.jp

¹ Faculty of Engineering, University of Toyama,
Toyama-shi 930-8555, Japan

² Faculty of Electrical, Information and Communication
Engineering, Kanazawa University, Kanazawa-shi 9201192,
Japan

³ Faculty of Engineering, Shanghai Normal University Tianhua
College, Shanghai, China

solve real-world problems, achieving success in arranging fire trucks. Brain storm optimization (BSO) [32, 33] inspired by human creative problem solving process uses the idea of clustering to search for the local optimum. Recently, an increasing number of nature-inspired metaheuristic algorithms are emerging and have been applied to practical engineering problems successfully [34, 35].

The metaheuristic is a process of embodying how to find, select, or generate a satisfactory solution. The general formulation is to choose a set of parameters (variables) that will lead to the optimum of the objective while satisfying a set of relevant constraints [36]. The optimization problem can be generally expressed in the following mathematical form:

$$\min f(X) \text{ or } \max f(X), X = (x_1, \dots, x_n), \quad (1)$$

subject to

$$h_i(X) = 0, (i = 1, 2, \dots, n);$$

$$g_j(X) \leq 0, (j = 1, 2, \dots, n).$$

Because of iterative methods, the metaheuristic strategy does not guarantee to obtain the global optimal solution [37–39]. However, compared to the computational effort required by heuristic or iterative algorithms, metaheuristic algorithms have the advantage of producing better solutions with less computational effort [40, 41]. The focus of the metaheuristic is on reinforcement, which is performed on the optimal principle for the solution to be chosen for the problem [42, 43]. Metaheuristic algorithms are universal because they involve specific assumptions about the problem to be solved [44–46].

With the continuous progress of computer science, the performance of new algorithms is getting better and better, which can solve more and more complex problems and demands [47, 48]. However, no algorithm is perfect. People try to integrate different algorithms to explore more potential algorithm mechanism [49, 50]. The leitmotiv of the hybrid algorithm is to study the root why algorithms behave differently in various aspects, and then combine the mechanisms of various algorithms in a certain way to achieve the purpose of complementing their advantages [51]. The common methods are to exchange information between two algorithms by running them in parallel or crossover, or to split the number of iterations into complementary running [52]. These hybrid algorithms have commonalities from the aspects of population, search mechanism, parallel connection, etc [53, 54].

There are numerous existing mainstream approaches to enhance the performance of hybrid algorithms, such as incorporating enhancement strategies into original algorithm, modifying parameters by self adaptive mechanism, or improving formula of algorithm to approximate objective function [55]. For example, the PSO-BP algorithm [56] combines PSO with the back-propagation algorithm (BP) to

train feed-forward neural networks. The hybrid algorithm assumes PSO and BP for global and local search, respectively. It has made a considerable progress in aspects of convergence and generalization ability. The AC-ABC algorithm [57] allocated the artificial bee colony algorithm to determine the best subset of features and the ant colony algorithm to avoid the stagnant time. Through hybridization, they manage to avoid the stagnant time of searching for the optimum. This algorithm has superior performance in improving the classification accuracy and finding the best selection of features. However, existing hybrid algorithms may still fail to find the global optimum for complex optimization problems because of their incomplete mechanism. Even the algorithm may play unstably and the performance depends heavily on the specific problem [58]. Metaheuristic algorithms nowadays tend to be more complex in structure and more cumbersome in the mechanism. Although the aspects taken into account are becoming more comprehensive, the algorithms always have different core flaws. It can be seen, from the above examples among many others, that there is a lot of room for mining to obtain better algorithms via hybridizing several different methods by combining their inherent characteristics [59–62].

In terms of intelligent algorithms, cooperative coevolution (CC) has proven to be an effective search strategy. It has been developed to further improve solution quality for large-scale optimization problems. CC is a common and valid way to enhance the effectiveness of the algorithm [63]. For an optimization problem, CC takes in a decomposition strategy that divides a problem into several groups and optimizes them in groups. Then, all groups cooperate to complete the optimization of the whole problem [64]. Many examples have proved the feasibility of CC in promoting the algorithm performance.

In this paper, a new algorithm called CCWFSSE that combines two recently developed algorithms—wingsuit flying search (WFS) [65] and spherical evolution (SE) [66] in the coevolutionary way was proposed. Both considered algorithms have their own characteristics. We utilize these features to combine algorithmic structures and explore more efficient modes of algorithm operation. The search mechanism of SE can provide a comprehensive solution search space, so as to avoid falling into local optimum. WFS can quickly identify the contemporary optimal individual and pass the Halton sequence to exploit several potential individuals. As SE focuses on powerful exploitation while WFS possesses the comprehensive exploration capabilities, hybridization of the search dynamics of these algorithms can reach a equilibrium between exploration and exploitation, thereby resulting in better search efficiency [67]. Meanwhile, we add CC to the original algorithm with a judgment mechanism for purpose of improving the population diversity of CCWFSSE. When solving optimization problems,

the fundamental aim is to find the global optimum solution within an acceptable time complexity, which becomes the basic principle guiding the design of intelligent algorithms. Therefore, we hope to achieve a balance between global search and local search by combining SE and WFS, while enhancing the convergence speed and optimization result of algorithm [68]. A series of experimental data analysis can verify that CCWFSSE has better search efficiency and robustness than other comparable algorithms [69].

The main contribution to this work can be outlined as follows: (1) Inspired by the complementary characteristics of different algorithms, we innovatively propose a hybrid algorithm that incorporates SE which focuses on local exploitation, into WFS, which possesses excellent global exploration capabilities. (2) Add the spherical search to the process of finding neighborhood points while using cooperative coevolution in parallel to maintain population diversity, which resulted in a good capability and robustness. (3) We have done a variety of experiments to test the proposed algorithm, which can prove that CCWFSSE performs well. This work provides more thoughts on how two or more different algorithms should be combined to build a better algorithm [70].

The structure of the rest part is: Sect. 2 and Sect. 3 separately introduce the algorithm WFS and SE. Section 4 clarifies specific ideas of the hybrid algorithm CCWFSSE. Section 5 gives the experimental results which are analyzed and discussed in detail. A discussion regarding many aspects of algorithm is done in Sect. 6. Ultimately, Sect. 7 reveals the conclusion and does a vision of the future.

2 Brief Description of WFS

The WFS algorithm is derived from wingsuit flying, a extreme sport. The core idea of WFS is to obtain an approximate image of the entire solution space. When this algorithm (flier) starts running (flying), it reaches a global minimum (landing site) in the search space (Earth's surface). The entire search space is probed by updating the population of points at each iteration. This process gradually shifts the focus to lower regions just as a flying machine acquires a gradually clearer image.

In essence, the algorithm is a population-based search mode, which constantly updates the solution points. Then, we specifically figure the operating mechanism of the algorithm in considering four aspects.

2.1 Population Initialization

In the initial search space, we define N as the number of population size and $N^* = N - 2$ as the number of initial points. In Fig. 1, there are initial points $\mathbf{x} = [x_1, x_2, \dots, x_n]^T$ randomly located in a box search space. Each point is uniformly distributed in an n -dimensional grid. N_0 is set as the number of

Fig. 1 The draft of N_0^2 initial points arrangement

nodes. There is an discrete distance between each point $\delta\mathbf{x}$, which is defined as:

$$\delta\mathbf{x} = \frac{\mathbf{x}_{\max} - \mathbf{x}_{\min}}{N_0} \quad (2)$$

Ascertain other points based on x_1 to constitute a full grid. Then choose the point whose the cost function evaluates the lowest. This step is intended to achieve a better uniform distribution of the populations to be selected in the whole solution space.

2.2 Identify the Neighborhood Size

After the first iteration, the points selected from the previous iteration are sorted in ascending order with respect to their solution values. The point with the highest solution value is assigned to a largest neighborhood with the number of points $P_{\max}$, the second gets a smaller neighborhood, and so on down the list. The function $P(i)$ with respect to rank i can be obtained as:

$$P(i) = \left\lceil P_{\max} \left(1 - \frac{i-1}{N-1} \right) \right\rceil \quad (3)$$

The assignment of corresponding neighbourhood spaces to points possessing different solution values is the core motivation for the algorithm to mimic a flying machine.

2.3 Call Neighborhood Points into Being

With the previous step done, we turn our attention to generating new valid points. Figure 2 shows an case demonstration. The dots filled in blue represent neighborhood points, and the white dots are where they might be located. For each point $\mathbf{x}_i^{(m)}$, create a vector oriented to the current solution defined as $\mathbf{v}_i^{(m)}$. The coordinate of each neighborhood point is originated based on Eq. (4):

Fig. 2 The draft of $N^{(m)}$ points and their neighborhoods

$$\begin{aligned} S_1(x_i^{(m)}) &= \{x_i^{(m)} - \delta x^{(m)}, x_i^{(m)}\}, \text{ if } v_i^{(m)} < 0 \\ S_2(x_i^{(m)}) &= \{x_i^{(m)}, x_i^{(m)} + \delta x^{(m)}\}, \text{ if } v_i^{(m)} > 0 \\ S_3(x_i^{(m)}) &= S_1(x_i^{(m)}) \cup S_2(x_i^{(m)}), \text{ if } v_i^{(m)} = 0 \end{aligned} \quad (4)$$

The probability of finding the global optimum is further improved by generating new valid points in the vicinity of the optimal solution selected in the previous generation.

2.4 Generating Centroid and Random Points

The flier often prefers to locate the middle of the ground surface. Similarly, we set a centroid to help locate the present search position and to effectively perform spatial localization. Further, we choose a point randomly from the minimum boundary of m -th iteration and set it as a random point to add to the search space. These steps can effectively enhance the exploration ability of WFS.

3 Spherical Evolution

SE is a recently developed NMH algorithm. It is a mathematical model based on search operators inspired by the traditional search pattern. SE studied the search patterns of a large number of traditional algorithms and found that most of the functions they use can be understood as a matrix search style, which enlightened the development of the spherical evolutionary algorithm. The SE operating mechanism can be briefly explained by three random vectors — X_1 , X_2 , and X_3 . These three vectors are selected from the population and the initial solution is X_1 . When searching a circular area by the radius $|X_2X_3|$, we will get a new vector X_{new} to shift the old solution X_{old} . On the 2D space, the radius and angle are constantly adjusted to create updated units so that the whole search area can be covered. Thus, this mechanism can be simply represented as:

$$X_{i,d}^{\text{new}} = X_{\gamma,d} + \sum_{k=1}^n S(X_{\alpha,d}^k, X_{\beta,d}^k) \quad (5)$$

where d is the number of dimension. $X_{i,d}^{\text{new}}$ demonstrates the new i -th solution. X_{α} , X_{β} and X_{γ} , represent the three solutions picked based on a specific strategy. $S(X_{\alpha,d}^k, X_{\beta,d}^k)$ stands for the updated units.

SE can well realize the operation of the search operator in all dimensions. When the number of dimensions increases, the spherical search functions based on the Euclidean distance. Euclidean distance refers to the true interval between two units in m -dimensional space. In one-dimensional, two-dimensional and high-dimensional approaches, the search schema of SE can be conducted by Eqs. (6, 7, 8), respectively:

$$SS_1(X_{\alpha,d}, X_{\beta,d}) = SF() \cdot |X_{\alpha,d} - X_{\beta,d}| \cdot \cos(\theta) \quad (6)$$

$$SS_2(X_{\alpha,d}, X_{\beta,d}) = SF() \cdot |X_{\alpha,*} - X_{\beta,*}| \cdot \sin(\theta) \quad (7)$$

$$SS_2(X_{\alpha,d}, X_{\beta,d}) = SF() \cdot |X_{\alpha,*} - X_{\beta,*}| \cdot \cos(\theta)$$

$$\begin{aligned} SS_{\geq 3}(X_{\alpha,d}, X_{\beta,d}) &= SF() \cdot ||X_{\alpha,*} - X_{\beta,*}||_2 \\ &\cdot \prod_{k=d}^{dim-1} \sin(\theta), d = 1 \\ SS_{\geq 3}(X_{\alpha,d}, X_{\beta,d}) &= SF() \cdot ||X_{\alpha,*} - X_{\beta,*}||_2 \\ &\cdot \cos(\theta_{d-1}) \cdot \prod_{k=d}^{dim-1} \sin(\theta), 1 < d \leq dim - 1 \\ SS_{\geq 3}(X_{\alpha,d}, X_{\beta,d}) &= SF() \\ &\cdot ||X_{\alpha,*} - X_{\beta,*}||_2 \cdot \cos(\theta_{d-1}), d = dim \end{aligned} \quad (8)$$

where $|X_{\alpha,d} - X_{\beta,d}|$ stands for the absolute distance between two in 1-dimension. $||X_{\alpha,*} - X_{\beta,*}||_2$ represents the distance in the high-dimensional case. θ denotes a random angle between $X_{\alpha,*}$ and $X_{\beta,*}$.

The mechanism of SE is simple and the search range is large. Therefore, it is an effective algorithm that can be applied to a variety of problems [71].

4 Spherical Mechanism-Driven Cooperative Coevolution Wingsuit Flying Search

4.1 Motivation

By studying the mechanism of metaheuristic algorithms, we find that most metaheuristic algorithms have two characteristics in common. The first one is about search

patterns. A simple and efficient search pattern can help each individual population find a better solution. The second to be concerned is the selection way of individual. Several methods have been proposed, such as stochastic universal sampling method, tournament selection, and roulette wheel selection [4].

Search pattern has always been the core problem of algorithms, and it has a very important impact on algorithm. The common characteristics of different search patterns cannot be represented in a single template. For example, in grey wolf optimization (GWO) [72], grey wolf i updates its position by the differential perturbation between it and the three best wolves. In DE, the initial point is denoted as j and its position is updated by randomly selected individuals. In PSO, a particle k renovates its position with a velocity item as two update units. In fact, many metaheuristic algorithms utilize perturbation differences between individuals or binary crossover to implement search patterns.

To conclude, the common behavior of search, such as binary search and sequential search, usually plays a crucial role in the operation of an algorithm. WFS uses a search pattern based on population by updating candidate solution points. At the same time, SE is a simple and efficient heuristic approach that has more potential to search the whole promising solution space. As WFS and SE have their own operating characteristics, this inspires us to incorporate search scheme into the design of an algorithm. Next, we specifically explain how to combine WFS and SE through coevolution to achieve the purpose of complementary advantages.

4.2 Issues of WFS

WFS can independently optimize different locations of the whole solution space. However, in the iterative process, several candidate populations are missing due to the search mechanism. By contrast, SE possesses excellent performance in terms of search capability, while its problem-solving ability and convergence speed are unsatisfactory. Therefore, we attempt to incorporate SE into the procedure of exploring the local space in WFS, hoping that a perfect combination of both can be achieved.

From Eqs. (4), we can determine the size of the biggest neighborhood. Depending on the value of the vector $\mathbf{v}_i^{(m)}$, there are two types of neighborhoods, one directed and one non-strictly directed. Thus, when choosing neighborhood points, there are also two choices. Points are selected from the directed neighborhood or some remaining points are selected from the non-strictly directed neighborhood.

When generating neighbourhood points, WFS uses the Halton sequence and the rectangular search to determine

the location of the solution. Therefore, this process inevitably misses the opportunity to obtain the best solution. To enhance the exploration capacities of the algorithms, a random point is selected by a specific strategy and added to the solution space. Therefore, to improve the probability of obtaining the best solution, we intend to improve the original method of generating neighborhood points. Thus, SE is considered due to its powerful search capability.

4.3 Spherical Search Scheme

Most of the existing mainstream algorithms adopt a hypercube search scheme. As shown in Fig. 3, the hypercube is represented as a rectangle space in 2D. By regulating the radius $|B_iD|$ or $|B_iA|$, the rectangle search space can be covered. The black line with the arrow (B_iC_i) represents the updating search trajectory of DE when the crossover rate (CR) is 1 in 2D space. The blue one indicates the updating search trajectory of DE in the case of only one dimension to choose. The red one indicates the updating search trajectory of PSO as the dimensional scale values are different.

It is well known that the area of a circle is greatest for a certain circumference. Inspired by the above, spherical evolutionary is invented. In contrast, the spherical search pattern covers the solution space by continually adjusting the vector. To illustrate the mechanism of SE, an example in 2D is shown. From the Fig. 4, the dashed lines with arrows OA_i and OB_i represent two different solution vectors. The entire region occupied by the circle is regarded as a complete solution space. By rotating the angle α and adjusting the radius $|A_iB_i|$, a updated point B'_i is generated. Evidently, when rotating the angle α and the radius from 0 to $|A_iB_i|$, we can search the entire area of the circle. In Figs. 3 and 4, the spherical search mechanism covers a full-scale solution area than the hypercube does. Thus, the spherical mechanism possesses a better exploration capability than the hypercube does. The search mechanism of SE can thus address the shortcomings

Fig. 3 Hypercube search in 2D space

Fig. 4 Spherical search in 2D space

of WFS with regard to the insufficient use of the search space and can increase the population diversity as well.

4.4 Cooperative Coevolution

As the complexity of systems increased, the concept of modularity could be introduced to the solution process. CC solves complex optimization problems in the form of co-fitted subcomponents. In CC, complex problems are decomposed into subproblems, which are solved in evolutionary subpopulations. Individual evaluation depends on cooperation between subpopulations, and a complete solution is obtained by combining representative individuals of each subpopulation [73].

The original cooperative coevolution can be briefly outlined: (1) Problem decomposition, n -dimensional solution vector is split into multiple low-dimensional subparts. (2) Random initialization of subpopulations. (3) Perform a cycle to generate progeny subpopulations for individual fitness value assessment. (4) Update the solution vector and select the next generation subpopulation.

The loop includes the complete evolution of all subcomponents. When applying a CC, the epistatic interaction between variables is an important aspect to be considered. The interaction may have a negative impact on the

convergence rate. Compared with conventional evolutionary algorithms, a salient advantage of CC is the good diversity and maneuverability using several subcomponents.

4.5 Spherical Mechanism-Driven Cooperative Coevolution Wingsuit Flying Search

Considering the different characteristics of the algorithm, we propose a new hybrid algorithm CCWFSSE herein. The differences between the mechanisms of WFS and SE result in differences between their performance. In WFS, each iteration of the population shifts the focus to lower regions. The main feature of this algorithm is its emphasis on more potential points. That is to say, exploitation of the solution space receives more attention in this process, while exploration is relatively sub-priority. On the contrary, in SE, the search is more directional than in other metaheuristic algorithms because the optimal population is based on the prior best (pbest) and global best positions (gbest). In summary, WFS and SE have their own advantages and disadvantages. The objective is to associate the superiority of both algorithms in order to improve the comprehensive ability. For example, introduce an SE mechanism to enhance the global search capability of WFS while avoiding reducing the search speed. Synchronously adding CC in the search process can increase the potential of searching the entire space and improve the population diversity.

WFS uses Horton sequences to generate random points, which is a standard hypercube mechanism. In this process, we introduce the phasor difference and spherical search pattern. Based on two random selected solution vectors X_1, X_2 , a updated vector X_{new} is produced and is add to the population. Owing to the ability of searching a comprehensive solution space, SE uses δ for local space exploitation, which leads to a risk of neighbourhood points entering the local optimum. However, the spherical mechanism promotes the exploration of the solution space in addition to balancing its exploration and search capabilities.

Algorithm 1: Overall pseudo-code of CCWFSSE

```

1 begin
2   / Initialization /
3    $v \leftarrow$  a random real number between [10,100]
4    $N = N^* - 2$ 
5    $N_0 = \text{floor}(\text{sqrt}[\frac{1}{n}]N)$ 
6    $\Delta x = (x_{\max} - x_{\min}) * N_0$ 
7    $X \leftarrow$  Randomly generate  $N_0$  individuals
8   Generate the rest  $N - N_0$  individuals based on  $X$  and  $\Delta x$ , and incorporated into  $X$ 
9    $f(X) \leftarrow$  Evaluate fitness of  $X$ 
10  while Not satisfied the ultimate condition do
11    / WFS operator /
12     $\alpha = 1 - v^{-\frac{m-1}{M-1}}$ 
13     $P_{\max} = \text{floor}(\alpha \times N)$ 
14     $N = \text{floor}(\frac{2N}{P_{\max}})$ 
15     $X \leftarrow$  sort  $X$  through  $f(X)$  and choose top  $N$  individuals
16     $P \leftarrow$  calculate neighborhood size
17    / SE operator /
18    for  $i=1 : N$  do
19      for  $j=1 : P$  do
20        if  $\text{mod}(j) == 0$  then
21           $x \leftarrow$  Generate neighborhood points by WFS
22        end
23        else
24           $C = C_{\min} + t_{\text{current}}/t_{\max} \times (C_{\max} - C_{\min})$  if  $C > \text{rand}$  then
25             $x = CC(x)$ 
26          end
27          else
28            Randomly choose two individual  $X_1, X_2$ 
29             $\text{dim} \leftarrow$  randomly choose dimension to be improved
30             $x = SS(X_1, X_2, \text{dim})$ 
31          end
32        end
33         $X = [X; x]$ 
34      end
35      Generate a centroid point and a random point, then update  $X$ 
36    end
37  end
38 end

```

Therefore, a combination of parallelized computation and rapid convergence enables the proposed algorithm to achieve a good performance on exploration and exploitation. This performance can be considered as a unique property of the algorithm because CCWFSSE is essentially parameter-free.

The general steps of CCWFSSE can be represented as follows:

- (1) The algorithm generates an initial population based on a certain strategy.
- (2) The threshold of aviator is set, and a proposed solution is selected via ranking.

- (3) Neighborhood points are generated for a selected point using the phasor difference and spherical search pattern and are added to the solution space. In addition to these two strategy, cooperative coevolution operator is more possible to be adopted than SE when iteration number is relatively high.
- (4) The eligible points selected are added to the population.
- (5) Repeat the Steps (2)–(4) until the limiting condition is reached.

Figure 5 reveals the operating flowchart of this algorithm in details. The CCWFSSE pseudocode is shown in Algorithm. 1. Lines 11–16 represent the process that sorts X and selects the appropriate point. Subsequently, we generate the

Fig. 5 The flowchart of CCWFSSE

neighborhood point. As shown in lines 18–34, we utilize the vector difference to select a random point when iteration number is relatively small. When iteration number is relatively high, we tend to choose cooperative coevolution to select point. Then add that point and the centroid point to the population.

5 Experimental Results

We expect that the improved algorithm can be significantly competitive. Therefore, the performance of the proposed CCWFSSE algorithm was verified using IEEE CEC2017. In this section, first, the CEC2017 benchmark functions are introduced. Second, comparison between CCWFSSE and other metaheuristic algorithms is presented. Then, a series of experimental results in terms of the mean error with the known optimal values [74–76] and statistical test are analyzed. Third, the performance of CCWFSSE on some real-world problems is described and analyzed.

5.1 Benchmark Functions

IEEE CEC2017 is recognized as a valid test suite for evaluating algorithm performance. This benchmark is comprised by 30 different problems. The F2 function was excluded in this study due to its unstable performance in high dimensions. Therefore, 29 functions including unimodal functions (F1 and F3), simple multimodal functions (F4–F10), hybrid functions (F11–F20) and composition functions (F21–F30) were used.

5.2 Experiment Setup

The parameters are set as follows: the size of population N was set as 30,000; The dimension of functions was 30, so the $MaxFEs$ was $10^4 \times 30$. All algorithms were performed on a PC with a Intel(R) Core(TM) i5-7400 CPU at 3.00 GHz with 16GB of RAM. Each algorithm was run 51 times independently for each function to achieve accurate results.

5.3 Performance Comparison

We performed three different evaluation to assess the performance of CCWFSSE. The first was the Wilcoxon rank-sum test which was conducted to compare the performance of CCWFSSE with that of WFS, PSO, DE, and butterfly optimization algorithm (BOA) [77]. BOA is a natural heuristic algorithm based on the feeding process of butterflies, which has a high convergence accuracy. This test is for determining whether there is a significant difference between two groups of data with unknown distribution types and different sample sizes. The parameter settings are shown in Table 1.

Table 1 Parameter settings of the heuristic algorithms

Algorithms	Parameters
DE	$F = 0.8, CR = 0.2$
PSO	$c_1 = 2, c_2 = 2$
BOA	$p = 0.8$
GWO	α linearly decreases from 2 to 0
WFS	v randomly distributes from 10 to 100
CCWFSSE	v randomly distributes from 10 to 100

Table 2 reveals the experiment results. The best results for each tested problem among all compared algorithms are highlighted in bold. The sign “+”, “ $\approx$ ” and “–” represent that the target algorithm is better/similar to/worse than its competitor, respectively. The notations “w/t/l” denotes whether CCWFSSE performs better/the same as/worse than the other algorithms. The bolded values indicate the best results in a control group. We can learn that the number of wins, ties, and losses of CCWFSSE are 13/13/3, 29/0/0, 29/0/0, 29/0/0 against WFS, PSO, DE, and BOA, respectively. There is no significant difference between CCWFSSE and the original algorithm WFS in terms of the unimodal functions, because these functions only consist of a global optimum. However, for multimodal functions, given the existence of local optimum which may cause WFS to stall prematurely, the improved algorithm CCWFSSE performs significantly superior than WFS. It can prove that our approach is effective in avoiding falling into local optimum. From the data, CCWFSSE has a good performance particularly on separable benchmark functions, multimodal and non-separable benchmark functions.

The second evaluation is the convergence curve graph which can display the convergence rate of different algorithms. It exhibits the track of the current optimal solution. Figure 6 shows the six typical functions F5, F7, F9, F16, F20, and F28. The X-axis represents the number of evaluations, and the Y-axis refers to the average fitness obtained until a given time. These results show that the slope of the curve of CCWFSSE is gentler than that of the other four algorithms. At the initial phase, CCWFSSE exhibits a clear intention of converging, while other four algorithms converge considerably slower in comparison. It is worth mentioning that there are more concerns about whether the solution provided by NMH algorithm is the correct method to solve the problem. This validity sometimes comes in form of the final result of convergence. The fact that our proposed CCWFSSE converges efficiently verifies the validity of the algorithm. Thus, CCWFSSE can quickly converge to the desired target and effectively eliminate the local optimum.

The last evaluation is the box-and-whisker plot, which reflects the distribution and dispersion of the data. Figure 7 presents box-and-whisker graphs on F5, F7, F9, F16,

Table 2 Experimental results of CCWFSSE and other algorithms on CEC2017 benchmark functions

Algorithm	F1	F2	F3	F4	F5
CCWFSSE	1.70E+09 ± 9.68E+08	1.97E+04 ± 6.35E+03	6.55E+02 ± 7.08E+01	6.48E+02 ± 3.18E+01	6.24E+02 ± 6.22E+00
WFS	1.24E+09 ± 7.54E+08 –	2.17E+04 ± 7.16E+03 +	6.19E+02 ± 4.64E+01 ≈	6.50E+02 ± 3.45E+01 +	6.29E+02 ± 7.13E+00 +
PSO	1.56E+10 ± 2.92E+09 +	7.56E+04 ± 1.27E+04 +	1.24E+03 ± 3.40E+02 +	8.00E+02 ± 2.01E+01 +	6.49E+02 ± 5.56E+00 +
DE	4.10E+10 ± 3.65E+09 +	1.15E+05 ± 1.19E+04 +	7.04E+03 ± 1.12E+03 +	8.86E+02 ± 1.97E+01 +	6.76E+02 ± 4.36E+00 +
BOA	5.75E+10 ± 7.38E+09 +	1.30E+05 ± 2.03E+04 +	5.81E+03 ± 1.15E+03 +	8.56E+02 ± 1.74E+01 +	6.75E+02 ± 4.76E+00 +
Algorithm	F6	F7	F8	F9	F10
CCWFSSE	9.62E+02 ± 3.59E+01	9.41E+02 ± 3.45E+01	2.96E+03 ± 1.07E+03	5.42E+03 ± 6.96E+02	1.39E+03 ± 5.96E+01
WFS	9.43E+02 ± 2.65E+01 ≈	9.49E+02 ± 3.15E+01 ≈	3.65E+03 ± 1.64E+03 +	5.64E+03 ± 9.57E+02 +	1.53E+03 ± 1.12E+02 +
PSO	1.44E+03 ± 8.61E+01 +	1.09E+03 ± 1.72E+01 +	8.54E+03 ± 1.57E+03 +	8.04E+03 ± 3.87E+02 +	2.49E+03 ± 2.93E+02 +
DE	2.01E+03 ± 8.95E+01 +	1.16E+03 ± 1.63E+01 +	1.34E+04 ± 1.61E+03 +	8.06E+03 ± 2.70E+02 +	6.68E+03 ± 1.01E+03 +
BOA	1.23E+03 ± 2.79E+01 +	1.10E+03 ± 1.52E+01 +	9.49E+03 ± 1.04E+03 +	8.35E+03 ± 3.49E+02 +	7.08E+03 ± 2.07E+03 +
Algorithm	F11	F12	F13	F14	F15
CCWFSSE	5.96E+07 ± 4.00E+07	2.51E+06 ± 4.76E+06	1.60E+04 ± 1.23E+04	1.98E+05 ± 1.86E+05	2.48E+03 ± 2.67E+02
WFS	9.24E+07 ± 7.03E+07 +	2.15E+06 ± 3.29E+06 –	1.32E+04 ± 1.15E+04 ≈	3.00E+05 ± 6.06E+05 +	2.54E+03 ± 2.83E+02 ≈
PSO	5.13E+08 ± 1.55E+08 +	8.47E+07 ± 3.31E+07 +	1.20E+05 ± 7.68E+04 +	9.12E+06 ± 5.45E+06 +	3.41E+03 ± 2.88E+02 +
DE	3.88E+09 ± 7.40E+08 +	1.48E+09 ± 4.07E+08 +	5.51E+05 ± 2.34E+05 +	1.19E+08 ± 3.56E+07 +	4.13E+03 ± 1.67E+02 +
BOA	8.16E+09 ± 1.55E+09 +	4.25E+09 ± 1.11E+09 +	1.03E+06 ± 5.69E+05 +	5.23E+08 ± 1.78E+08 +	4.38E+03 ± 2.54E+02 +
Algorithm	F16	F17	F18	F19	F20
CCWFSSE	1.99E+03 ± 1.22E+02	3.60E+05 ± 3.68E+05	4.15E+05 ± 3.18E+05	2.35E+03 ± 1.32E+02	2.43E+03 ± 2.88E+01
WFS	2.08E+03 ± 1.17E+02 ≈	3.40E+05 ± 2.54E+05 –	4.11E+06 ± 3.56E+06 ≈	2.41E+03 ± 1.21E+02 +	2.44E+03 ± 2.56E+01 ≈
PSO	2.46E+03 ± 1.64E+02 +	2.19E+06 ± 1.19E+06 +	2.13E+07 ± 1.39E+07 +	2.59E+03 ± 1.34E+02 +	2.57E+03 ± 2.25E+01 +
DE	2.83E+03 ± 1.44E+02 +	6.83E+06 ± 2.59E+06 +	1.96E+08 ± 9.55E+07 +	2.75E+03 ± 9.27E+01 +	2.66E+03 ± 1.58E+01 +
BOA	2.53E+03 ± 2.03E+02 +	1.85E+07 ± 8.36E+06 +	6.48E+08 ± 2.83E+08 +	2.78E+03 ± 1.12E+02 +	2.64E+03 ± 2.43E+01 +
Algorithm	F21	F22	F23	F24	F25
CCWFSSE	2.59E+03 ± 1.18E+02	2.80E+03 ± 3.23E+01	2.97E+03 ± 2.40E+01	3.03E+03 ± 4.51E+01	4.81E+03 ± 8.24E+02
WFS	2.59E+03 ± 1.08E+02 ≈	2.82E+03 ± 3.90E+01 ≈	2.97E+03 ± 3.04E+01 ≈	3.05E+03 ± 6.07E+01 +	5.04E+03 ± 8.78E+02 +
PSO	3.99E+03 ± 2.79E+02 +	2.92E+03 ± 2.98E+01 +	3.07E+03 ± 1.50E+01 +	3.66E+03 ± 2.22E+02 +	6.53E+03 ± 4.48E+02 +
DE	7.46E+03 ± 5.24E+02 +	3.12E+03 ± 2.98E+01 +	3.27E+03 ± 2.77E+01 +	5.74E+03 ± 4.68E+02 +	8.68E+03 ± 4.53E+02 +
BOA	5.49E+03 ± 6.53E+02 +	3.12E+03 ± 4.36E+01 +	3.27E+03 ± 3.40E+01 +	3.82E+03 ± 1.30E+02 +	8.72E+03 ± 5.71E+02 +
Algorithm	F26	F27	F28	F29	w/hll
WFSSE	3.23E+03 ± 3.62E+01	3.43E+03 ± 8.29E+01	3.70E+03 ± 1.65E+02	2.47E+06 ± 2.17E+06	13/13/3
WFS	3.29E+03 ± 3.39E+01 ≈	3.45E+03 ± 8.54E+01 ≈	3.89E+03 ± 1.50E+02 +	8.63E+06 ± 7.22E+06 +	29/0/0
PSO	3.25E+03 ± 1.76E+01 +	3.78E+03 ± 1.55E+02 +	4.26E+03 ± 2.25E+02 +	1.87E+07 ± 9.11E+06 +	29/0/0
DE	3.56E+03 ± 4.93E+01 +	5.88E+03 ± 3.69E+02 +	5.33E+03 ± 1.95E+02 +	1.80E+08 ± 4.56E+07 +	29/0/0
BOA	3.65E+03 ± 9.26E+01 +	5.17E+03 ± 2.76E+02 +	4.74E+03 ± 2.90E+02 +	4.53E+08 ± 1.48E+08 +	29/0/0

F25, and F28. The shorter distance between the limit values represents the smaller deviation and the stable search performance. The red crosses are outliers in a set of data that allow people to analyze the cause of the outliers and eliminate them. The top and bottom black lines represent the boundary fitness values. The middle line indicates the median value of the data. From the Fig. 7, the experimental results reveals that all kinds of image indexes of CCWFSSE are almost lower than others. This can prove that CCWFSSE possesses the ability to search out the better solutions. With respect to the simple multimodal functions F5, F9 and composition function F28, the CCWFSSE has better stability compared with the other algorithms.

Baese on these experiments, we can conclude that CCWFSSE is effective on numerical function optimization. Overall, the CCWFSSE is a considerable improvement compared with the original algorithm in terms of convergence speed, mean error, and stability.

5.4 Real-World Problems

To understand the performance of CCWFSSE more clearly, 22 IEEE CEC2011 real-world problems are selected as a test set. Whether an algorithm can adapt to different real complex problems is also an important metric. Table 3 shows the mean, standard deviation, and the final result of this test. We compare CCWFSSE with a control group of WFS, PSO, DE, and GWO. The outcome represents that the CCWFSSE exceeds other algorithms in performance.

In particular, satisfactory results were achieved for F1, F5, F7, F10, and F13. F1 is a six-dimensional optimization problem that works to optimize the FM settings. This problem is a key issue in the field of music. F5 involves the evaluation of the atomic potential of covalent systems, especially silicon covalent systems. It has recently attracted considerable research interest. F7 is a question about radar pulses. When designing a radar system, the proper waveform is of paramount importance. F10 is a spherical antenna array problem that can be applied to sonar, sensing, communication and other fields. F13 is a spacecraft trajectory optimization problem to find the best safe path for the vehicle. Therefore, based on our findings, CCWFSSE can cope with complex real-world problems to a certain extent.

6 Discussion

6.1 Exploitation and Exploration

During the operation of an algorithm, the different stages can be abstractly understood as a dynamic development process.

Two main aspects of the operating process are exploitation and exploration. Exploitation refers to the probe on a specific local area, considering that a suitable solution is observed or predicted, while exploration refers to generating the diverse solutions on a global scale [78]. These two can well explain the behavior pattern of the algorithm in different stages and regions. When choosing an optimal solution, local exploration can speed up the convergence to the optimum, while diverse exploitation can prevent the algorithm from falling into a local optimum [79]. An appropriate alliance of aspects can ensure the implementation of a global optimal and accelerate the convergence rate.

To assess the performance of CCWFSSE about exploration and exploitation, typical graphs of population distribution on three types of functions with two dimensions were plotted, as show in Fig. 8. These functions includes unimodal function (F1), multimodal function (F8), and composition function (F27), where the size of population is 30,000, the maximum number of iterations is 10, and the search range of each dimension is $[-100; 100]$. Alphabet t means the current number of iteration. The blue point represents the global optimum. The red points denote the current location of each individual. The distribution density of individuals in the initial iteration ($t = 1$) is uniform. These individuals are uniformly generated based on the Holden sequence and begin to enter the WFS mechanism to find relatively suitable individuals. The entire algorithm begins to enter the exploration process. The first clear turning point is that when $t = 4$, the individual converges to the best sub-region until a stagnant state is reached. Subsequently, the spherical search mechanism is used to search for neighbors of the stagnant crowd; thus, the number of available populations increases. This represents a suitable exploitation process. Finally, the individual converges to the global optimal region. CCWFSSE compensates for the one-sidedness of other algorithms by controlling the mutual conversion of these two search mechanisms.

6.2 Conceptual Comparison Between CCWFSSE and Other Heuristic Algorithms

Currently, most mainstream metaheuristic algorithms use a pattern matrix. The pattern matrix includes solutions considered in a single algorithm iteration and multiple global iterations [80]. The optimal solution can be rapidly improved through iteration, and the degree of improvement is called the convergence speed. In genetic algorithms, for example, a chromosome is equivalent to a pattern matrix. In PSO, the matrix is considered as an ensemble, with each pattern corresponding to a particle. Several heuristic algorithms improve the pattern matrix by using basic genetic rules and laws of operations. Nevertheless, the variety in the properties of these algorithms decreases as

Table 3 The experimental results of CCWFSSE and other algorithms on real-world problems

Algorithm	F_1	F_2	F_3	F_4	F_5
CCWFSSE	$1.11\text{E}+01 \pm 4.04\text{E}+00$	$-1.20\text{E}+01 \pm 1.85\text{E}+00$	$1.15\text{E}-05 \pm 5.67\text{E}-19$	$1.47\text{E}+01 \pm 3.34\text{E}+00$	$-3.05\text{E}+01 \pm 2.94\text{E}+00$
WFS	$1.42\text{E}+01 \pm 5.54\text{E}+00 +$	$-9.23\text{E}+00 \pm 2.59\text{E}+00 +$	$1.15\text{E}-05 \pm 1.09\text{E}-18 \approx$	$1.99\text{E}+01 \pm 2.41\text{E}+00 +$	$-2.59\text{E}+01 \pm 3.45\text{E}+00 +$
PSO	$1.61\text{E}+01 \pm 3.08\text{E}+00 +$	$-5.55\text{E}+00 \pm 1.06\text{E}+00 +$	$1.15\text{E}-05 \pm 1.26\text{E}-13 \approx$	$1.81\text{E}+01 \pm 3.89\text{E}-01 +$	$-1.95\text{E}+01 \pm 1.68\text{E}+00 +$
DE	$1.91\text{E}+01 \pm 2.36\text{E}+00 +$	$-1.07\text{E}+01 \pm 8.62\text{E}-01 +$	$1.15\text{E}-05 \pm 5.92\text{E}-15 \approx$	$1.71\text{E}+01 \pm 2.72\text{E}+00 +$	$-1.47\text{E}+01 \pm 1.70\text{E}+00 +$
GWO	$1.38\text{E}+01 \pm 5.88\text{E}+00 +$	$-1.64\text{E}+01 \pm 1.65\text{E}+00 -$	$1.15\text{E}-05 \pm 2.45\text{E}-12 \approx$	$1.44\text{E}+01 \pm 2.19\text{E}-01 +$	$-2.39\text{E}+01 \pm 4.14\text{E}+00 +$
Algorithm	F_6	F_7	F_8	F_9	F_{10}
CCWFSSE	$-2.23\text{E}+01 \pm 2.93\text{E}+00$	$1.14\text{E}+00 \pm 2.02\text{E}-01$	$2.20\text{E}+02 \pm 0.00\text{E}+00$	$1.73\text{E}+06 \pm 1.11\text{E}+05$	$-1.69\text{E}+01 \pm 2.12\text{E}+00$
WFS	$-2.00\text{E}+01 \pm 2.95\text{E}+00 \approx$	$1.32\text{E}+00 \pm 2.51\text{E}-01 +$	$2.20\text{E}+02 \pm 2.13\text{E}+00 \approx$	$1.72\text{E}+06 \pm 1.80\text{E}+05 \approx$	$-1.65\text{E}+01 \pm 2.08\text{E}+00 \approx$
PSO	$-1.37\text{E}+01 \pm 1.72\text{E}+00 +$	$1.63\text{E}+00 \pm 1.36\text{E}-01 +$	$2.20\text{E}+02 \pm 0.00\text{E}+00 \approx$	$6.69\text{E}+05 \pm 1.34\text{E}+05 -$	$-1.44\text{E}+01 \pm 1.52\text{E}+00 +$
DE	$4.97\text{E}+00 \pm 3.83\text{E}+00 +$	$1.67\text{E}+00 \pm 9.81\text{E}-02 +$	$2.20\text{E}+02 \pm 0.00\text{E}+00 \approx$	$7.61\text{E}+06 \pm 1.92\text{E}+05 +$	$-1.02\text{E}+01 \pm 5.11\text{E}-02 +$
GWO	$-1.33\text{E}+01 \pm 6.32\text{E}+00 +$	$1.51\text{E}+00 \pm 2.37\text{E}-01 +$	$2.20\text{E}+02 \pm 0.00\text{E}+00 \approx$	$2.02\text{E}+06 \pm 9.28\text{E}+04 \approx$	$-1.03\text{E}+01 \pm 1.73\text{E}-02 +$
Algorithm	$F_{11,1}$	$F_{11,2}$	$F_{11,3}$	$F_{11,4}$	$F_{11,5}$
CCWFSSE	$1.61\text{E}+06 \pm 1.86\text{E}+05$	$4.38\text{E}+07 \pm 1.05\text{E}+06$	$1.55\text{E}+04 \pm 8.88\text{E}+00$	$1.92\text{E}+04 \pm 1.69\text{E}+02$	$3.43\text{E}+04 \pm 5.44\text{E}+03$
WFS	$1.64\text{E}+06 \pm 1.90\text{E}+05 \approx$	$4.16\text{E}+07 \pm 1.75\text{E}+06 +$	$1.55\text{E}+04 \pm 8.47\text{E}+00 \approx$	$1.92\text{E}+04 \pm 2.16\text{E}+02 \approx$	$3.31\text{E}+04 \pm 5.29\text{E}+01 \approx$
PSO	$1.43\text{E}+08 \pm 1.72\text{E}+07 +$	$3.57\text{E}+07 \pm 1.23\text{E}+06 -$	$1.55\text{E}+04 \pm 2.37\text{E}+01 \approx$	$1.92\text{E}+04 \pm 2.05\text{E}+02 +$	$3.29\text{E}+04 \pm 1.01\text{E}+02 -$
DE	$6.05\text{E}+07 \pm 3.89\text{E}+06 +$	$1.18\text{E}+08 \pm 1.06\text{E}+06 +$	$1.64\text{E}+04 \pm 8.57\text{E}+02 \approx$	$6.51\text{E}+05 \pm 1.11\text{E}+05 +$	$1.18\text{E}+08 \pm 6.68\text{E}+06 +$
GWO	$8.14\text{E}+06 \pm 1.52\text{E}+05 +$	$4.59\text{E}+07 \pm 1.00\text{E}+06 +$	$1.55\text{E}+04 \pm 1.94\text{E}+01 \approx$	$1.94\text{E}+04 \pm 2.09\text{E}+02 \approx$	$8.80\text{E}+07 \pm 1.54\text{E}+04 +$
Algorithm	$F_{11,6}$	$F_{11,7}$	$F_{11,8}$	$F_{11,9}$	$F_{11,10}$
CCWFSSE	$1.42\text{E}+05 \pm 4.36\text{E}+03$	$3.16\text{E}+08 \pm 7.72\text{E}+08$	$3.38\text{E}+07 \pm 8.48\text{E}+06$	$3.54\text{E}+07 \pm 9.57\text{E}+06$	$3.42\text{E}+07 \pm 7.12\text{E}+06$
WFS	$1.42\text{E}+05 \pm 2.96\text{E}+03 \approx$	$1.06\text{E}+08 \pm 1.99\text{E}+08 -$	$3.25\text{E}+07 \pm 8.15\text{E}+06 -$	$3.67\text{E}+07 \pm 9.24\text{E}+06 +$	$3.43\text{E}+07 \pm 9.25\text{E}+06 \approx$
PSO	$1.33\text{E}+05 \pm 2.69\text{E}+03 -$	$2.12\text{E}+08 \pm 1.65\text{E}+08 -$	$1.87\text{E}+06 \pm 4.53\text{E}+05 -$	$2.27\text{E}+06 \pm 4.53\text{E}+05 -$	$2.01\text{E}+06 \pm 3.46\text{E}+05 -$
DE	$6.68\text{E}+08 \pm 3.60\text{E}+06 +$	$8.45\text{E}+11 \pm 9.24\text{E}+08 +$	$1.74\text{E}+08 \pm 1.51\text{E}+07 +$	$1.68\text{E}+08 \pm 1.23\text{E}+07 +$	$1.74\text{E}+08 \pm 1.20\text{E}+07 +$
GWO	$6.05\text{E}+08 \pm 8.18\text{E}+05 +$	$8.10\text{E}+11 \pm 1.42\text{E}+09 +$	$3.38\text{E}+07 \pm 1.34\text{E}+06 +$	$3.39\text{E}+07 \pm 1.26\text{E}+06 -$	$3.38\text{E}+07 \pm 1.12\text{E}+06 -$
Algorithm	F_{12}	F_{13}	$w/t/l$		
CCWFSSE	$2.33\text{E}+01 \pm 3.28\text{E}+00$	$3.35\text{E}+01 \pm 4.63\text{E}+00$			
WFS	$2.28\text{E}+01 \pm 5.17\text{E}+00 -$	$3.59\text{E}+01 \pm 6.11\text{E}+00 +$	8/11/3		
PSO	$3.11\text{E}+01 \pm 4.82\text{E}+00 +$	$3.83\text{E}+01 \pm 5.70\text{E}+00 +$	12/3/7		
DE	$3.15\text{E}+01 \pm 5.41\text{E}+00 +$	$3.56\text{E}+01 \pm 4.75\text{E}+00 +$	19/3/0		
GWO	$2.41\text{E}+01 \pm 4.41\text{E}+00 +$	$3.46\text{E}+01 \pm 5.74\text{E}+00 +$	14/5/3

Fig. 6 The convergence graphs of WFSSE and other algorithms on F5, F7, F9, F16, F20, and F28

the iteration progresses, and the main problem lies in generating new forms or properties for the pattern matrix [81].

In CCWFSSE, the pattern matrix includes all the points that the vehicle can explore in one iteration. CCWFSSE eschews the traditional crossover, variational rules and instead generates new solutions by finding new neighborhoods. Through a fast screening mechanism of each

generation, CCWFSSE can theoretically find the global optimum of the objective function in exponential time. To sustain the diversity, the Halton sequence and grid logic are chosen during the process of initialization. In addition to the pseudo-random points in the specified neighborhood, random points and the center of mass are adopted at each iteration to advocate the diversity.

Fig. 7 The box-and-whisker diagrams on F5, F7, F9, F16, F20 and F28

6.3 Population Diversity Analysis

Population diversity is also a considerable parameter in algorithm evaluation. It can indicate whether an algorithm is prone to prematurely stalling, i.e., falling into local optimum [82]. This indicator Div can be calculated using Eq. (9):

$$Div = \frac{1}{N} \sqrt{\sum_{i=1}^N (\|X_i - X_{\text{mean}}\|)^2} \quad (9)$$

where N represents the size of population. X_i is the i -th individual. X_{mean} , which is defined as $X_{\text{mean}} = \frac{1}{N} \sum_{i=1}^N X_i$, is the average of the population.

However, owing to the particularity of the CCWFSSE mechanism, in the initial population of 30,000, relatively suitable possible solutions were quickly selected, and the neighborhood points were then generated based on the Halton sequence. Therefore, it is unfair for CCWFSSE to use common mechanisms when comparing its population diversity with other outstanding algorithms. The variation trends in six representative ones are compared between CCWFSSE and WFS, as shown in Fig. 9. CCWFSSE maintains a high level of diversity, especially in the middle iteration stage. As the iterative process goes on, the diversity of CCWFSSE remains stable at a high level. In the entire optimization process, we use SE's vector difference and cooperative coevolution to provide each individual with a driving force for diffusion, to expand a extensive

Fig. 8 Search history of individuals of CCWFSSE

search space. This manipulation affords more opportunities for individuals and ensures a higher level of population diversity throughout the search phase.

6.4 Time Complexity

In the above text, we have proved the effectiveness of the CCWFSSE on the benchmark functions. In the ending, we analyze the computational complexity of the algorithm. The

time taken to execute an algorithm is theoretically impossible to calculate, but we can speculate by the number of statements executed and the degree of program complexity. It can be a good reflection of the advantages and disadvantages of an algorithm. The main procedures of this analysis can be demonstrated as:

- (1) The process of population initialization may cost $O(n)$;

Fig. 9 Population diversity on F4, F9, F13, F14, F17, and F28

- (2) Generate extract point and update population require time complexity $O(n^2)$;
- (3) Generating neighborhood size has $O(n)$ complexity;
- (4) Selected point using the vector difference and spherical evolution strategy require $O(n^2)$ cost;
- (5) Updating the random point and centroid point needs $O(n)$

The execution time of other program segments which are independent of the problem size n can be denoted as $O(1)$.

To sum up, the computational complexity of CCWFSSE can be organized as:

$$T(N) = O(n) + T[O(n^2) + O(n) + O(n^2) + O(n)] \\ = (2T + 1)O(n) + 2TO(n^2) \quad (10)$$

We generally only need to keep the highest power of the calculated result. Therefore, the time complexity $T(N)$ of CCWFSSE is identified as $O(N^2)$, which is the same as that of WFS, but better than SE. It means there is still room for CCWFSSE to improve in designing program execution.

7 Conclusion

A novel hybrid algorithm termed CCWFSSE was presented in this article. This algorithm utilized the spherical search to improve the exploration ability, and cooperative coevolution to enhance the population diversity and exploitation ability. A sufficiently large test suite IEEE CEC2017 and CEC2011 real-world problems were chosen to assess various aspects of CCWFSSE's performance. CCWFSSE used the Halton sequence to generate points in the distribution space and used the vector difference and spherical search patterns to select and generate adjacent points to utilize the entire search space more effectively. To verify how well CCWFSSE actually performs, we chose WFS, PSO, DE, GWO and BOA as control objectives. Through the above experimental data, we proved that CCWFSSE has relatively good robustness and feasibility. Considering the structural complexity of CCWFSSE, specifically the fact that it was essentially parameter-free, the findings were encouraging, because CCWFSSE can successfully compete with related algorithms.

Although it was verified that CCWFSSE is an effective algorithm, it appeared to face difficulties when addressing complex and high-dimensional issues. In future work, we will focus on simplifying multiple diversity-driven strategies and on processing search information. Moreover, we can apply CCWFSSE to address more practical problems, such as multi-task learning [83], dynamic community detection [84], routing problem [85, 86], and multiple objective optimization [71, 87, 88].

Acknowledgements We thank the editor and anonymous reviewers for their constructive comments, which helped us to improve the manuscript.

Author Contributions JY: Methodology, Information query, Writing-Manuscript preparation. YZ: Specificadon, Data processing. ZW: Software, Validation. YT: Writing- Reviewing and Editing. BL: Writing-Reviewing and Editing, Validation. SG: Conceptualization, Resources, Supervision, Writing- Reviewing and Editing.

Funding This research was partially supported by the JSPS KAKENHI Grant Number JP19K12136, and Natural Science Foundation of Shanghai (No. 19ZR1402000).

Availability of data and material Related data and material can be found at <https://toyamaailab.github.io/>.

Declarations

Conflict of interest The authors declare that they have no conflicts of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long

as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Bonabeau, E., Dorigo, M., Theraulaz, G.: Inspiration for optimization from social insect behaviour. *Nature* **406**(6791), 39–42 (2000)
2. Shang, X., Shen, D., Wang, F.-Y., Nyberg, T.R.: A heuristic algorithm for the fabric spreading and cutting problem in apparel factories. *IEEE/CAA J. Autom. Sin.* **6**(4), 961–968 (2019)
3. Han, F., Qi-Shao, L.: An improved chaos optimization algorithm and its application in the economic load dispatch problem. *Int. J. Comput. Math.* **85**(6), 969–982 (2008)
4. Dokeroglu, T., Sevinc, E., Kucukyilmaz, T., Cosar, A.: A survey on new generation metaheuristic algorithms. *Comput. Ind. Eng.* **137**, 106040 (2019)
5. Del Javier, S., Eneko, O., Daniel, M., Xin-She, Y., Sancho, S.-S., David, C., Swagatam, D., Ponnuthurai, N.S., Carlos, A.C.C., Francisco, H.: Where we stand and what's next: Bio-inspired computation. *Swarm Evol. Comput.* **48**, 220–250 (2019)
6. Boussaid, I., Lepagnot, J., Siarry, P.: A survey on optimization metaheuristics. *Inf. Sci.* **237**, 82–117 (2013)
7. Zhao, D., Dai, Y., Zhang, Z.: Computational intelligence in urban traffic signal control: A survey. *IEEE Trans. Syst. Man Cybern. Part C* **42**(4), 485–494 (2011)
8. Akbar, T., Amir, H., Asadollah, S.: A survey of evolutionary computation for association rule mining. *Inf. Sci.* **524**, 318–352 (2020)
9. Wang, Y., Yang, Y., Cao, S., Zhang, X., Gao, S.: A review of applications of artificial intelligent algorithms in wind farms. *Artif. Intell. Rev.* **53**(5), 3447–3500 (2020)
10. Gao, S., Yang, Y., Wang, Y., Wang, J., Cheng, J., Zhou, M.C.: Chaotic local search-based differential evolution algorithms for optimization. *IEEE Trans. Syst. Man Cybern. Syst.* **51**(6), 3954–3967 (2021)
11. Lei, Z., Gao, S., Zhang, Z., Zhou, M.C., Cheng, J.: MO4: A many-objective evolutionary algorithm for protein structure prediction. *IEEE Trans. Evol. Comput.* (2021). <https://doi.org/10.1109/TEVC.2021.3095481>
12. Rafael, S.P., Heitor, S.L.: New inspirations in swarm intelligence: A survey. *Int. J. Bio-Inspir. Comput.* **3**(1), 1–16 (2011)
13. Neri, F., Cotta, C.: Memetic algorithms and memetic computing optimization: A literature review. *Swarm Evol. Comput.* **2**, 1–14 (2012)
14. Li, Q., Liu, S.-Y., Yang, X.-S.: Influence of initialization on the performance of metaheuristic optimizers. *Appl. Soft Comput.* **91**, 106193 (2020)
15. Lones, M.A.: Mitigating metaphors: A comprehensible guide to recent nature-inspired algorithms. *SN Comput. Sci.* **1**(1), 49 (2020)
16. Wang, P., Zhou, Y., Luo, Q., Han, C., Niu, Y., Lei, M.: Complex-valued encoding metaheuristic optimization algorithm: A comprehensive survey. *Neurocomputing* **407**, 313–342 (2020)
17. Yang, X.-S.: Nature-inspired optimization algorithms: Challenges and open problems. *J. Comput. Sci.* **46**, 101104 (2020)
18. Shi, W., Xiao, Y., Zonghui, C., Lin, Z., Shangge, G.: An improved firefly algorithm enhanced by negatively correlated search

- mechanism. In: 2018 IEEE International Conference on Progress in Informatics and Computing (PIC), IEEE, pp. 67–72 (2018)
19. Pratik, R., Ghanshaym, S.M., Kashi, N.D.: Forecasting of software reliability using neighborhood fuzzy particle swarm optimization based novel neural network. *IEEE/CAA J. Autom. Sin.* **6**(6), 1365–1383 (2019)
 20. Guangxiao, S., Zhijie, W., Fang, H., Shenyi, D., Muhammad, A.I.: Music auto-tagging using deep recurrent neural networks. *Neurocomputing* **292**, 104–110 (2018)
 21. Khan, B., Han, F., Wang, Z., Masood, R.J.: Bio-inspired approach to invariant recognition and classification of fabric weave patterns and yarn color. *Assem. Autom.* **36**(2), 152–158 (2016)
 22. Pham, D.T., Ghanbarzadeh, A., Koç, E., Otri, S., Rahim, S., Zaidi, M.: The bees algorithm-a novel tool for complex optimisation problems. In: *Intelligent Production Machines and Systems*, pp. 454–459. Elsevier, Amsterdam (2008)
 23. Karaboga, D., Akay, B.: A comparative study of artificial bee colony algorithm. *Appl. Math. Comput.* **214**(1), 108–132 (2009)
 24. Ji, J., Song, S., Tang, C., Gao, S., Tang, Z., Todo, Y.: An artificial bee colony algorithm search guided by scale-free networks. *Inf. Sci.* **473**, 142–165 (2019)
 25. Wang, Y., Yang, Y., Gao, S., Pan, H., Yang, G.: A hierarchical gravitational search algorithm with an effective gravitational constant. *Swarm Evol. Comput.* **46**, 118–139 (2019)
 26. Lei, Z., Gao, S., Gupta, S., Cheng, J., Yang, G.: An aggregative learning gravitational search algorithm with self-adaptive gravitational constants. *Expert Syst. Appl.* **152**, 113396 (2020)
 27. Hang, Y., Zhe, X., Shangce, G., Yirui, W., Yuki, T.: PMPSO: a near-optimal graph planarization algorithm using probability model based particle swarm optimization. In: 2015 IEEE International Conference on Progress in Informatics and Computing (PIC), IEEE, pp. 15–19 (2015)
 28. Yue-Jiao, G., Jing-Jing, L., Yicong, Z., Yun, L., Henry, S.-H.C., Yu-Hui, S., Jun, Z.: Genetic learning particle swarm optimization. *IEEE Trans. Cybern.* **46**(10), 2277–2290 (2015)
 29. Sun, J., Gao, S., Dai, H., Cheng, J., Zhou, M.C., Wang, J.: Bi-objective elite differential evolution for multivalued logic networks. *IEEE Trans. Cybern.* **50**(1), 233–246 (2020)
 30. Tang, Y., Ji, J., Zhu, Y., Gao, S., Tang, Z., Todo, Y.: A differential evolution-oriented pruning neural network model for bankruptcy prediction. *Complexity* **2019**, 8682124 (2019)
 31. Gao, S., Wang, Y., Wang, J., Cheng, J.J.: Understanding differential evolution: A Poisson law derived from population interaction network. *J. Comput. Sci.* **21**, 140–149 (2017)
 32. Yu, Y., Yang, L., Wang, Y., Gao, S.: Brain storm algorithm combined with covariance matrix adaptation evolution strategy for optimization. In: *Brain Storm Optimization Algorithms*, pp. 123–154. Springer, Berlin (2019)
 33. Wang, Y., Gao, S., Yang, Y., Zhe, X.: The discovery of population interaction with a power law distribution in brain storm optimization. *Memet. Comput.* **11**(1), 65–87 (2019)
 34. Gong, Y.-J., Chen, W.-N., Zhan, Z.-H., Zhang, J., Li, Y., Zhang, Q., Li, J.-J.: Distributed evolutionary algorithms and their models: A survey of the state-of-the-art. *Appl. Soft Comput.* **34**, 286–300 (2015)
 35. Cheng, J., Yuan, G., Zhou, M., Gao, S., Liu, C., Duan, H., Zeng, Q.T.: Accessibility analysis and modeling for IoV in an urban scene. *IEEE Trans. Veh. Technol.* **69**(4), 4246–4256 (2020)
 36. Wang, J., Yuan, L., Zhang, Z., Gao, S., Sun, Y., Zhou, Y.: Multiobjective multiple neighborhood search algorithms for multi-objective fleet size and mix location-routing problem with time windows. *IEEE Trans. Syst. Man Cybern. Syst.* **51**(4), 2284–2298 (2021)
 37. Wang, S., Yang, Y., Zou, L., Li, S., Hang, Y., Todo, Y., Gao, S.: A novel median dendritic neuron model for prediction. *IEEE Access* **8**, 192339–192351 (2020)
 38. Shreya, P., Anil, K., Varun, B., Girish, K.S.: A context sensitive multilevel thresholding using swarm based algorithms. *IEEE/CAA J. Autom. Sin.* **6**(6), 1471–1486 (2017)
 39. Sheng-Yong, D., Liu, Z.-G.: Hybridizing particle swarm optimization with jade for continuous optimization. *Multimed. Tools Appl.* **79**(7), 4619–4636 (2020)
 40. Li, X., Cai, Z., Wang, Y., Todo, Y., Cheng, J., Gao, S.: TDSD: A new evolutionary algorithm based on triple distinct search dynamics. *IEEE Access* **8**, 76752–76764 (2020)
 41. Črepinšek, M., Liu, S.-H., Mernik, M.: Exploration and exploitation in evolutionary algorithms: A survey. *ACM Comput. Surv. (CSUR)* **45**(3), 1–33 (2013)
 42. Jia, D., Tong, Y., Yu, Y., Cai, Z., Gao, S.: A novel backtracking search with grey wolf algorithm for optimization. In: 2018 10th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), IEEE, pp. 73–76 (2018)
 43. Yang, Y., Gao, S., Wang, Y., Lei, Z., Cheng, J., Todo, Y.: A multiple diversity-driven brain storm optimization algorithm with adaptive parameters. *IEEE Access* **7**, 126871–126888 (2019)
 44. Elbeltagi, E., Hegazy, T., Grierson, D.: Comparison among five evolutionary-based optimization algorithms. *Adv. Eng. Inf.* **19**(1), 43–53 (2005)
 45. Peng, X., Wang, Z., Han, F., Song, G., Ding, S.: A novel time-event-driven algorithm for simulating spiking neural networks based on circular array. *Neurocomputing* **292**, 121–129 (2018)
 46. Gao, S., Wang, K., Tao, S., Jin, T., Dai, H., Cheng, J.: A state-of-the-art differential evolution algorithm for parameter estimation of solar photovoltaic models. *Energy Convers. Manag.* **230**, 113784 (2021)
 47. Zhu, Q., Shen, J., Han, F., Wenlian, L.: Bifurcation analysis and probabilistic energy landscapes of two-component genetic network. *IEEE Access* **8**, 150696–150708 (2020)
 48. Fang, H., Zhi-Jie, W., Hong, F., Tao, G.: Robust synchronization in an E/I network with medium synaptic delay and high level of heterogeneity. *Chin. Phys. Lett.* **32**(4), 040502 (2015)
 49. Anguluri, R., Nandar, L., Swagatam, D., Ponnuthurai, N.S.: Computing with the collective intelligence of honey bees-a survey. *Swarm Evol. Comput.* **32**, 25–48 (2017)
 50. Gao, K., Cao, Z., Zhang, L., Chen, Z., Han, Y., Pan, Q.: A review on swarm intelligence and evolutionary algorithms for solving flexible job shop scheduling problems. *IEEE/CAA J. Autom. Sin.* **6**(4), 904–916 (2019)
 51. Choi, K., Jang, D.-H., Kang, S.-I., Lee, J.-H., Chung, T.-K., Kim, H.-S.: Hybrid algorithm combining genetic algorithm with evolution strategy for antenna design. *IEEE Trans. Magn.* **52**(3), 1–4 (2015)
 52. Yang, Y., Gao, S., Wang, Y., Todo, Y.: Global optimum-based search differential evolution. *IEEE/CAA J. Autom. Sin.* **6**(2), 379–394 (2018)
 53. Piotrowski, A.P.: Review of differential evolution population size. *Swarm Evol. Comput.* **32**, 1–24 (2017)
 54. Wang, Y., Gao, S., Yang, Y., Cai, Z., Wang, Z.: A gravitational search algorithm with hierarchy and distributed framework. *Knowl. Based Syst.* **218**, 106877 (2021)
 55. Wu, D., Junjie, X., Xiao-Zhi, G., Huimin, Z.: An enhanced msiqde algorithm with novel multiple strategies for global optimization problems. *IEEE Trans. Syst. Man Cybern. Syst.* (2020). <https://doi.org/10.1109/TSMC.2020.3030792>
 56. Jing-Ru, Z., Jun, Z., Tat-Ming, L., Michael, R.L.: A hybrid particle swarm optimization-back-propagation algorithm for feed-forward neural network training. *Appl. Math. Comput.* **185**(2), 1026–1037 (2007)
 57. Shunmugapriya, P., Kanmani, S.: A hybrid algorithm using ant and bee colony optimization for feature selection and classification (ac-abc hybrid). *Swarm Evol. Comput.* **36**, 27–36 (2017)

58. Knowles, J.: Parego: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Trans. Evol. Comput.* **10**(1), 50–66 (2006)
59. Hasan, M.J.A., Ramakrishnan, S.: A survey: Hybrid evolutionary algorithms for cluster analysis. *Artif. Intell. Rev.* **36**(3), 179–204 (2011)
60. Gao, S., Wang, W., Dai, H., Li, F., Tang, Z.: Improved clonal selection algorithm combined with ant colony optimization. *IEICE Trans. Inf. Syst.* **91**(6), 1813–1823 (2008)
61. Poonam, S., Ramdevsinh, L.J., Vimal, S.: Effect of hybridizing biogeography-based optimization (bbo) technique with artificial immune algorithm (aia) and ant colony optimization (aco). *Appl. Soft Comput.* **21**, 542–553 (2014)
62. Zhe, X., Wang, Y., Li, S., Liu, Y., Todo, Y., Gao, S.: Immune algorithm combined with estimation of distribution for traveling salesman problem. *IEEJ Trans. Electr. Electron. Eng.* **11**, S142–S154 (2016)
63. Sun, L., Lin, L., Gen, M., Li, H.: A hybrid cooperative coevolution algorithm for fuzzy flexible job shop scheduling. *IEEE Trans. Fuzzy Syst.* **27**(5), 1008–1022 (2019)
64. Yang, Z., Tang, K., Yao, X.: Large scale evolutionary optimization using cooperative coevolution. *Inf. Sci.* **178**(15), 2985–2999 (2008)
65. Covic, N., Lacevic, B.: Wingsuit flying search- a novel global optimization algorithm. *IEEE Access* **8**, 53883–53900 (2020)
66. Tang, D.: Spherical evolution for solving continuous optimization problems. *Appl. Soft Comput.* **81**, 105499 (2019)
67. Chakkarapani, M., Guru, R.R., Guru, P.R., Saravana, I.G., Chilakapati, N.: A hybrid algorithm for tracking of GMPP based on P&O and pso with reduced power oscillation in string inverters. *IEEE Trans. Ind. Electron.* **63**(10), 6097–6106 (2016)
68. Wang, G.-G., Tan, Y.: Improving metaheuristic algorithms with information feedback models. *IEEE Trans. Cybern.* **49**(2), 542–555 (2017)
69. Jacinto, C., Salvador, G., Rueda, M.M., Das, S., Francisco, H.: Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: Practical guidelines and a critical review. *Swarm Evol. Comput.* **54**, 100665 (2020)
70. Khalilpourazari, S., Khalilpourazary, S.: An efficient hybrid algorithm based on water cycle and moth-flame optimization algorithms for solving numerical and constrained engineering optimization problems. *Soft Comput.* **23**(5), 1699–1722 (2019)
71. Wang, J., Cen, B., Gao, S., Zhang, Z., Zhou, Y.: Cooperative evolutionary framework with focused search for many-objective optimization. *IEEE Trans. Emerg. Top. Comput. Intell.* **4**(3), 398–412 (2020)
72. Mirjalili, S., Mirjalili, S.M., Lewis, A.: Grey wolf optimizer. *Adv. Eng. Softw.* **69**, 46–61 (2014)
73. De Ivanoe, F., Antonio, D.C., Giuseppe, A.T.: Investigating surrogate-assisted cooperative coevolution for large-scale global optimization. *Inf. Sci.* **482**, 1–26 (2019)
74. Fan, G.-F., Qing, S., Wang, H., Hong, W.-C., Li, H.-J.: Support vector regression model based on empirical mode decomposition and auto regression for electric load forecasting. *Energies* **6**(4), 1887–1901 (2013)
75. Chen, Y.H., Hong, W.-C., Shen, W., Huang, N.N.: Electric load forecasting based on a least squares support vector machine with fuzzy time series and global harmony search algorithm. *Energies* **9**(2), 70 (2016)
76. Li, M.-W., Wang, Y.-T., Geng, J., Hong, W.-C.: Chaos cloud quantum bat hybrid optimization algorithm. *Nonlinear Dyn.* **103**(1), 1167–1193 (2021)
77. Arora, S., Singh, S.: Butterfly optimization algorithm: A novel approach for global optimization. *Soft Comput.* **23**(3), 715–734 (2019)
78. Gao, S., Wang, R.-L., Ishii, M., Tang, Z.: An artificial immune system with feedback mechanisms for effective handling of population size. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.* **93**(2), 532–541 (2010)
79. Yang, Y., Dai, H., Gao, S., Wang, Y., Jia, D., Tang, Z.: Complete receptor editing operation based on quantum clonal selection algorithm for optimization problems. *IEEJ Trans. Electr. Electron. Eng.* **14**(3), 411–421 (2019)
80. Wang, Y., Gao, S., Yang, Y., Wang, Z., Cheng, J., Yuki, T.: A gravitational search algorithm with chaotic neural oscillators. *IEEE Access* **8**, 25938–25948 (2020)
81. Wang, Y., Gao, S., Zhou, M.C., Yang, Y.: A multi-layered gravitational search algorithm for function optimization and real-world problems. *IEEE/CAA J. Autom. Sin.* **8**(1), 94–109 (2021)
82. Yang, Y., Gao, S., Cheng, S., Wang, Y., Song, S., Yuan, F.: CBSO: a memetic brain storm optimization with chaotic local search. *Memet. Comput.* **10**(4), 353–367 (2017)
83. Gao, S., Zhou, M.C., Wang, Y., Cheng, J., Yachi, H., Wang, J.: Dendritic neural model with effective learning algorithms for classification, approximation, and prediction. *IEEE Trans. Neural Netw. Learn. Syst.* **30**(2), 601–604 (2019)
84. Jiu, J.C., Gui, Y.Y., Meng, C.Z., Shangce, G., Zhen, H.H., Cong, L.: A connectivity prediction-based dynamic clustering model for VANET in an urban scene. *IEEE Internet Things J* **7**(9), 8410–8418 (2020)
85. Cheng, J.J., Cheng, J.L., Zhou, M.C., Liu, F.Q., Gao, S.C., Liu, C.: Routing in internet of vehicles: A review. *IEEE Trans. Intell. Transp. Syst.* **16**(5), 2339–2352 (2015)
86. Cheng, J., Yuan, G., Zhou, M.C., Gao, S., Liu, C., Duan, Hua: A fluid mechanics-based data flow model to estimate VANET capacity. *IEEE Trans. Intell. Transp. Syst.* **21**(6), 2603–2614 (2019)
87. Wang, J., Sun, Y., Zhang, Z., Gao, S.: Solving multitrip pickup and delivery problem with time windows and manpower planning using multiobjective algorithms. *IEEE/CAA J. Autom. Sin.* **7**(4), 1134–1153 (2020)
88. He, C., Li, L., Tian, Y., Zhang, X., Cheng, R., Jin, Y., Yao, X.: Accelerating large-scale multiobjective optimization via problem reformulation. *IEEE Trans. Evol. Comput.* **23**(6), 949–961 (2019)

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