IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 21 February 2024, accepted 22 March 2024, date of publication 10 April 2024, date of current version 17 April 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3387308

RESEARCH ARTICLE

The Method to Integrate Species Explode and Deracinate Algorithm With Particle Swarm Optimization Algorithm

YONGJIAN YANG, YOUWEI DENG, BINGSONG XIAO, AND XIAOHONG ZHAO
Aviation Engineering School, Air Force Engineering University, Xi'an 710038, China

Corresponding author: Yongjian Yang (yangyongjian_king@126.com)

This work was supported in part by the National Key Research and Development Program of China under Grant 2021YFA0715400, and in part by the National Natural Science Foundations of China under Grant 62104260.

ABSTRACT Effectively combining various evolutionary computing algorithms and leveraging the advantages of each can significantly enhance the convergence speed and solution quality of the algorithm. However, a mere combination of evolutionary computing algorithms may not comprehensively improve optimization performance and may even lead to poorer performance in certain optimization problems. The aim of the paper is to provide a fundamental integrating platform and method based on species explode and deracinate algorithm. Utilizing the species explode and deracinate algorithm as a foundation, this study presents a hybrid algorithm named SED-PSO algorithm by utilizing the particle swarm optimization algorithm as an exemplar. The outcomes of the simulations conducted on 27 benchmark functions published by the Competition on Evolutionary Constrained demonstrate that the SED-PSO algorithm exhibits exceptional convergence accuracy, robust stability, and rapid convergence speed. The simulation results comprehensively illustrate that the species explode and deracinate algorithm serves as a fundamental integrating platform for diverse evolutionary computing algorithms, while also incorporating the strengths of each algorithm. Additionally, the outcomes of the optimization of sensor network coverage reveal that the SED-PSO algorithm exhibits superior solution quality, minimal occurrence of local extremum, and enhanced stability and efficacy.

INDEX TERMS Evolutionary computing algorithm, species explode and deracinate algorithm, particle swarm optimization algorithm, competition on evolutionary constrained, optimization of sensor network coverage.

I. INTRODUCTION

Evolutionary computation (EC) [1], [2] is well suited to solving complex optimization problems. Such evolutionary computing technologies are widely used to solve complex optimization problems as genetic algorithm (GA) [3], particle swarm optimization (PSO) algorithm [4], differential evolution (DE) [5], ant colony optimization (ACO) [6], [7]. The randomness is the root of the simple theory and flexible design of EC algorithms, but it also makes EC algorithms non-analytic and the optimization results are uncertain. To date, there is no one EC algorithm has been able to comprehensively outperform the other EC algorithms in all CEC benchmark functions [8], [9]. However, it is still the goal of researchers to improve the performance of EC algorithms, and many improved and new EC algorithms have been proposed successively, such as improved GA algorithms [10], [11], improved PSO algorithms [12], [13], species explode and deracinate (SED) algorithm [14] and so on.

Compared with other EC algorithms, PSO algorithm has been proved to be easy to implement, and has the characteristics of fast convergence and high convergence accuracy. There are a number of improved PSO algorithms have been proposed [15], [16], [17], [18], [19], [20], [21]. These

The associate editor coordinating the review of this manuscript and approving it for publication was P. K. Gupta.

improvement methods mainly include four categories: (1) the methods to set the parameters of PSO algorithm [22], [23], (2) the methods to integrating of PSO algorithm with other EC algorithms [24], [25], (3) the methods to improving the topology structure of PSO algorithm [12], [19] and (4) the methods to combining of control theory and PSO algorithm [26]. These improved methods can effectively improve the optimization performance of the basic PSO algorithm. However, the optimization performance of a single improved method is not overall superior in all CEC benchmark functions. Even if the different improved methods are combined, the performance of PSO algorithm may not be further improved, even worsen.

Not only the improved PSO algorithms have above problems, but also the improved methods of other EC algorithms show such same problems as: (1) each EC algorithm has its own advantages, but no algorithm can comprehensively outperform other EC algorithms; (2) each improved EC algorithms can effectively improve the performance of the basic algorithm, but none of them can comprehensively outperform other improved methods; (3) the simple combination of various EC algorithms or the combination of improved methods of each EC algorithm cannot comprehensively improve the optimization performance of the algorithm, and the performance even worse in some optimization problems.

If there is a method that can effectively integrate various EC algorithms and give full play to their advantages, the problem that different EC algorithms have different performance in various optimization problems can be solved. Based on the explode and deracinate of species in the evolutionary history, we have proposed a new EC algorithm, named SED algorithm which is likely to integrate various EC algorithms and take advantage of the advantages of each EC algorithm.

The SED algorithm mainly consists of two operation stages. The first stage is species exploding. On the basis of existing species, the number of species reaches M times by derivation. The second stage is species deracinating. After the species exploding, the number of species are decreased, and the optimal species are retained and the rest are eliminated. SED algorithm has the following characteristics: (1) Simplicity. SED algorithm requires less configuration parameters which can be found in TABLE 2, and is easy to implement. (2) Global optimal. SED algorithm can effectively avoid falling into local extremum. (3) High efficiency. SED algorithm can quickly find the global optimal solution, and the solution accuracy is high. The above characteristics of SED algorithm can also be clearly found in the subsequent simulation results of this paper.

To reduce the computational burden, we only integrate SED algorithm with the basic PSO algorithm which named SED-PSO algorithm in this paper. And the feasibility and superior performance of SED algorithm integrated with other EC algorithms will be illustrated by SED-PSO algorithm.

The organization of this paper is as follows. In section II, the basic principle of SED algorithm, PSO algorithm and ECE-PSO (explorative capability enhancement in PSO) algorithm [12] are introduced. In section III, the method of integrating SED algorithm with PSO algorithm are specifically discussed. In section IV, the performance of SED-PSO algorithm is validated on CEC benchmark functions and sensors deployment applications.

II. RELATED WORK

A. SED ALGORITHM

SED algorithm [14] which is based on the species exploding and deracinating phenomena found in the evolutionary history of organisms and the idea of species catastrophe evolution theory, is a new EC algorithm. By species exploding and deracinating, SED algorithm can find the optimal result. A balance will be achieved between the global optimum and the local extremum by introducing the strategies of main branch transfer and species derivative ability shrink.

In SED algorithm, each iteration includes the exploding and deracinating operations of all species, and the maximal iteration is denoted as $FE_{\max}$ in this paper. Assuming the dimension of the search area is D, and the number of surviving species is S. $\boldsymbol{X}_s = (\boldsymbol{x}_{s,1}, \boldsymbol{x}_{s,2}, \ldots, \boldsymbol{x}_{s,D})$, denotes the s-th species, and $1 \leq s \leq S$. The fitness value of the s-th species is denoted as $Y_s = f(\boldsymbol{X}_s)$. Function f is the fitness function.

$A \in [A_{\min}, A_{\max}]$ represents the derivative ability of species, $A_{\max}$ represents the maximum derivative ability of species, and $A_{\min}$ represents the minimum derivative ability of species. Generally, $A_{\max} < L_{\max}$, $A_{\min} > L_{\min}$ and $A_{\min} > 0$. $[L_{\min}, L_{\max}]$ represents the search interval.

1) SPECIES EXPLODING

Species exploding means the birth of a large number of new species. In SED algorithm, species exploding is a derivation function, and each surviving species produces M ($M > 0$) new species, that is, exploding of M times. A new species derived from the s-th species is denoted $\boldsymbol{XP}_s$, and its derivation function is as follows.

$$\boldsymbol{XP}_s^{k,j} = \boldsymbol{X}_s^k + \boldsymbol{rand} \times A_s^k \tag{1}$$

$$AP_s^{k,j} = \frac{A_{\max} - A_{\min}}{G_s + 1} \tag{2}$$

where $\boldsymbol{XP}_s^{k,j}$ denotes the new species derived from the s-th species in the j-th exploding at the k-th iteration, and $1 \leq s \leq S$, $1 \leq j \leq M$, $1 \leq k \leq FE_{\max}$. $\boldsymbol{X}_s^k$ denotes the s-th surviving species in the k-th iteration. $\boldsymbol{rand}$ is a random vector between (0, 1) with the same dimension as $\boldsymbol{X}_s^k$. A_s^k represents the derivative ability of the s-th species in the k-th iteration. $AP_s^{k,j}$ represents the derivative ability of the new species derived from the s-th species in the j-th exploding at the k-th iteration. G_s is the number of generations of the s-th surviving species.

The species that meet the requirements of new species derivation called the main branch species. Only the main branch species can derive new species. After each exploding, the fitness value of the new species will be compared with the main branch species. If the fitness value of the new species is better, the new species will be labeled as the main branch

TABLE 1. 27 benchmark functions published by CEC.

Name	Benchmark functions	Search range $[L_{min}, L_{max}]$	Optimal	Global minimum	Error goal
Sphere	$f_1(x)=\sum_{d=1}^{D}x_d^2$	$[-100, 100]^{30}$	$[0]^{30}$	0	1e-5
Elliptic	$f_2(x)=\sum_{d=1}^{D}[(10)^6]^{\frac{d-1}{D-1}}x_d^2$	$[-100, 100]^{30}$	$[0]^{30}$	0	1e-8
Sum squares	$f_3(x)=\sum_{d=1}^{D}dx_d^2$	$[-10, 10]^{30}$	$[0]^{30}$	0	1e-8
Sum Powers	$f_4(x)=\sum_{d=1}^{D}\lvert x_d\rvert^{(d+1)}$	$[-1, 1]^{30}$	$[0]^{30}$	0	1e-8
Schwefel's P2.22	$f_5(x)=\sum_{d=1}^{D}\lvert x_d\rvert+\prod_{d=1}^{D}\lvert x_d\rvert$	$[-100, 100]^{30}$	$[0]^{30}$	0	1e-8
Step	$f_6(x)=\sum_{d=1}^{D}(\lfloor x_d+0.5\rfloor)^2$	$[-100, 100]^{30}$	$0.5\leq x_d<0.5$	0	1e-8
Quadric	$f_7(x)=\sum_{d=1}^{D}\left(\sum_{j=1}^{d}x_j\right)^2$	$[-100, 100]^{30}$	$[0]^{30}$	0	1e-6
Noise	$f_8(x)=\sum_{d=1}^{D}d(x_d)^2+\mathrm{rand}[0,1)$	$[-1.28, 1.28]^{30}$	$[0]^{30}$	0	1e-2
Hyper Ellipsoid	$f_9(x)=\sum_{d=1}^{D}\sum_{j=1}^{d}(x_d)^2$	$[-100, 100]^{30}$	$[0]^{30}$	0	1e-8
Rosenbrock	$f_{10}(x)=\sum_{d=1}^{D-1}\left[100\left(x_{d+1}-x_d^2\right)^2+(x_d-1)^2\right]$	$[-10, 10]^{30}$	$[1]^{30}$	0	1e2
Griewank	$f_{11}=\frac{1}{4000}\sum_{d=1}^{D}x_d^2-\prod_{d=1}^{D}\cos(\frac{x_d}{\sqrt{d}})+1$	$[-600, 600]^{30}$	$[0]^{30}$	0	1e-1
Generalized penalizede	$f_{12}=0.1\left\{\sin^2(3\pi x_1)+\sum_{d=1}^{D-1}(x_d-1)^2\left[1+\sin^2(3\pi x_{d+1})\right]+(x_D-1)^2\left[1+\sin^2(2\pi x_D)\right]\right\}+\sum_{d=1}^{D}u(x_d,5,100,4)$ where, $u(x_d,a,k,m)=\begin{cases}k(x_d-a)^m, & x_d>a\\ 0, & -a\leq x_d\leq a\\ k(-x_d-a)^m, & x_d<-a\end{cases}$	$[-32.768, 32.768]^{30}$	$[1]^{30}$	0	1e1
Levy	$f_{13}=\sin^2(3\pi x_1)+\sum_{d=1}^{D-1}(x_d-1)^2\left[1+\sin^2(3\pi x_{d+1})\right]+\lvert x_D-1\rvert\left[1+\sin^2(3\pi x_D)\right]$	$[-10, 10]^{30}$	$[1]^{30}$	0	1e1
Himmelblau	$f_{14}=\frac{1}{D}\sum_{d=1}^{D}\left(x_d^4-16x_d^2+5x_d\right)$	$[-5, 5]^{30}$	NA	-78.3324	-70
Alpine	$f_{15}=\sum_{d=1}^{D}\lvert x_d\sin(x_d)+0.1x_d\rvert$	$[-10, 10]^{30}$	$[0]^{30}$	0	1e-8
Weierstrass	$f_{16}=\sum_{d=1}^{D}\left\{\sum_{k=1}^{k_{max}}a^k\cos\left[2\pi b^k(x_d+0.5)\right]\right\}-D\sum_{k=1}^{k_{max}}a^k\cos(2\pi b^k 0.5)$ where, $a=0.5,\ b=3,\ k_{max}=20$	$[-0.5, 0.5]^{30}$	$[0]^{30}$	0	1e-8
NCRastrigin	$f_{17}=\sum_{d=1}^{D}\left[y_d^2-10\cos(2\pi y_d)+10\right],\ y_d=\begin{cases}x_d, & \lvert x_d\rvert<0.5\\ round(2x_d)/2, & \lvert x_d\rvert\geq 0.5\end{cases}$	$[-5.12, 5.12]^{30}$	$[0]^{30}$	0	1e-8
Michalewics	$f_{18}=-\sum_{d=1}^{D}\sin x_d\sin^{20}\frac{dx_d^2}{\pi}$	$[0, \pi]^{100}$	NA	-99.2784, D=100	-60
Schwefel	$f_{19}=-418.9829D-\sum_{d=1}^{D}x_d\sin\left(\sqrt{\lvert x_d\rvert}\right)$	$[-500, 500]^{30}$	$[420.9687]^{30}$	0	1e4
Rastrigin	$f_{20}=\sum_{d=1}^{D}\left[x_d^2-10\cos(2\pi x_d)+10\right]$	$[-5.12, 5.12]^{30}$	$[0]^{30}$	0	1e2
Ackley	$f_{21}=-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{d=1}^{D}x_d^2}\right)-\exp\left(\frac{1}{D}\sum_{d=1}^{D}\cos(2\pi x_d)\right)+20+e$	$[-32, 32]^{30}$	$[0]^{30}$	0	1e-8
Shifted Sphere	$f_{22}(x)=\sum_{d=1}^{D}z_d^2-450,\ \boldsymbol{z}=\boldsymbol{x}-\boldsymbol{O},\boldsymbol{x}=[x_1,x_2,\mathrm{L}\ ,x_d],\boldsymbol{O}=[o_1,o_2,\mathrm{L}\ ,o_d]$	$[-100, 100]^{30}$	$\boldsymbol{O}$	-450	-350
Shifted Schwefel Problem 1.2	$f_{23}(x)=\sum_{d=1}^{D}\sum_{j=1}^{d}z_j^2-450,\ \boldsymbol{z}=\boldsymbol{x}-\boldsymbol{O},\boldsymbol{x}=[x_1,x_2,\mathrm{L}\ ,x_d],\boldsymbol{O}=[o_1,o_2,\mathrm{L}\ ,o_d]$	$[-100, 100]^{30}$	$\boldsymbol{O}$	-450	-350

TABLE 1. *(Continued.)* **27 benchmark functions published by CEC.**

Name	Benchmark functions	Search range $[L_{\min}, L_{\max}]$	Optimal	Global minimum	Error goal
Shifted Rosenbrock	$f_{24}(x)=\sum_{d=1}^{D-1}\left[100\left(z_{d+1}-z_d^2\right)^2+\left(z_d-1\right)^2\right]+390$ $z=\boldsymbol{x}-\boldsymbol{O}+1, \boldsymbol{x}=[x_1,x_2,\mathrm{L}\ ,x_d], \boldsymbol{O}=[o_1,o_2,\mathrm{L}\ ,o_d]$	$[-100, 100]^{30}$	$\boldsymbol{O}$	390	490
Shifted Rastrigin	$f_{25}(x)=\sum_{d=1}^{D}\left[z_d^2-10\cos(2\pi z_d)+10\right]-330$ $z=\boldsymbol{x}-\boldsymbol{O}, \boldsymbol{x}=[x_1,x_2,\mathrm{L}\ ,x_d], \boldsymbol{O}=[o_1,o_2,\mathrm{L}\ ,o_d]$	$[-5, 5]^{30}$	$\boldsymbol{O}$	-330	-230
Shifted Ackley	$f_{26}(x)=-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{d=1}^{D}z_d^2}\right)-\exp\left(\frac{1}{D}\sum_{d=1}^{D}\cos(2\pi z_d)\right)+20+e-140$ $z=\boldsymbol{x}-\boldsymbol{O}, \boldsymbol{x}=[x_1,x_2,\mathrm{L}\ ,x_d], \boldsymbol{O}=[o_1,o_2,\mathrm{L}\ ,o_d]$	$[-32, 32]^{30}$	$\boldsymbol{O}$	-140	-135
Shifted Griewank	$f_{27}(x)=\frac{1}{4000}\sum_{d=1}^{D}z_d^2-\prod_{d=1}^{D}\cos(\frac{z_d}{\sqrt{d}})+1-180$ $z=\boldsymbol{x}-\boldsymbol{O}, \boldsymbol{x}=[x_1,x_2,\mathrm{L}\ ,x_d], \boldsymbol{O}=[o_1,o_2,\mathrm{L}\ ,o_d]$	$[-600, 600]^{30}$	$\boldsymbol{O}$	-180	-170

TABLE 2. **Parameters setting of four algorithms.**

Algorithm	Parameters
PSO	N=30, ω=0.724, c_1=c_2=2.
SED	M=20, S=30, α=0.05.
ECE-PSO	N=30, ω=0.724, c_1=c_2=1.468, P=5, *Grid*=100, the number of ECE particles is 0.1×N.
SED-PSO	M=20, S=30, α=0.05, N=2, ω=0.724, c_1=c_2=2, *iter*=5.

species, otherwise the main branch species will not change. If the main branch species changed, G_s increases by 1, and the derivative ability of the main branch species is replaced by the derivative ability of the new species.

$$\boldsymbol{X}_s^k = \boldsymbol{X}P_s^{k,j} \tag{3}$$

$$A_s^k = AP_s^{k,j} \tag{4}$$

2) SPECIES DERACINATING

Species deracinating means the disappearance or destruction of the non-renewability of a species. In SED algorithm, some species with poor fitness values will not be able to produce new species. The species produced by exploding will be ranked, and the S species with the best fitness value will be retained, while the remaining species will be eliminated. And the derivative ability of all surviving species will shrink, and the derivative ability of surviving species will be adjusted as follows.

$$\begin{cases} A_s^k = \alpha A_s^k & \text{while} \quad A_s^k > A_{\min} \\ A_s^k = A_{\min} & \text{while} \quad A_s^k \le A_{\min} \end{cases} \tag{5}$$

where α is the shrinkage coefficient and $\alpha < 1$. The smaller α is, the faster the algorithm converges, but the algorithm is more likely to fall into local extremum. On the contrary, the larger α is, the slower the convergence speed of the algorithm is, but the exploration of the search area by the algorithm is more comprehensive and detailed, which is conducive to finding the true global optimal. Experiments show that when α is 0.1 $\sim$ 0.5, SED algorithm can achieve a better balance between convergence speed and falling into local extremum.

According to Eq. (5), with the increase of the iteration, the derivative ability of species will decline rapidly, that is, the older the species, the weaker the derivative ability. $A_{\min}$ can prevent the derivative ability of species shrinking to zero, so that SED algorithm can continue to explore the optimal solution and maintain the diversity of the population.

Species Deracinating is a redistribution of computing resources, which allows high-quality species and their descendants to have more derivative opportunities, and makes the area with search value full of species, so as to realize fine exploration. Therefore, SED algorithm can take into account the convergence ability and solution quality.

In SED algorithm, the derivative ability of the main branch species does not always decrease. As shown in Figure 1, assuming that a smaller fitness value represents better performance of the species. Species a as the main branch derived b. Since $Y_b > Y_a$, species c was derived from a. However, species d was derived from c which is the main branch because $Y_c < Y_a$. And species d becomes the main branch because $Y_d < Y_c$. According to this law, the species *next* continues to be derived, so as to approach the optimal point T. Since the derived species c has a better fitness value, the main branch transfer occurs, and the species c has a greater derivative ability than species a, because the restrictions of shrinkage coefficient and the number of generations.

3) THE IMPLEMENTATION OF SED ALGORITHM

The specific steps of SED algorithm are as follows:

Step 1: Initialize surviving species which are main branch species, and the number of main branches is S. And initialize derivative capacity A and shrinkage coefficient α of each main branch species.

Step 2: Perform species exploding. Each main branch species will randomly derive a new species within its range of derivative ability. Detect whether the new species crosses the boundary. If the new species crosses the boundary, the

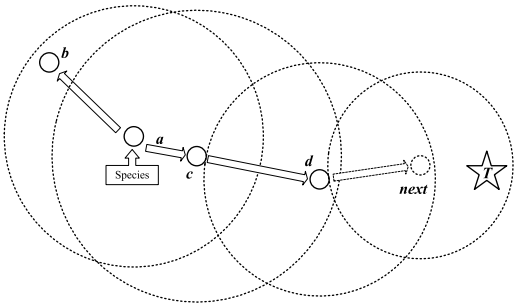

FIGURE 1. Diagrammatic sketch of species derivatives.

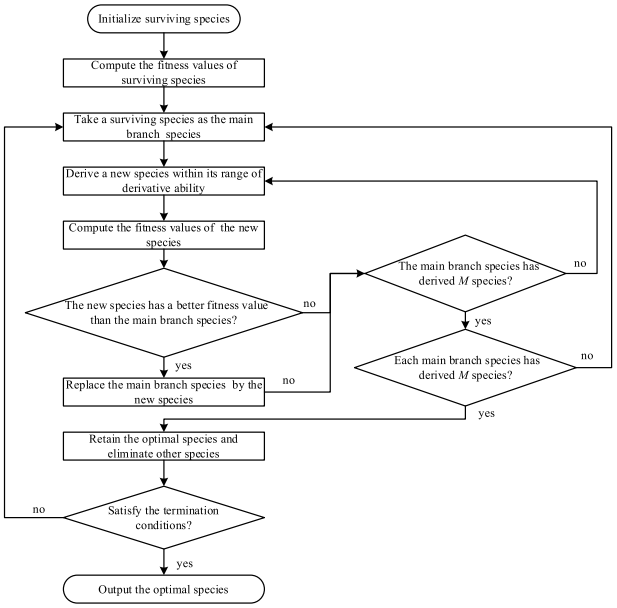

FIGURE 2. The flow chart of SED algorithm.

new species will be processed according to Part D ''Transboundary Processing'' in Section II, the generation number G_s of new species is added by 1, and the new derivative ability of new species is obtained according to Eq. (2). If the new species has a better fitness value than the main branch species, the main branch species will be replaced by the new species.

Step 3: Perform species deracinating. According to the fitness values of all species, retain the optimal species and eliminate other species. The retained species are surviving species, and the number of surviving species is S.

Step 4: Shrink the derivative ability of surviving species. The derivative ability of all surviving species is shrunk according to Eq (3).

Step 5: Complete this iteration and check the termination conditions. If the conditions are satisfactory, the optimal species will be output and the algorithm will terminate. Otherwise, switch to *step*2.

The flow chart of SED algorithm is shown in Figure 2.

B. PSO ALGORITHM

PSO algorithm originated from the group behavior of birds and fish. In PSO algorithm, the population is composed of a large number of particles. Each particle which represents a potential solution is a point in the search space, and has a fitness value and a velocity. PSO does not require any derivate information of the optimized function, uses only rudimentary mathematical operators, and is conceptually very simple.

Assuming the dimension of the search area is D, and the number of particles in the PSO algorithm is N (i.e., the population size is N). The position and velocity of the particle are $\boldsymbol{x}_i = (\boldsymbol{x}_{i,1}, \boldsymbol{x}_{i,2}, \ldots, \boldsymbol{x}_{i,D})$, $\boldsymbol{v}_i = (\boldsymbol{v}_{i,1}, \boldsymbol{v}_{i,2}, \ldots, \boldsymbol{v}_{i,D})$, $1 \leq i \leq N$. The fitness value of the particle is $Y = f(\boldsymbol{x}_i)$.

The relationship of position and the velocity after the k-th iteration are obtained in the following updating formula.

$$v_{i,d}^{k+1} = \omega \cdot v_{i,d}^{k} + c_1 \cdot r_1 \cdot [p_{i,d}^{k} - x_{i,d}^{k}] + c_2 \cdot r_2 \cdot [g_d^{k} - x_{i,d}^{k}], \; 1 \leq d \leq D \tag{6}$$

$$x_{i,d}^{k+1} = x_{i,d}^{k} + v_{i,d}^{k+1} \tag{7}$$

where ω is inertia weight, and $\boldsymbol{p}_i$ is the best previous position of $\boldsymbol{x_i}$ while $\boldsymbol{g}$ is the best overall position achieved by a particle within the entire population. $p_{i,d}^{k}$ is the d-th dimension of the $\boldsymbol{p}_i$ in the k-th iteration, and g_d^{k} is the d-th dimension of the $\boldsymbol{g}$ in the k-th iteration. The acceleration factors c_1 and c_2 are positive constants that control the relative impact of the personal knowledge and common knowledge on the movement of each particle. r_1 and r_2 are independent, uniformly distributed random variables in the range of (0, 1). To avoid the particle beyond the optimized range, the particle position and speed must be limited, i.e., $v_{i,d}^{k} \in [-V_{\max}, V_{\max}]$, $x_{i,d}^{k} \in [L_{\min}, L_{\max}]$, where $[L_{\min}, L_{\max}]$ represents the search interval. Usually, $V_{\max} = L_{\max} - L_{\min}$, $V_{\min} = -V_{\max}$.

C. ECE-PSO ALGORITHM

In ECE-PSO algorithm, each particle produces some virtual particles which represent candidate positions. The particle will be replaced by the best one among virtual particles and itself. The main idea of ECE-PSO algorithm is [12]: after each iteration, particles with better fitness value under the current iteration times are selected as ECE particles. Each ECE particle generates P virtual particles according to Eq. (8). If the fitness of the virtual particles is better than the current ECE particles, the current ECE particles are replaced with the virtual particles with the best fitness among S virtual particles.

$$x_{i,d}^{p} = x_{i,d} + \sin(2\pi \cdot rand_1) \cdot R \cdot rand_2 \tag{8}$$

where $p = 1, 2, \ldots, P$. $rand_1$ and $rand_2$ are independent random numbers between (0, 1). The virtual particles are randomly located in a circle whose center is $\boldsymbol{x}_i$ and the radius is R. R is related to the specific optimization problem. The method of adaptive adjustment of R is shown in Eq. (9).

$$R = \frac{L}{Grid} \cdot \left(1 - 0.9 \cdot \frac{k}{FE_{\max}}\right) \tag{9}$$

where $L = L_{\max}$-$L_{\min}$ represents search domain, and $FE_{\max}$ is the maximal iteration. *Grid* represents the number of subintervals which are produced by equally dividing L.

D. TRANSBOUNDARY PROCESSING

It is inevitable for particles and species to cross the boundary in PSO algorithm and SED algorithm. Therefore, the particles and species must be detected for transboundary, and if they exceed the required range, the transboundary particles and species must be processed.

Taking PSO algorithm as an example, the methods of transboundary processing are as follows.

① Absorption boundary. The particle takes its boundary value in this dimension if the particle crosses the boundary in this dimension.

② Reflection Boundary. The particle has the same velocity in this dimension but the direction is reversed, if the particle crosses the boundary in this dimension.

③ Invisible boundary. If the particle crosses the boundary, the particle does not participate in the next iteration.

④ Random reflection boundary. Reinitializes the particle value in this dimension, if the particle crosses the boundary in this dimension.

III. SED-PSO ALGORITHM

In basic SED algorithm, each species represents a single individual. If each species in SED algorithm is defined as a population composed of several similar individuals, SED algorithm can provide a basic algorithm combination platform for integrating various EC algorithms. In this way, each surviving species in the SED algorithm has only one individual, while species exploding, each new species generated by the species exploding can continue to generate a number of species with the same type. That is, each new species generated by the species exploding can adopt other EC algorithm to generate a number of species with the same type.

To reduce the computation burden and illustrate the excellent performance of the integrating SED algorithm with other EC algorithms, only the basic PSO algorithm and basic SED algorithm are integrated which named SED-PSO algorithm in this paper. The specific steps of SED-PSO algorithm are as follows:

Step 1: Set the parameters of SED algorithm, including: the number of surviving species S, the derivative ability A of all surviving species, the generation number G of surviving species, species exploding times M, the shrinkage coefficient α, and the iteration number of the algorithm $FE_{\max}$.

Step 2: Set the parameters of the PSO algorithm, including: population size N, acceleration factors c_1 and c_2, inertia weight ω, the maximum particle moving speed $v_{\max}$ and $v_{-\max}$, and the iteration number of the PSO algorithm *iter*.

Step 3: Establish fitness function (i.e., objective function) according to optimization problem.

Step 4: Initial the surviving species of SED which are main branch species, and then calculate the fitness of each surviving species. The initialization formulas for each dimension of each species are as follows.

$$X_{s,d} = L_{\max} + (L_{\min} - L_{\max}) \times rand \tag{10}$$

$$A_s = A_{\max} - A_{\min} \tag{11}$$

where $X_{s,d}$ represents the d-th dimensional value of the s-th surviving species. A_s represents the derivative ability of the s-th surviving species. *rand* is a random scalar between (0, 1). $[L_{\min}, L_{\max}]$ represents the search interval. $A_{\max}$ and $A_{\min}$ represent the maximum and the minimum derivative ability of species individually.

Step 5: Perform species exploding. Each main branch species will *randomly* derive a new species within its range of derivative ability according to Eq. (1) and Eq. (2).

Step 6: Detect whether the new species crosses the boundary. The value of each dimension of the new species should be *within* the interval of $[L_{\min}, L_{\max}]$. If it is not within this interval, the dimension crosses the boundary (i.e., the new species crosses the boundary). For the transboundary species, the value of transboundary dimension is replaced by random number in the interval of $[L_{\min}, L_{\max}]$.

Step 7: Use PSO algorithm to deeply exploration for each new species generated by species exploding.

① Generate N particles based on the new species generated by species *exploding*. Each particle is generated as follows.

$$x_{i,d} = rand \times XP_{s,d}^{k,j} \tag{12}$$

$$v_{i,d} = (L_{\max} - L_{\min}) \times (rand - 0.5) \tag{13}$$

where $x_{i,d}$ represents the d-th dimensional value of the i-th particle. $v_{i,d}$ represents the velocity value of the d-th dimension of the i-th particle. $XP_{s,d}^{k,j}$ represents the d-th dimensional value of the $\boldsymbol{X}P_s^{k,j}$. $i = 1,2,\ldots,N$.

② Detect whether the position of each particle crosses the boundary. The value of each dimension of each particle position should be within the interval of $[L_{\min}, L_{\max}]$. If it is not within this interval, the dimension crosses the boundary (i.e., the particle crosses the boundary). For the transboundary particle position, the value of transboundary dimension is replaced by random number in the interval of $[L_{\min}, L_{\max}]$.

③ Calculate the fitness values of each particle according to the fitness function. And select the global optimal *solution* $\boldsymbol{g}$, and the best previous solution $\boldsymbol{p}_i$ of each particle is the current position of the particle.

④ Update the velocity and position for each particle according to *Eq.* (6) and Eq. (7), and detect whether the position or velocity crosses the boundary. The velocity value of each dimension of each particle should be within the interval of $[-(L_{\max} - L_{\min}), (L_{\max} - L_{\min})]$. If it is not within this interval, the dimension crosses the boundary. For the transboundary particle velocity, the value of transboundary dimension is replaced by random number in the interval of $[-(L_{\max} - L_{\min}), (L_{\max} - L_{\min})]$. The position value of each dimension of each particle should be within the interval of $[L_{\min}, L_{\max}]$. If it is not within this interval, the dimension crosses the boundary. For the transboundary particle position, the value of transboundary dimension is replaced by random number in the interval of $[L_{\min}, L_{\max}]$.

⑤ Calculate the fitness values of each particle according to the fitness function. And select the global optimal solution $\boldsymbol{g}$ and the best previous solution $\boldsymbol{p}_i$.

For each particle, if the fitness value of updated particle is better than the fitness value of $\boldsymbol{p}_i$, the $\boldsymbol{p}_i$ is replaced by the position of the updated particle. After all particle position are updated in each iteration, the particle with the best fitness value is selected. If the fitness value of this particle is better than the fitness value of the global optimal solution $\boldsymbol{g}$, the global optimal solution $\boldsymbol{g}$ is replaced with the position of this particle.

⑥ Repeat *steps* ④∼⑤ until the iteration number of PSO algorithm is equal to *iter*.

⑦ Replace the value of the new species generated by *species* exploding with the global optimal solution $\boldsymbol{g}$, i.e., $\boldsymbol{X}P_s^{k,j} = \boldsymbol{g}^{iter}$, if the fitness value of $\boldsymbol{g}$ is better than the fitness value of $\boldsymbol{X}P_s^{k,j}$.

Step 8: Mark the main branch species. If the fitness value of *the* new species is better, the new species was labeled as the main branch species; otherwise, the old species was the main branch species. Only main branch species can derive a new species. If the main branch species changed, G_s increases by 1. And the derivative ability of the main branch species is replaced by the derivative ability of the new species as shown in Eq. (3) and Eq. (4).

Step 9: *Repeat* Steps 5 to 8, until all surviving species exploding M times.

Step 10: Order the fitness values of all new species generated by species exploding. Select S species with the *best* fitness value as the surviving species. Shrink the derivative ability of the surviving species according to Eq. (5). And eliminate other species.

Step 11: Repeat *step* 5 to 10, until the termination condition is met.

The flow chart of SED-PSO *algorithm* is shown in Figure 3.

IV. SIMULATION AND PERFORMANCE ANALYSIS ON THE BEHCHMARK FUNCTION PUBLISHED BY CEC

To analyze the performance of SED-PSO algorithm in detail, 27 benchmark functions published by CEC are selected, All the benchmark problems considered here are the minimization problems. For each problem, the formula, search range, global minimum, and error goal are recorded in Table 1. These benchmark functions include unimodal functions, multimodal functions, and the biased function. Except f_{18} which is 100 dimensions, the other functions are 30 dimensions. The performance of SED-PSO algorithm in solving the 27 selected CEC benchmark functions are compared with three optimization algorithms known as: PSO, ECE-PSO and SED.

A. PARAMETER SETTING

The parameter settings of PSO, SED, ECE-PSO and SED-PSO algorithm are shown in Table 2. The maximum of iteration of the above algorithms are 2000 (i.e., $FE_{\max}$ =2000). And Monte Carlo experimental analysis method is adopted, and Monte Carlo experiment times is 30 times.

B. PERFORMANCE EVALUATION INDEX

To comprehensively evaluate the performance of SED-PSO algorithm, six indexes including the success rate (*SR*), the metrics success performance (*SP*), the best fitness value of the optimal solution (*BF*), the mean fitness value of the optimal solution (*MF*), the variance of the fitness value of the optimal solution (*VF*) and the worst fitness value of the optimal solution (*WF*) are adopted. For the cases where *SP*s are not available, we use the fitness value.

A run during which the algorithm achieves a solution at the fixed accuracy level within the maximum number of function evaluations (i.e., $FE_{\max}$) is considered to be successful.

The success rate is the proportion of the number of successful runs in the total number of runs (i.e., Monte Carlo experiment times is the total number of runs).

$$SR = \#\text{of successful runs/total \#of runs} \tag{14}$$

The success performance (*SP*) is the number of function evaluations for the algorithm to reach the fixed accuracy level. The mean *SP* [27] is

$$\begin{aligned} &mean(SP) = \\ &[(1 - SR)/SR]\, FE_{\max} \\ &\quad + \text{mean}(\#\text{of function evaluations for successful runs}) \end{aligned} \tag{15}$$

The optimal solution is the solution that is closest to the expected solution in a run.

The *BF* is the best fitness value in all of Monte Carlo experiments results.

The *MF* is the mean fitness value of all Monte Carlo experiments results.

The *VF* is the variance of fitness values of all Monte Carlo experiment results.

The *WF* is the fitness value of solution with the largest deviation from the expected solution in all Monte Carlo experiments.

C. SIMULATION RESULTS ANALYSIS

The changing curves of fitness values of the four optimization algorithms along with iterations are shown in Figure 4. It is obvious SED-PSO algorithm has an excellent performance in most minimization problems listed in Table 1, except $f_{12}, f_{13}, f_{14}, f_{18}, f_{25}$.

The *MF*, *SP* and *SR* of four optimization algorithms are shown in Table 3. From the table we can find the *MF*, *SP* and *SR* of SED-PSO algorithm are best in 17, 23 and 23 minimization problems individually, compared with SED, PSO and ECE-PSO algorithms. The number of best *MF*, *SP* and *SR* of SED algorithm are 2, 12 and 12, individually. The number of best *MF*, *SP* and *SR* of PSO algorithm are 0, 2 and 2, individually. The number of best *MF*, *SP* and *SR* of ECE-PSO algorithm are 8, 18 and 18, individually. These indexes indicate SED-PSO algorithm can effectively improves the performance of SED and PSO algorithms, and

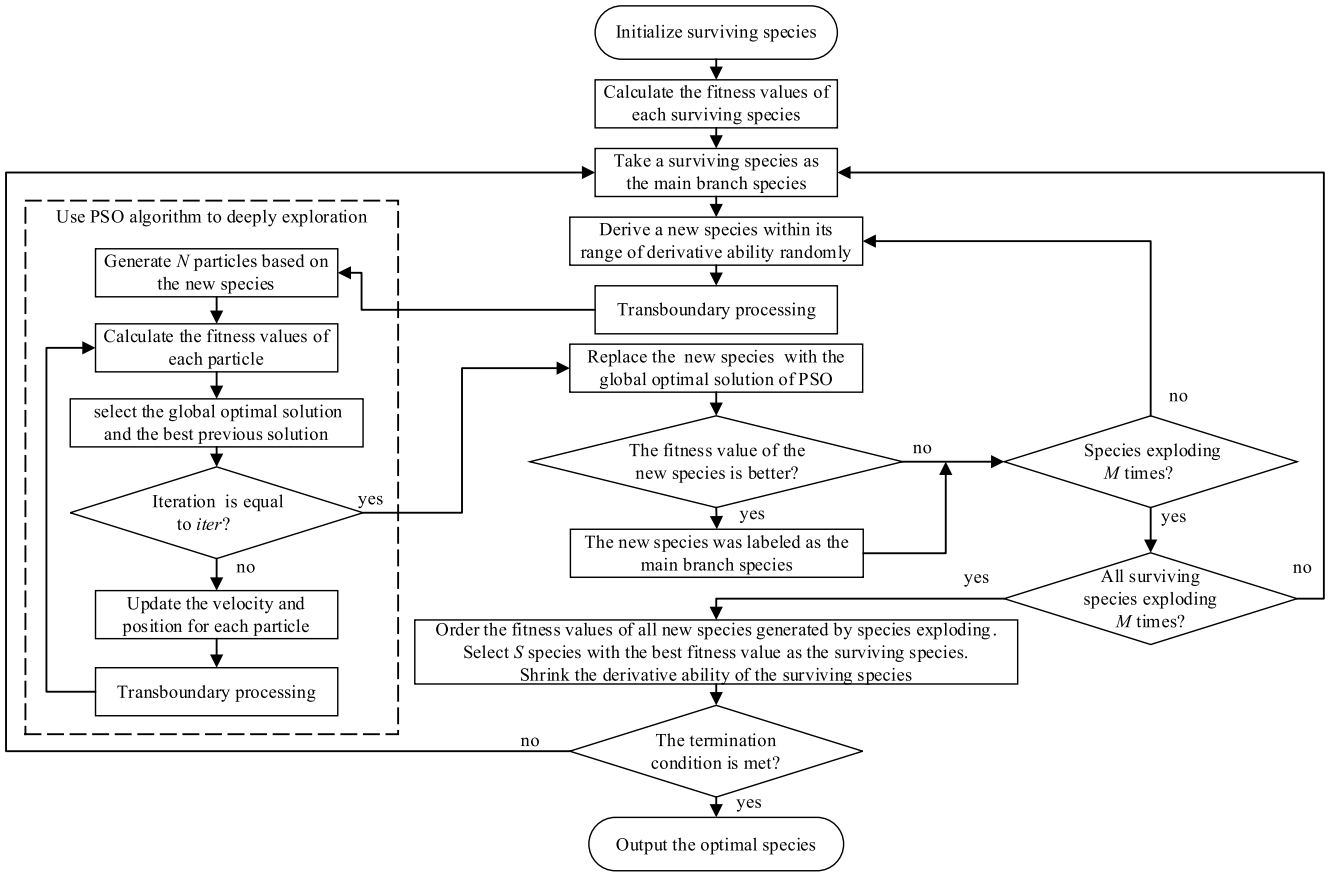

FIGURE 3. The flow chart of SED-PSO algorithm.

the *SP* of SED-PSO is significantly reduced, which means SED-PSO needs less iteration than SED and PSO algorithms.

The *BF*, *MF*, *WF*, *VF* of the four optimization algorithms in 30 Monte-Carlo experiments are shown in Table 4. From the table we can find the best *BF*, *WF* and *VF* of SED-PSO algorithm are 19, 20, 19 individually, compared with SED, PSO and ECE-PSO algorithms. The number of best *BF*, *WF* and *VF* of SED algorithm are 5, 2 and 1, individually. The number of best *BF*, *WF* and *VF* of PSO algorithm are 0, 0 and 1, individually. The number of best *BF*, *WF* and *VF* of ECE-PSO algorithm are 11, 8 and 6, individually. These indexes indicate SED-PSO algorithm in general outperform other algorithms on most of the minimization problems. And SED-PSO algorithm has a better stability than SED and PSO algorithms, because the *VF* of SED-PSO algorithm is less than SED and PSO algorithms. The experimental results also indicate that the convergence speed and solution quality of SED-PSO algorithm outperform the ECE-PSO algorithm.

The above results of 27 minimization problems comprehensively reflect the performance of the SED-PSO algorithm, which indicates SED-PSO algorithm has a high solution quality, strong stability and fast convergence speed, and can be applied to the large complex optimization problems. Although SED-PSO algorithm does not achieve the purpose of overall dominance compared with the SED and PSO algorithms, SED-PSO algorithm has a better solution quality in 20 minimization problems. Except f_{25} (Shifted Rastrigin), the solutions quality of SED-PSO algorithm are between SED and PSO algorithms in 6 minimization problems. It is obviously that SED algorithm can provide a basic platform to integrate various EC algorithms, and can take into account the advantages of each EC algorithm.

V. SIMULATION AND PERFORMANCE ANALYSIS ON COVERAGE OPTIMIZATION OF SENSOR NETWORKS

A. SENSOR COVERAGE MODEL

Assuming the sensing radius of sensor is r_s, and the uncertainty range of sensing is δr_s (δ<1). Then the detection probability of target $T(x_t, y_t)$ to sensor S_j is as follows.

$$p_{R_j} = \begin{cases} 0 & d(S_j, T)/r_s - 1 \geq \delta \\ \lambda_2 & \exp(-\dfrac{\lambda_1 \alpha_1^{\beta_1}}{\alpha_2^{\beta_2}}) - \delta \leq d(S_j, T)/r_s - 1 < \delta \\ \lambda_2 & d(S_j, T)/r_s - 1 < -\delta \end{cases} \tag{16}$$

where $d(S_j, T) = \sqrt{(x_t - x_{s,j})^2 + (y_t - y_{s,j})^2}$ is the distance between target T and sensor S_j. λ_1, λ_2, β_1 and β_2 are parameters related to the characteristics of the sensor. α_1 and α_2 are

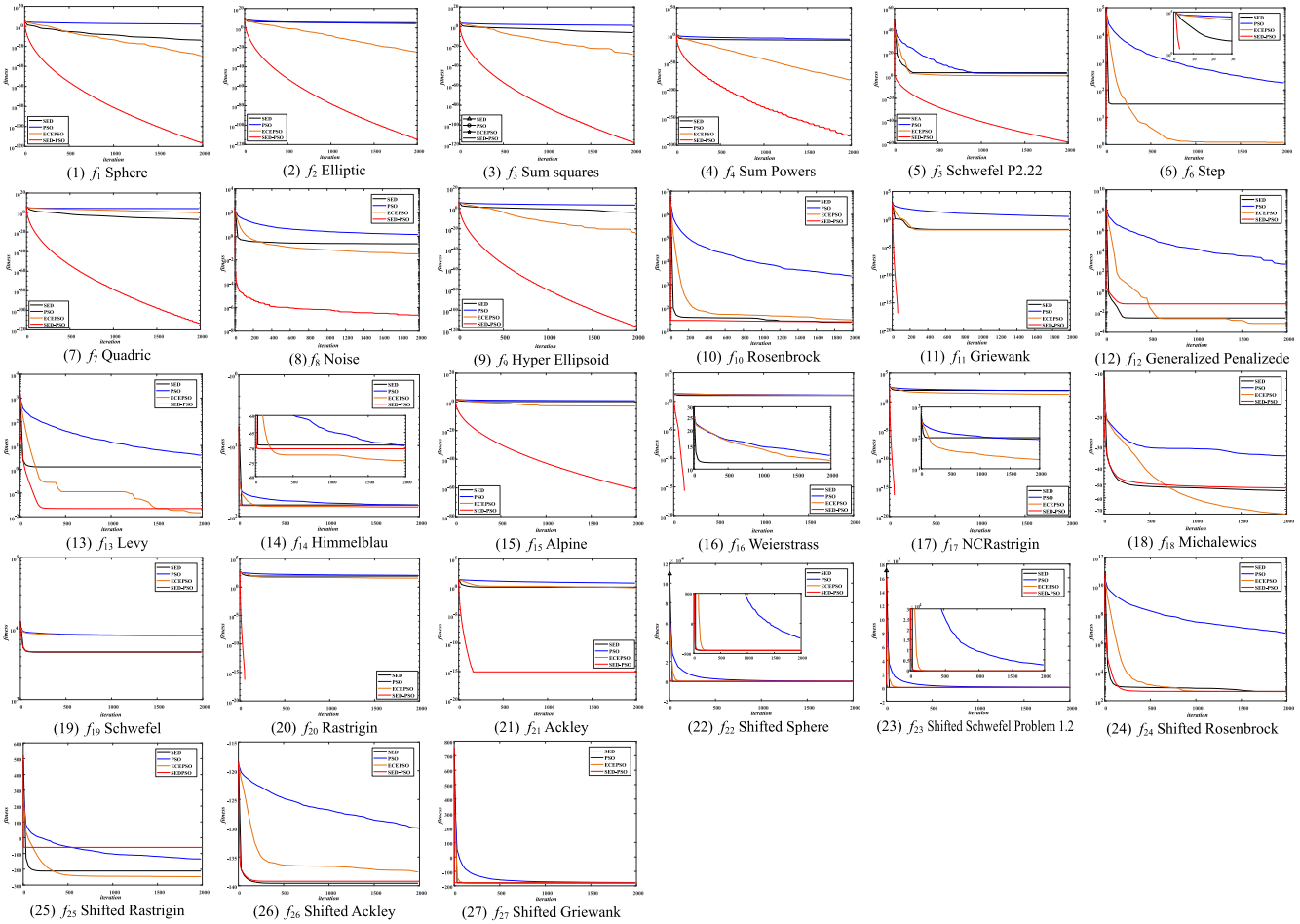

FIGURE 4. The changing curves of fitness values along with iterations in 27 benchmark functions.

input parameters, and their value are related to r_s and $d(S_j, T)$. The relationship is as follows.

$$\alpha_1 = r_s(\delta - 1) + d(S_j, T)$$
$$\alpha_2 = r_s(\delta - 1) - d(S_j, T) \tag{17}$$

The coverage model of sensor node shown in Eq. (16) is usually referred to as the probabilistic coverage model, which is derived from the binary coverage model. To the noise interference, the probabilistic coverage model is more realistic than the binary coverage model. However, to the sensor networks coverage optimization methods, the probabilistic coverage model is more complicated than the binary coverage model, and it is often difficult to use geometric analytic methods to find the optimal location deployment of sensor nodes. Because SED-PSO and ECE-PSO have an excellent performance in most minimization problems listed in Table 1, we use SED-PSO and ECE-PSO algorithm to find the optimal location deployment of sensor nodes.

Definition 1: Effective coverage point: when $p_R \geq p_{\text{th}}$, the target is perceived by the sensor node with a high probability, and the location of the target is called effective coverage point.

Definition 2: Coverage rate: The ratio of all effective coverage area to the area needs the node deployment.

In a sensor network, the number of sensor nodes is S, and all the sensor nodes have the same sensing radius. The detection probability of a target in the sensor network is as follows.

$$p_R = 1 - \prod_{j=1}^{S}(1 - p_{R_j}) \tag{18}$$

where p_{R_j} is the detection probability of the j-th sensor.

B. SIMULATION SETTING

Assuming the sensor network has such characteristics as:

① the sensor network consists of static sensor nodes and movable sensor nodes. The position of all nodes can be obtained, and the movable node can accurately move to the optimized position.

② the coverage model of nodes in sensor network is the probabilistic coverage model shown in Eq. (16).

Because the clustering can reduce network energy consumption and improve network fault tolerance and scalability.

TABLE 3. Optimization results with 30 trials.

Algorithm	*MF*	*SP*	*SR*(%)	*MF*	*SP*	*SR*(%)	*MF*	*SP*	*SR*(%)
	Sphere (f_1)			Elliptic (f_2)			Sum squares/Quartic (f_3)		
SED	1.78e-14	456.33	**100**	2.85e+05	-	0	6.70e-07	59619	3.33
PSO	2.01e+02	—	0	3.12e+04	-	0	2.16e+01	-	0
ECE-PSO	2.67e-30	417.27	**100**	7.72e-26	877.37	**100**	4.07e-29	536.47	**100**
SED-PSO	**1.89e-117**	**14.17**	**100**	**2.42e-115**	**40.40**	**100**	**1.17e-118**	**20.80**	**100**
	Sum Powers (f_4)			Schwefel P2.22 (f_5)			Step (f_6)		
SED	1.16e-09	570.70	**100**	5.54e+02	-	0	3.03e+01	-	0
PSO	4.61e-08	4.57e+03	40	1.28e+02	-	0	1.81e+02	-	0
ECE-PSO	1.44e-82	114.40	**100**	1.05e+00	-	0	1.20e+00	2.91e+03	46.67
SED-PSO	**6.20e-184**	**3.33**	**100**	**1.34e-59**	**66.50**	**100**	**0.00e+00**	**4**	**100**
	Quadric (f_7)			Noise (f_8)			Hyper Ellipsoid (f_9)		
SED	7.51e-08	1.59e+03	**100**	2.29e-03	-	0	1.29e-04	-	0
PSO	1.19e+04	-	0	1.39e+00	-	0	2.35e+03	-	0
ECE-PSO	6.28e-01	-	0	3.19e-02	-	0	6.19e-25	630.73	**100**
SED-PSO	**2.83e-114**	**20.43**	**100**	**2.25e-07**	**4.50**	**100**	**5.01e-117**	**27.90**	**100**
	Rosenbrock (f_{10})			Griewank (f_{11})			Generalized penalizede (f_{12})		
SED	**2.38e+01**	107.50	**100**	1.50e-02	151.43	**100**	2.5e-03	17.70	**100**
PSO	2.33e+03	-	0	3.26e+00	-	0	3.79e+02	-	0
ECE-PSO	3.05e+01	269.43	**100**	1.26e-02	183.30	**100**	**7.32e-04**	204.33	**100**
SED-PSO	2.52e+02	**2.80**	**100**	**0**	**6**	**100**	6.19e-02	**2**	**100**
	Levy (f_{13})			Himmelblau (f_{14})			Alpine (f_{15})		
SED	1.25e+00	**16.80**	**100**	-6.90e+01	4.68e+03	30	1.84e+00	-	0
PSO	3.84e+00	1.25e+03	96.67	-6.91 e+01	4.88e+03	36.67	6.76e+00	-	0
ECE-PSO	**1.47e-02**	100.30	**100**	**-7.43** e+01	**4.02e+02**	**100**	1.3e-03	7.07e+03	26.67
SED-PSO	2.20e-02	19.13	**100**	-7.02 e+01	2.01e+03	50	**9.68e-62**	**44.67**	**100**
	Weierstrass (f_{16})			NCRastrigin (f_{17})			Michalewics (f_{18})		
SED	1.12e+01	-	0	1.02e+02	-	0	-5.45e+01	14527	13.33
PSO	1.27e+01	-	0	8.82e+01	-	0	-3.40e+01	-	-
ECE-PSO	1.16e+01	-	0	1.96e+01	-	0	**-7.49e+01**	**1.03e+03**	**96.67**
SED-PSO	**0.00e+00**	**71.23**	**100**	**0.00e+00**	**23.03**	**100**	-5.25e+01	58534	3.33
	Schwefel (f_{19})			Rastrigin (f_{20})			Ackley (f_{21})		
SED	**4.65e+03**	2.03	**100**	8.21e+01	485.04	83.33	9.57e-01	-	-
PSO	7.88e+03	4.33	**100**	1.09e+02	5.47e+03	33.33	5.01e+00	-	-
ECE-PSO	7.87e+03	2.23	**100**	3.61e+01	132.33	**100**	6.66e-01	5.99e+03	30
SED-PSO	4.70e+03	**1.93**	**100**	**0.00e+00**	**2**	**100**	**8.88e-16**	**53.70**	**100**
	Shifted Sphere (f_{22})			Shifted Schwefel Problem 1.2 (f_{23})			Shifted Rosenbrock (f_{24})		
SED	**-450.0000**	**17.5667**	100	-449.9998	**113.2000**	**100**	447.4477	1.05e+03	76.67
PSO	-248.6212	1.18e+04	16.67	2463.727	-	-	5.011e+06	-	-
ECE-PSO	**-450.0000**	129.3667	**100**	**-450.0000**	190.9333	**100**	**430.4750**	760.2500	**93.33**
SED-PSO	**-450.0000**	24.1000	**100**	-449.9993	133.7333	**100**	442.2840	**744.2500**	80
	Shifted Rastrigin (f_{25})			Shifted Ackley (f_{26})			Shifted Griewank (f_{27})		
SED	-211.1030	6.6377e+03	23.33	**-139.5700**	18.9000	**100**	-179.9775	10.6000	**100**
PSO	-134.8013	-	-	-129.9719	-	-	-177.1678	856.6667	**100**
ECE-PSO	**-247.1427**	**1.1215e+03**	**73.33**	-137.6001	448.1429	93.33	-179.9652	58.7000	**100**
SED-PSO	-61.36196	-	-	-139.2270	**16.9667**	**100**	**-179.9803**	**2**	**100**

Thus, the implementation steps of sensor networks coverage optimization in this paper are as follows.

Step 1: Establish the sensor coverage model. The coverage model of sensor nodes is established according to Eq. (16).

Step 2: Clustering. K-means algorithm is used to cluster static sensors, and all sensor nodes are divided into 4 clusters. 4 movable sensors are arranged at the center of each cluster as the cluster head, and one movable sensor is arranged at the center of the monitoring area as the network center.

Step 3: Optimize the location of movable sensors. The PSO algorithm is used to optimize the location of the remaining movable sensor nodes.

Assuming 20 static sensors and 18 movable sensors are randomly distributed within the monitoring area [4km × 4km]. The sensing radius of all sensor is r_s =1.2km, the uncertainty parameter δ is 0.6, and λ_1 =1, λ_2 =0.6, β_1 =3, β_2 =2. The parameter settings of ECE-PSO and SED-PSO algorithms are shown in Table 2. The fitness function is the inverse of the sensor network coverage rate.

The sensor network coverage rate is calculated as follows.

Divide the monitoring by the granularity g = 40m, which leads the monitoring area is divided into [100 × 100] rasters. Calculate the number of the effective coverage rasters whose center detection probability is greater than p_{th} = 0.6. Then the coverage rate is the ratio of the number of the effective coverage rasters to the number of monitoring area rasters.

C. SIMULATION RESULTS ANALYSIS

Fig. 5 and Fig. 6 show the positions of all sensors and the coverage area using SED-PSO algorithm, individually. Fig. 7 shows the fitness value along with iteration. It can be seen SED-PSO algorithm has a better performance than ECE-PSO

TABLE 4. Optimization results with 30 trials.

Function	SED				PSO				ECE-PSO				SED-PSO			
	BF	*MF*	*WF*	*VF*	*BF*	*MF*	*WF*	*VF*	*BF*	*MF*	*WF*	*VF*	*BF*	*MF*	*WF*	*VF*
f_1	8.69e-15	1.78e-14	4.36e-14	7.50e-15	3.69e+01	2.01e+02	6.94e+02	1.46e+02	7.47e-38	2.70e-30	5.29e-29	9.71e-30	**1.09e-119**	**1.89e-117**	**9.91e-117**	**2.70e-117**
f_2	1.53e+05	2.85e+05	4.45e+05	7.99e+04	8.65e+03	3.12e+04	6.87e+04	1.67e+04	1.21e-32	7.72e-26	1.25e-24	2.55e-25	**7.55e-117**	**2.42e-115**	**1.51e-114**	**3.68e-115**
f_3	5.55e-10	6.70e-07	8.72e-06	1.76e-06	4.70e+00	2.16e+01	6.71e+01	1.50e+01	1.09e-38	4.07e-29	7.08e-28	1.51e-28	**4.87e-121**	**1.17e-118**	**2.06e-117**	**3.73e-118**
f_4	3.14e-10	1.16e-09	2.06e-09	3.70e-10	2.13e-10	4.61e-08	5.62e-07	1.02e-07	2.33e-100	1.44e-82	2.68e-81	5.64e-82	**2.26e-196**	**6.20e-184**	**1.71e-182**	**0.00e+00**
f_5	3.19e+02	5.54e+02	8.03e+02	1.18e+02	6.73e+00	1.28e+02	5.19e+02	1.69e+02	7.33e-06	1.05e+00	1.85e+01	3.95e+00	**1.25e-60**	**1.34e-59**	**4.65e-59**	**9.92e-60**
f_6	1.30e+01	3.03e+01	6.90e+01	1.20e+01	5.20e+01	1.81e+02	4.54e+02	9.79e+01	**0.00e+00**	1.20e+00	1.10e+00	2.11e+00	**0.00e+00**	**0.00e+00**	**0.00e+00**	**0.00e+00**
f_7	5.09e-09	7.51e-08	4.13e-07	1.01e-07	4.69e+03	1.19e+04	2.11e+04	3.96e+03	1.42e-02	6.28e-01	5.91e+00	1.13e+00	**1.50e-116**	**2.83e-114**	**5.61e-113**	**1.02e-113**
f_8	1.05e-01	2.29e-01	5.18e-01	9.69e-02	5.48e-01	1.39e+00	2.79e+00	5.34e-01	1.84e-02	3.19e-02	4.75e-02	7.69e-03	**9.79e-09**	**2.25e-07**	**5.44e-07**	**1.37e-07**
f_9	5.96e-08	1.29e-04	1.67e-03	3.70e-04	5.00e+02	2.35e+03	8.53e+03	1.79e+03	1.62e-36	6.19e-25	1.78e-23	3.25e-24	**5.23e-119**	**5.01e-117**	**5.73e-116**	**1.08e-116**
f_{10}	1.18e+01	**2.38e+01**	2.79e+01	2.92e+00	7.06e+02	2.33e+03	6.72e+03	1.60e+03	**2.70e-01**	3.05e+01	8.10e+01	2.65e+01	2.50e+01	2.52e+01	**2.55e+01**	**1.33e-01**
f_{11}	3.14e-13	1.50e-02	6.15e-02	1.75e-02	1.43e+00	3.26e+00	1.09e+01	1.80e+00	1.11e-16	1.26e-02	4.92e-02	1.50e-02	**0.00e+00**	**0.00e+00**	**0.00e+00**	**0.00e+00**
f_{12}	4.74e-16	2.50e-03	7.46e-02	1.44e-02	2.12e+01	3.80e+02	3.55e+03	7.44e+02	**1.35e-32**	**7.32e-04**	**1.10e-02**	**2.80e-03**	1.21e-15	6.19e-02	1.54e-01	5.58e-02
f_{13}	1.31e-06	1.25e+00	6.11e+00	2.10e+00	1.07e+00	3.84e+00	1.09e+01	2.17e+00	**1.35e-31**	**1.47e-02**	**1.10e-01**	**3.80e-02**	1.19e-06	2.20e-02	**1.10e-01**	4.47e-01
f_{14}	-72.6776	-69.0021	-64.1956	2.12e+00	-73.8171	-69.1821	-65.7555	2.44e+00	**-77.3899**	**-74.3107**	**-71.7352**	**1.64e+00**	-73.6201	-70.2273	-66.0805	1.96e+01
f_{15}	5.33e-01	1.84e+00	6.60e+00	1.41e+00	1.24e+00	6.76e+00	1.25e+01	3.11e+00	5.41e-11	1.30e-03	3.46e-02	6.30e-03	**1.78e-62**	**9.68e-62**	**4.45e-61**	**9.15e-62**
f_{16}	6.31e+00	1.12e+01	1.64e+01	3.32e+00	6.94e+00	1.27e+01	1.79e+01	2.21e+00	4.07e+00	1.16e+01	2.05e+01	6.82e+00	**0.00e+00**	**0.00e+00**	**0.00e+00**	**0.00e+00**
f_{17}	4.40e+01	1.02e+02	1.81e+02	2.55e+01	5.32e+01	8.82e+01	1.30e+02	2.00e+00	1.30e+01	1.96e+01	2.70e+01	3.88e+00	**0.00e+00**	**0.00e+00**	**0.00e+00**	**0.00e+00**
f_{18}	-61.5039	-54.5110	-47.3371	3.73e+00	-40.6318	-34.0375	-29.7344	**3.28e+00**	**-82.4044**	**-74.8908**	**-56.4421**	5.89e+00	-61.9315	-52.5297	-44.3949	4.23e+00
f_{19}	**3.14e+03**	**4.65e+03**	6.20e+03	7.41e+02	6.97e+03	7.88e+03	8.44e+03	3.36e+02	7.10e+03	7.87e+03	8.25e+03	**2.70e+00**	3.79e+03	4.70e+03	**6.14e+03**	5.30e+02
f_{20}	5.37e+01	8.21e+01	1.08e+02	1.59e+01	6.45e+01	1.09e+02	1.47e+02	2.13e+01	2.29e+01	3.61e+01	5.37e+01	9.02e+00	**0.00e+00**	**0.00e+00**	**0.00e+00**	**0.00e+00**
f_{21}	3.10e-08	9.57e-01	2.32e+00	8.22e-01	2.81e+00	5.01e+00	7.55e+00	1.13e+00	1.51e-14	6.66e-01	2.12e+00	7.64e-01	**8.88e-16**	**8.88e-16**	**8.88e-16**	**0.00e+00**
f_{22}	**-450.000**	**-450.000**	**-450.000**	8.77e-14	-425.953	-248.621	58.27330	1.13e+02	**-450.000**	**-450.000**	**-450.000**	1.78e-12	**-450.000**	**-450.000**	**-450.000**	**8.24e-14**
f_{23}	**-450.000**	-449.999	-449.998	5.14e-04	220.4924	2.46e+03	8.46e+03	2.11e+03	**-450.000**	**-450.000**	**-450.000**	**1.43e-11**	**-450.000**	-449.999	-449.986	2.7e-03
f_{24}	407.357	447.448	610.159	5.58e+01	4.09e+05	5.01e+06	1.57e+07	3.72e+06	**390.050**	**430.475**	**520.307**	**3.73e+01**	410.723	442.284	571.741	4.76e+01
f_{25}	-274.283	-211.103	-142.949	3.25e+01	-201.249	-134.801	-58.0956	3.60e+01	**-285.227**	**-247.143**	**-181.752**	2.33e+01	-61.3619	-61.3620	-61.3620	**4.21e-14**
f_{26}	**-140.000**	**-139.570**	**-138.498**	**5.88e-01**	-134.250	-129.972	-125.934	2.15e+00	-138.952	-137.600	-133.464	1.30e+00	**-140.000**	-139.227	-137.683	7.55e-01
f_{27}	**-180.000**	-179.978	-179.922	2.28e-02	-178.693	-177.168	-175.020	1.13e+00	**-180.000**	-179.965	-179.755	4.66e-02	**-180.000**	**-179.980**	**-179.941**	**1.79e-02**
NB	5	4	2	1	0	0	0	1	11	8	8	6	**19**	**17**	**20**	**19**

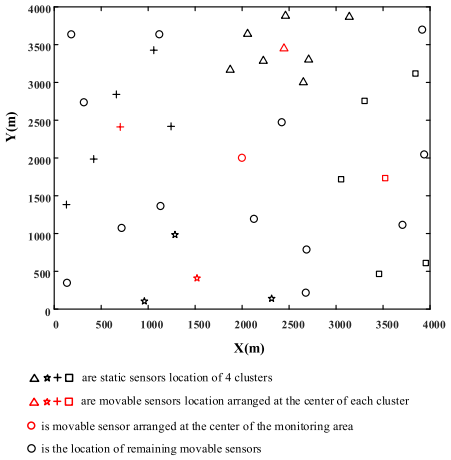

FIGURE 5. The positions of sensors using SED-PSO algorithm.

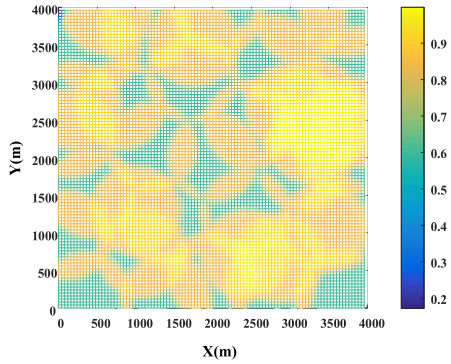

FIGURE 6. The coverage area using SED-PSO algorithm.

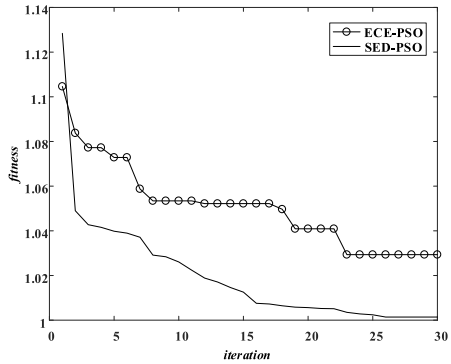

FIGURE 7. The fitness value along with iteration.

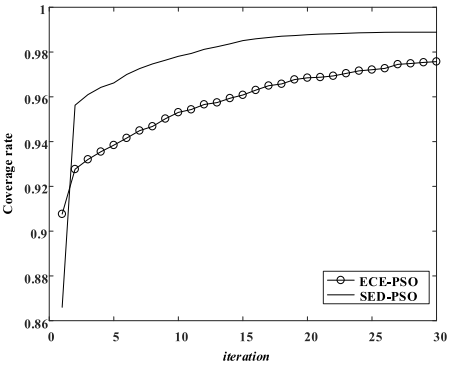

FIGURE 8. Average coverage ratio along with iteration.

algorithm, which is specifically reflected in the following two points: (1) the solution has a smaller fitness value, and (2) the number of falling into the local extremum is minimum, i.e., the number of times that the fitness value stagnating is smaller.

To further analyze the stability and effectiveness of SED-PSO algorithm, we use 30 Monte-Carlo test to verify the solutions of SED-PSO algorithm and ECE-PSO algorithm.

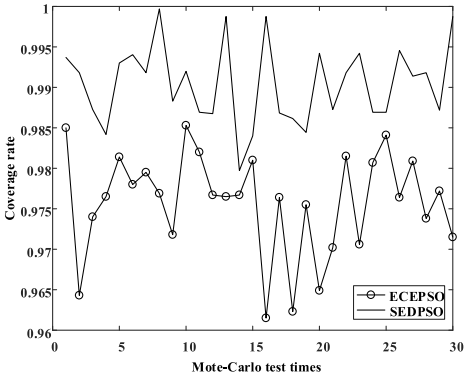

FIGURE 9. The final solutions of 30 Mote-Carlo tests.

Fig. 8 shows the curve of the average coverage ratio of 30 tests along with iterations. It is obvious the solutions of SED-PSO algorithm are better than ECE-PSO algorithm in each iteration. Fig. 9 shows the final solutions of 30 Mote-Carlo tests. it is obvious the SED-PSO algorithm achieves a better solution than ECE-PSO algorithms in every test.

The stability of ECE-PSO algorithm and SED-PSO algorithm in 30 tests can be reflected by the variance of the final solutions. The variance of ECE-PSO is 4.55e-05 and SED-PSO is 5.57e-06. It can be seen that the variance of SED-PSO algorithm is smaller, so the stability of SED-PSO is better than ECE-PSO algorithm.

VI. CONCLUSION

On the basis of SED algorithm, this paper tries to combine other EC algorithms effectively. Taking PSO algorithm as an example, SED-PSO algorithm is proposed. The simulation results on 27 benchmark functions published by CEC indicate SED-PSO algorithm has a high convergence accuracy, strong stability and fast convergence speed, and can be applied to solve the large and complex optimization problems. These simulation results fully demonstrate SED algorithm can provide a basic integrating platform for various EC algorithms, and can take into account the advantages of each EC algorithm, and has a wider application. The results of sensor network coverage optimization also indicate SED-PSO algorithm has a higher quality of solution, the minimum number of falling into the local extremum, and has a stronger stability and effectiveness.

REFERENCES

[1] Z.-J. Wang, Z.-H. Zhan, S. Kwong, H. Jin, and J. Zhang, "Adaptive granularity learning distributed particle swarm optimization for large-scale optimization," *IEEE Trans. Cybern.*, vol. 51, no. 3, pp. 1175–1188, Mar. 2021.

[2] S. Shadravan, H. R. Naji, and V. K. Bardsiri, "The sailfish optimizer: A novel nature-inspired metaheuristic algorithm for solving constrained engineering optimization problems," *Eng. Appl. Artif. Intell.*, vol. 80, pp. 20–34, Apr. 2019.

[3] D. E. Goldber, *Genetic Algorithms in Search, Optimization, and Machine Learning*, 1st ed. Boston, MA, USA: Addison-Wesley Longman, 1989.

[4] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. ICNN-Int. Conf. Neural Networks*, Perth, WA, Australia, Dec. 1995, pp. 1–8.

[5] Z.-J. Wang, Z.-H. Zhan, Y. Lin, W.-J. Yu, H. Wang, S. Kwong, and J. Zhang, "Automatic niching differential evolution with contour prediction approach for multimodal optimization problems," *IEEE Trans. Evol. Comput.*, vol. 24, no. 1, pp. 114–128, Feb. 2020.

[6] D. Liang, Z.-H. Zhan, Y. Zhang, and J. Zhang, "An efficient ant colony system approach for new energy vehicle dispatch problem," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 11, pp. 4784–4797, Nov. 2020.

[7] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," *IEEE Comput. Intell. Mag.*, vol. 1, no. 4, pp. 28–39, Nov. 2006.

[8] C. T. Yue, K. V. Price, P. N. Suganthan, J. J. Liang, M. Z. Ali, B. Y. Qu, N. H. Awad, and P. P. Biswas, "Problem definitions and evaluation criteria for the CEC 2020 special session and competition on single objective bound constrained numerical optimization," Zhengzhou Univ., Zhengzhou, China, and Nanyang Technol. Univ., Singapore, Glasgow, U.K., Tech. Rep., Jul. 2020.

[9] N. H. Awad, M. Z. Ali, P. N. Suganthan, J. J. Liang, and B. Y. Qu, "Problem definitions and evaluation criteria for the CEC 2017 special session and competition on single objective real-parameter numerical optimization," Nanyang Technol. Univ. Singapore, Jordan Univ. Sci. Technol., Jordan, Zhengzhou Univ. Zhengzhou, China, Tech. Rep., Nov. 2016.

[10] X.-Y. Zhang, J. Zhang, Y.-J. Gong, Z.-H. Zhan, W.-N. Chen, and Y. Li, "Kuhn–Munkres parallel genetic algorithm for the set cover problem and its application to large-scale wireless sensor networks," *IEEE Trans. Evol. Comput.*, vol. 20, no. 5, pp. 695–710, Oct. 2016.

[11] Z.-J. Wang, Z.-H. Zhan, and J. Zhang, "Solving the energy efficient coverage problem in wireless sensor networks: A distributed genetic algorithm approach with hierarchical fitness evaluation," *Energies*, vol. 11, no. 12, pp. 1–14, Dec. 2018.

[12] Y. Yongjian, F. Xiaoguang, Z. Zhenfu, W. Shengda, N. Jianguo, and C. Wenkui, "Improved particle swarm optimization based on particles' explorative capability enhancement," *J. Syst. Eng. Electron.*, vol. 27, no. 4, pp. 900–911, Aug. 2016.

[13] S. Molaei, H. Moazen, S. Najjar-Ghabel, and L. Farzinvash, "Particle swarm optimization with an enhanced learning strategy and crossover operator," *Knowl.-Based Syst.*, vol. 215, Mar. 2021, Art. no. 106768.

[14] Y. Yang, Z. Zhuo, B. Huang, X. Fan, Y. Deng, and B. Wang, "Species explode and deracinate algorithm," *Syst. Eng. Electron.*, vol. 40, no. 4, pp. 941–947, Apr. 2018.

[15] J. Liu, X. Ma, X. Li, M. Liu, T. Shi, and P. Li, "Random convergence analysis of particle swarm optimization algorithm with time-varying attractor," *Swarm Evol. Comput.*, vol. 61, no. 2, pp. 1–19, Mar. 2021.

[16] K. Jin'no, "Analysis of particle swarm optimization by dynamical systems theory," *Nonlinear Theory Appl. (IEICE)*, vol. 12, no. 2, pp. 118–132, Apr. 2021.

[17] H. Zhang, J. Xie, and B. Zong, "Bi-objective particle swarm optimization algorithm for the search and track tasks in the distributed multiple-input and multiple-output radar," *Appl. Soft Comput.*, vol. 101, Mar. 2021, Art. no. 107000.

[18] B. Wei, X. Xia, F. Yu, Y. Zhang, X. Xu, H. Wu, L. Gui, and G. He, "Multiple adaptive strategies based particle swarm optimization algorithm," *Swarm Evol. Comput.*, vol. 57, Sep. 2020, Art. no. 100731.

[19] N. Zeng, Z. Wang, W. Liu, H. Zhang, K. Hone, and X. Liu, "A dynamic neighborhood-based switching particle swarm optimization algorithm," *IEEE Trans. Cybern.*, vol. 52, no. 9, pp. 9290–9301, Sep. 2022.

[20] H. Yu, Y. Wang, and S. Xiao, "Multi-objective particle swarm optimization based on cooperative hybrid strategy," *Appl. Intell.*, vol. 50, pp. 256–269, Jul. 2019.

[21] F. Wang, H. Zhang, and A. Zhou, "A particle swarm optimization algorithm for mixed-variable optimization problems," *Swarm Evol. Comput.*, vol. 60, Feb. 2021, Art. no. 100808.

[22] K. Chen, F. Zhou, and A. Liu, "Chaotic dynamic weight particle swarm optimization for numerical function optimization," *Knowl.-Based Syst.*, vol. 139, pp. 23–40, Jan. 2018.

[23] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 240–255, Jun. 2004.

[24] Y.-J. Gong, J.-J. Li, Y. Zhou, Y. Li, H. S. Chung, Y.-H. Shi, and J. Zhang, "Genetic learning particle swarm optimization," *IEEE Trans. Cybern.*, vol. 46, no. 10, pp. 2277–2290, Oct. 2016.

[25] X. Zhang, X. Wang, Q. Kang, and J. Cheng, "Differential mutation and novel social learning particle swarm optimization algorithm," *Inf. Sci.*, vol. 480, pp. 109–129, Apr. 2019.

[26] W. Zhang, D. Ma, J.-J. Wei, and H.-F. Liang, "A parameter selection strategy for particle swarm optimization based on particle positions," *Exp. Syst. Appl.*, vol. 41, no. 7, pp. 3576–3584, Jun. 2014.
[27] M. Hu, T. Wu, and J. D. Weir, "An adaptive particle swarm optimization with multiple adaptive methods," *IEEE Trans. Evol. Comput.*, vol. 17, no. 5, pp. 705–720, Oct. 2013.

YONGJIAN YANG was born in 1988. He received the B.S. degree in information countermeasure from the University of Electronic Science and Technology of China, Chengdu, China, in 2010, and the M.S. degree in signal and information processing and the Ph.D. degree in information and communication engineering from Air Force Engineering University, Xi'an, China, in 2012 and 2016, respectively.

From 2018 to 2021, he was a Postdoctoral Fellow with Northwestern Polytechnical University, Xi'an. Since 2017, he has been a Lecturer with Air Force Engineering University. His research interests include radar signal and information processing, evolutionary computation algorithms, and multi-target tracking

YOUWEI DENG was born in 1981. He received the B.S. degree in electronic countermeasure and the M.S. degree in signal and information processing from Air Force Engineering University, Xi'an, China, in 2003 and 2009, respectively.

Since 2007, he has been a Lecturer with Air Force Engineering University. His research interests include radar signal and information processing, evolutionary computation algorithms, and multi-target tracking.

BINGSONG XIAO was born in 1982. He received the B.S., M.S., and Ph.D. degrees from Air Force Engineering University, Xi'an, China, in 2004, 2007, and 2010, respectively.

From 2010 to 2020, he was a Lecturer with Air Force Engineering University. Since 2021, he has been an Associate Professor with Air Force Engineering University. His research interest includes radar information processing.

XIAOHONG ZHAO was born in 1992. She received the B.S. and Ph.D. degrees in microelectronics from Xidian University, Xi'an, China, in 2014 and 2020, respectively. Since 2020, she has been a Lecturer with Air Force Engineering University. Her research interests include compound semiconductor, irradiation effect, and degradation mechanism.