Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 4600787, 23 pages
https://doi.org/10.1155/2022/4600787

Research Article

A Novel Compact Particle Swarm Optimization for Optimizing Coverage of 3D in Wireless Sensor Network

Ning Liu, Qing-Wei Chai, Shangkun Liu, and Wei-Min Zheng

College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao, Shandong, China

Correspondence should be addressed to Wei-Min Zheng; zhengwm901@126.com

Received 20 August 2022; Revised 7 October 2022; Accepted 10 October 2022; Published 18 October 2022

Academic Editor: Shaohua Wan

Copyright © 2022 Ning Liu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The problem of 3D coverage in a wireless sensor network (WSN) has always been an urgent problem to be solved. A novel compact particle swarm optimization algorithm (ncPSO) to solve this problem is proposed in this paper. This algorithm uses a Pareto distribution to describe the position of particle swarms. The ncPSO reduces memory usage compared to traditional heuristics. The ncPSO using the Pareto distribution is less likely to fall into local optima than other compact algorithms using the Gaussian distribution. We also add Gaussian perturbation strategy to the algorithm to better avoid the algorithm falling into local optimum. Among the test functions of CEC2013, the ncPSO achieves remarkable optimization ability on most test functions. Finally, we apply ncPSO to the 3D coverage problem of sensors. Compared with other algorithms, the ncPSO achieves satisfactory results.

1. Introduction

The heuristic algorithms develop rapidly, and many heuristic algorithms and their improved algorithms are proposed, such as particle swarm optimization (PSO) [1], genetic algorithm (GA) [2], whale optimization algorithm (WOA) [3], Black Hole (BH) [4], Artificial Bee Colony (ABC) [5], Sine Cosine Algorithm (SCA) [6], and Bat Algorithm (BA) [7]. The theorem of No Free Lunch (NFL) shows that no algorithm is applicable to all problems [8, 9]. Intelligent computing is applied in many fields such as architecture, transportation, education, medicine, and economics [10, 11]. Many improved heuristic algorithms have been proposed to solve different problems [12, 13]. Ji et al. proposed a multisurrogate-assisted multitasking particle swarm optimization to solve the problem of expensive multimodal optimization. Ji et al. introduced multitasking evolution into PSO, which makes the algorithm have faster convergence speed and better optimization ability in high-dimensional problems [14]. Song et al. proposed variable-size cooperative coevolutionary particle swarm optimization to solve the problem of "dimensional disaster." Using this algorithm, the feature selection problems with high-dimensional data are solved [15]. For the problem of mixed continuous and discrete data, Wang et al. proposed a new particle swarm optimization algorithm to solve this problem. Wang et al. tested the performance of the proposed algorithm with other algorithms on 28 test functions of CEC2013. The results show that the proposed algorithm by Wang et al. has better performance than other algorithms [16]. Fan et al. proposed a random reselection particle swarm optimization algorithm and applies it to the parameter optimization of solar photovoltaic modules [17]. The problem with existing heuristic algorithms is that they need large memory space. This problem is a limitation on tiny devices such as wireless sensor and microrobot. In order to solve this problem, compact Artificial Bee Colony (cABC) [18], compact Particle Swarm Optimization (cPSO) [19], compact Sine Cosine Algorithm (cSCA) [20], and compact Bat Algorithm (cBA) [21] are proposed. These algorithms with compact strategy all use normal distribution to represent the position of population. Although they can reduce the usage of memory space, they fall into local optimum simply. We propose a novel compact strategy to solve this problem.

The WSN has impacted every aspect of the life of people and has attracted the attention of many scholars [22, 23]. The goal of the WSN is to send collected information from monitoring area to base station for data analysis [24]. There are many types of sensors such as humidity, temperature, and brightness. Due to the many kinds of sensors which can collect different kinds of information, the WSN is applied in military, healthcare, environmental studies, and start home [25–27]. The coverage problem of the WSN is one of the most important problems to be solved. There are two main types of the coverage problem, namely, full coverage and partial coverage [28]. In this paper, we mainly research the partial coverage problems. However, the partial coverage problem in a given area includes two categories: the maximum coverage rate under a certain number of sensor nodes and the minimum number of sensor nodes to achieve a certain coverage rate. Increasing coverage rate by increasing the number of sensors will reduce the lifetime of the WSN. So it is a very vital topic to achieve maximum coverage rate with the minimum number of nodes.

Yoon and Kim first proposed that the coverage problem of WSN is NP-hard [29]. The coverage problem of WSN be studied in 3D is lesser than that in 2D. Researchers have proposed many methods to solve this problem of WSN in 2D. However, many solutions that perform well on 2D problems do not work well in 3D. The 3D coverage problems are more difficult than the 2D coverage problems, and the 3D coverage problems conform closer to the real world. In the 3D coverage problem, the sensor will encounter obstacles and affect the coverage of the area. The obstacle blocks the transmission of the sensor signal, so that the sensor cannot monitor the information of the area blocked by the obstacle. In this paper, we discuss how to solve the two-class partial coverage problems for sensors in a 3D environment containing obstacles.

The main contributions of this paper are as follows:

(1) Propose a novel compact strategy to solve the problem of large memory usage of heuristic algorithm. The novel compact strategy uses the Pareto distribution to describe the position of the whole particle swarm instead of the position of each particle to reduce memory usage

(2) Use Pareto perturbation to better avoid algorithm falling into local optimum. The Pareto perturbation is a heavy-tailed distribution. This distribution makes it easier for the algorithm to find the position outside the local optimum. This distribution enables particles to get the solution vector which deviates from the current optimum

(3) Combine the novel compact strategy and Pareto perturbation with PSO to solve the 3D coverage problem of sensor network. The performance of the ncPSO and other algorithms is compared on the CEC2013, and the optimization effect of ncPSO is also compared with other algorithms in 3D coverage problem

The rest of the paper is structured as follows. Section 2 covers a related work. We introduce the process of coming up with the ncPSO in Section 3. We show the performance test results in Section 4. Section 5 applies the ncPSO to the coverage problem of WSN. We give conclusion and outlook in Section 6.

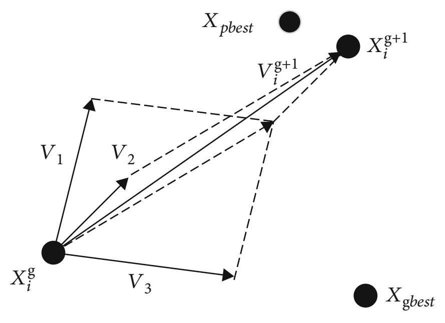


Figure 1: The movement process of the particle swarm.

2. Related Work

We introduce the PSO, cPSO, and current research progress on WSN coverage in this section. The coverage of WSN is one of the most important issues in current research. Because of the small size of the sensor, algorithms that require large memory are not suitable for it. Because intelligent computing can effectively solve the 3D coverage problem of WSN, the ncPSO is proposed to conveniently use for sensor in this paper. The traditional heuristic algorithms have the disadvantage of large memory usage. The proposed compact algorithms using Gaussian distribution have the disadvantage of easily falling into local optimum. In view of these two shortcomings, we propose the ncPSO that uses Pareto distribution to represent the position of particle swarm. Because the ncPSO solves the problems of large memory usage and easy falling into local optimum, the ncPSO is suitable for solving 3D coverage problems in WSN.

2.1. Particle Swarm Optimization. The PSO is proposed by Kennedy and Eberhart in 1995 according to the foraging behavior of birds [1]. The PSO is the most representative heuristic algorithm. Each particle records its own position, denoted by X_i. Every particle is a solution to the problem. The PSO will generate a current optimal solution pBest and a historical optimal solution gBest in each iteration. Each particle affected by current optimal position and historical optimal position will have a flight speed, and according to the current position and flight speed, they will move to a new position in the next iteration. The positions of particles are updated according to

$$\begin{aligned} V_i^{g+1} &= wV_i^g + c_1 \times \text{rand} \times \left(\text{pBest}_i - X_i^g\right) \\ &\quad + c_2 \times \text{rand} \times \left(\text{pBest} - X_i^g\right), \end{aligned} \tag{1}$$

$$X_i^{g+1} = X_i^g + V_i^{g+1}, \tag{2}$$

where X_i^g represents the position in the gth iteration of the i th particle, V_i^g represents the velocity in the gth iteration of the ith particle, w is a inertia weight, c_1 and c_2 are two random number in [0,1], pBest_i represents the current optimal

```
Initialize $X$ and $V$ of the PSO
Calculate the fitness of each particle
Initialize the $gBest$, $pBest_i$, $fitnessPBest$, $fitnessGBest$
while $g <= max\_iteration$ do
    while $i <= particles$ do
        Update $V_i$ and $X_i$ according to Equations (1) and (2)
        Calculate the fitness of the new particle
        if $fitness(i) < fitnessPBest$ then
            $pBest_i = X_i$
            $fitnessPBest = fitness(i)$
        end if
        if $fitness(i) < fitnessGBest$ then
            $gBest = X_i$
            $fitnessGBest = fitness(i)$
        end if
        $i = i + 1$
    end while
    $g = g + 1$
end while
```

Algorithm 1: The pseudocode of PSO.

solution, and gBset represents the historical optimal solution. The movement process of the particle swarm is shown in Figure 1.

V_1, V_2, and V_3 represent three speeds in Equation (1). V_1 represents the ωV_i^g, V_2 represents the $(\text{pBest}_i - X_i^g)$, and V_3 represents the $(\text{pBest} - X_i^g)$. V_i^{g+1} is the combination of the three speeds. The new position of X_i^{g+1} is based on X_i^g and V_i^{g+1}. The pseudocode of PSO is described in Algorithm 1.

2.2. Compact Particle Swarm Optimization. Microhardware has strict requirements on memory, and excessive memory space affects the normal use of hardware. The cPSO is proposed by Ferrante to adapt limited hardware availability [19]. The cPSO uses the distribution probability of the population to describe the position of the population. For each dimension, cPSO only uses perturbation vectors $PV^g(\mu^g, \sigma^g)$ to describe the swarm. The PV is a Gaussian distribution, μ is the mean of the PV, and σ is the standard deviation of the PV. The compact strategy produces a winner and a loser based on the comparison. μ and σ will update according to winner and loser in each iteration. The update formulas of μ and σ are shown as Equations (3) and (4).

$$\mu^{g+1} = \mu^g + \frac{1}{Np}(\text{winner} - \text{loser}), \tag{3}$$

$$\sigma^{g+1} = \sqrt{(\sigma^g)^2 + (\mu^g)^2 - (\mu^{g+1})^2 + \frac{1}{Np}\left(\text{winner}^2 - \text{loser}^2\right)}, \tag{4}$$

where Np is the virtual number of particles. A large number of experiments show that when Np is set to 300, the algorithm has the best effect. The memory usage of PSO and cPSO is shown in Table 1.

Table 1: The memory usage of PSO and cPSO.

Algorithm	Memory
PSO	Particles × dimension
cPSO	2 × dimension

Table 1 shows that the usage of PSO memory depends on the number of particle and the dimension of problem, but the usage of cPSO memory only depends on the dimension of problem. The process of compact strategy is introduced. Firstly, generate corresponding probability distribution function (PDF) according to PV vector. Then, calculate cumulative distribution function (CDF) by PDF and normalize CDF within the range of [0,1]. Then a random number will be generated to calculate a inverse cumulative distribution function (iCDF). The value of iCDF is represented as the position of the particle swarm. The equations of PDF and CDF of the Gaussian distribution are shown in

$$\text{PDF} = \frac{e^{-\left((x-\mu)^2/2\sigma^2\right)\times\sqrt{2/\pi}}}{\sigma \times \left(\operatorname{erf}\left((\mu+1)/\sqrt{2}\sigma\right) - \operatorname{erf}\left((\mu-1)/\sqrt{2}\sigma\right)\right)}, \tag{5}$$

$$\text{CDF} = \frac{\operatorname{erf}\left((\mu+1)/\sqrt{2}\sigma\right) + \operatorname{erf}\left((\mu-1)/\sqrt{2}\sigma\right)}{\operatorname{erf}\left((\mu+1)/\sqrt{2}\sigma\right) - \operatorname{erf}\left((\mu-1)/\sqrt{2}\sigma\right)}, \tag{6}$$

where the erf is the error function. The pseudocode of cPSO that combining compact strategy into PSO is shown in Algorithm 2.

2.3. Coverage of WSN in 3D. The current research problems on sensor network coverage problem mainly include 2D and 3D. Scholars study problems on 2D more than 3D. The 3D

```
Initialize the $\mu$ and $\sigma$ of $PV$
Initialize the $X$ according to $PV$
Initialize the $V$ by $V = V_{\min} + (V_{\max} - V_{\min}) \times \text{rand}$
Initialize the $gBest$, $pBest$, $fitnessPBest$, $fitnessGBest$
while $g <= \max\_iteration$ do
    $pBest$ = compact($PV(\mu, \sigma)$) according to Equations (5) and (6)
    Calculate the $fitnessPBest$ of the $pBest$
    Update $X$ and $V$ of each particle according to Equations (1) and (2)
    Calculate the fitness of the new $X$
    if $fitness(X) < fitnessPBest$ then
        $winner = X$
        $loser = pBest$
    else
        $winner = pBest$
        $loser = X$
    end if
    Update $PV$ according to Equations (3) and (4)
    $pBest = winner$
    $fitnessPBest = fitness(winner)$
    if $fitness(X) < fitnessGBest$ then
        $gBest = X$
        $fitnessGBest = fitness(X)$
    end if
    $g = g + 1$
end while
```

Algorithm 2: The pseudocode of cPSO.

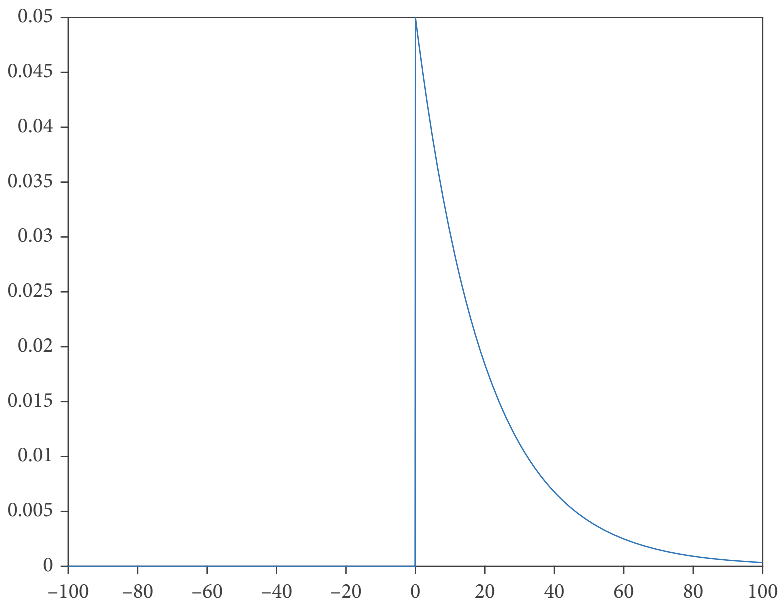
coverage problem is more in line with real life and should be the focus of research. Amulya et al. propose a modified particle swarm optimization to achieve maximum coverage in 2D. The result shows that using this algorithm could get bigger coverage than using conventional PSO [30]. Nguyen et al. proposed that different sensors have different transmission radius. They discussed how the sensors work together under different sensor transmission radius to achieve maximum coverage in 2D [31]. Huynh et al. applied the improved cuckoo algorithm to the 2D sensor coverage problem and achieved good results [32]. Osamah et al. used the bee algorithm to 2D coverage to reduce resource usage [33]. Junaid et al. proposed a method that dynamically adjust the state of the sensors to achieve maximum coverage while extending the lifetime of the sensor network [34]. Wang et al. added the idea of reverse learning to WOA to speed up the global search speed and make the sensor reach the maximum coverage faster in 2D [35].

Pan et al. improved the BH and applied it to the problem of 3D sensor network coverage with obstacles [36]. The feasibility of BH is proved by simulation experiments. However, there is still room for improvement in the optimization ability. Riham et al. proposed a new network configuration strategy for the 3D sensor network coverage problem [37]. Riham et al. proved that this strategy can better optimize the coverage problem than other similar strategies through simulation experiments. Wang et al. applied the improved virtual force algorithm to the k-coverage problem in underwater 3D and proved its feasibility through simulation experiments [38]. Cao et al. proposed a particle swarm optimizer with distributed parallelism and used it in the industrial field to achieve the 3D coverage problem with obstacle [39]. William et al. proposed a new strategy to achieve dynamic 3D coverage with limited energy using mobile sensors. Simulation experiments show that the strategy worked well [40].

3. A Novel Compact Particle Swarm Optimization

The original cPSO uses a Gaussian distribution to describe distribution statistics on the position of particle swarms. Gaussian distribution is a thin-tailed distribution. The biggest problem with this is that using a Gaussian distribution can easily lead to getting stuck in a local optimal solution. Considering this deficiency, we propose a novel compact particle swarm optimization (ncPSO). We use the Pareto distribution instead of the Gaussian distribution to express the position of particle swarms. Pareto distribution is a heavy-tailed distribution. It falls more slowly than Gaussian distribution for $x \longrightarrow \infty$. So the Pareto distribution has a greater probability of taking values that deviate from the normal level. This feature makes it fall into local optimum difficult in the search. The formulas of PDF and CDF of the Pareto are shown in

$$\text{PDF} = \frac{1}{\sigma}\left(1 + k\frac{(x-\theta)}{\sigma}\right)^{-1-1/k}, \tag{7}$$


(a) PDF of Pareto distribution

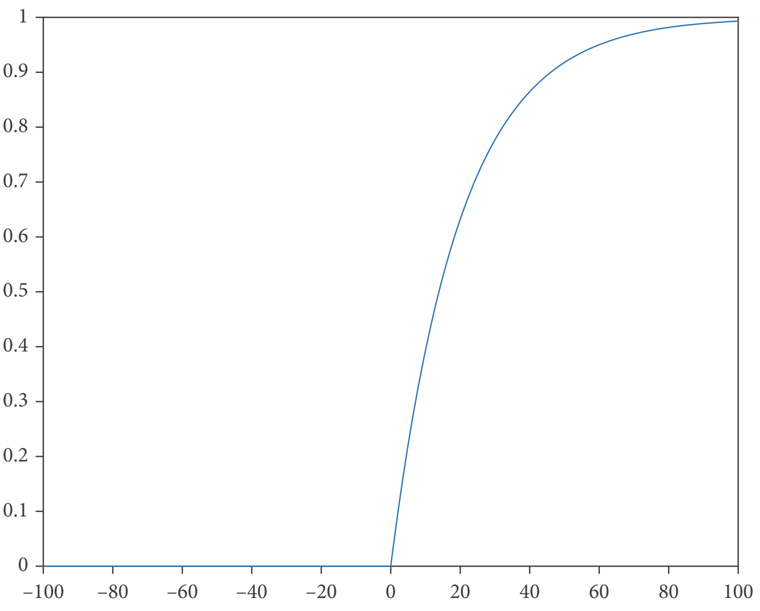

(b) CDF of Pareto distribution

Figure 2: PDF and CDF of Pareto distribution.

$$\mathrm{CDF} = \begin{cases} 1 - \left(1 + k\dfrac{(x-\theta)}{\sigma}\right)^{-1/k}, & k \neq 0, \\ 1 - \exp\left(-\dfrac{(x-\theta)}{\sigma}\right), & k = 0. \end{cases} \quad (8)$$

The generalized Pareto distribution has three parameters: threshold parameter θ, scale parameter σ, and shape parameter k. The figures of Pareto PDF and Pareto CDF are shown in Figure 2. The descending speed of the right tail of the Pareto distribution is significantly slower than that of the Gaussian distribution.

The ncPSO uses uniform distribution function to randomly generate the initial positions and initial velocities of particles within a feasible solution range. Because the Pareto distribution is one-sided function, we also generate a symmetric position on the other side at each iteration, then compare the two positions and take the better one as the next pBest. Because the Pareto distribution is a heavy-tailed distribution, it may occur that the CDF value does not reach 1 within the feasible solution range, resulting in the value of the inverse cumulative distribution function out of bounds. We solve this problem by regenerating a new random numbers to generate iCDF. Among the three

```
Initialize the $\theta$ and $\sigma$ of Pareto
Initialize the $X$ by uniform distribution
Initialize the $V$ by uniform distribution
Initialize the $gBest$, $pBest$, $fitnessPBest$, $fitnessGBest$
while $g <= \max\_iteration$ do
    $pBestR$ = novlel_compact($\theta$,$\sigma$) according to Equations (7) and (8)
    $pBestL = 2 \times pBest - pBestL$
    $pBest$ = compare(fitness($pBestR$),fitness($pBestL$))
    Calculate the $fitnessPBest$ of the $pBest$
    Update $X$ and $V$ of each particle according to Equations (1) and (2)
    Calculate the fitness of the new $X$
    [$loser$, $winner$]=compare(fitness($X$),fitness($pBest$))
    Update $\theta$ and $\sigma$ according to Equations (10) and (11)
    $pBest = winner$
    $fitnessPBest = fitness(winner)$
    Randomly choose a dimension $d$
    Gaussian perturbation for the $d-th$ dimension of $X$ by Equation (9)
    Calculate the fitness of the new $X$
    if $fitness(X) < fitnessGBest$ then
        $gBest = X$
        $fitnessGBest = fitness(X)$
    end if
    $g = g + 1$
end while
```

Algorithm 3: The pseudocode of ncPSO.

Table 2: Comparison of memory and function calls of different algorithms.

Algorithm	Dimension	Population	Memory	Function calls
PSO	D	N	$N \times D$	$N \times$ max_iteration
ABC	D	N	$N \times D$	$2 \times N \times$ max_iteration
WOA	D	N	$N \times D$	$N \times$ max_iteration
BH	D	N	$N \times D$	$N \times$ max_iteration
ncPSO	D	1	$2 \times D$	$3 \times$ max_iteration

parameters of the Pareto distribution, $k > 0$ is guaranteed, θ determines the convergence position, and σ determines the convergence speed. The ncPSO no longer restricts that the position of particle must be normalized to [-1,1], and the position of the particle is obtained directly in feasible solution space.

It is possible that only one dimension falling into the local optimum makes bad result of whole swarm. The ncPSO selects one dimension of the particle randomly to perform a Gaussian perturbation to better avoid the algorithm falling into local optimum in each iteration. The perturbation formula is shown in

$$X^d = \text{normrnd}\left(X^d, C\right) \tag{9}$$

where the C is a constant and determined by the actual problem to be solved. After Gaussian perturbation, the particle moves according to Equations (1) and (2) to a new position. Then, ncPSO compares the position of pBest and new position of particle to generate a winner and loser to update

Table 3: Parameter settings of different algorithms.

Algorithm	Parameters
PSO	$D = 50$, $N = 40$, $w = 0.9$, $r_1 = 2.0$, $r_2 = 2.0$
ABC	$D = 50$, $N = 40$, limit = 100
WOA	$D = 50$, $N = 40$, probability switch = 0.5
BH	$D = 50$, $N = 40$
ncPSO	$D = 50$, $N = 40$, $r_1 = 2.0$, $r_2 = 2.0$

θ and σ of the Pareto distribution. The update formulas of θ and σ are shown as

$$\theta^{g+1} = \sigma^g + \frac{1}{Np}(\text{winner} - \text{loser}), \tag{10}$$

$$\sigma^{g+1} = \sqrt{(\sigma^g)^2 + (\theta^g)^2 - \left(\theta^{g+1}\right)^2 + \frac{1}{Np}\left(\text{winner}^2 - \text{loser}^2\right)}. \tag{11}$$

Table 4: Comparison of performance testing and Wilcoxon's sign test of conventional algorithms.

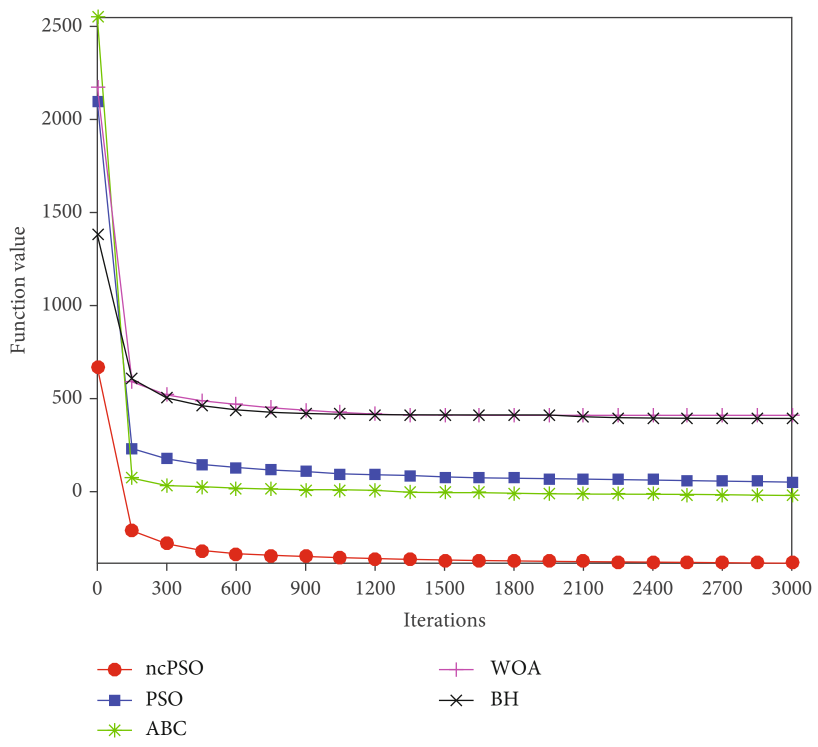
	PSO		ABC		WOA		BH		ncPSO
$f1$	$-1.33E+03$	>	$4.06E+03$	>	$-1.34E+03$	>	$-1.40E+03$	=	$-1.40E+03$
$f2$	$8.83E+06$	>	$2.29E+08$	>	$8.07E+07$	>	$2.70E+07$	>	$1.74E+06$
$f3$	$2.65E+09$	>	$2.45E+13$	>	$3.47E+10$	>	$4.46E+09$	>	$1.04E+09$
$f4$	$1.11E+03$	<	$8.59E+04$	>	$5.90E+04$	<	$3.09E+04$	<	$8.12E+04$
$f5$	$-9.76E+02$	>	$1.22E+03$	>	$-7.96E+02$	>	$-8.84E+02$	>	$-1.00E+03$
$f6$	$-8.04E+02$	>	$-4.18E+01$	>	$-6.79E+02$	>	$-8.16E+02$	>	$-8.23E+02$
$f7$	$-6.76E+02$	>	$1.58E+03$	>	$9.12E+02$	>	$-6.13E+02$	>	$-6.84E+02$
$f8$	$-6.79E+02$	=	$-6.79E+02$	=	$-6.79E+02$	=	$-6.79E+02$	=	$-6.79E+02$
$f9$	$-5.44E+02$	>	$-5.54E+02$	>	$-5.30E+02$	>	$-5.33E+02$	>	$-5.44E+02$
$f10$	$-4.43E+02$	>	$6.41E+02$	>	$-1.79E+02$	>	$-4.69E+02$	>	$-4.99E+02$
$f11$	$5.13E+01$	>	$-2.05E+01$	>	$4.10E+02$	>	$3.93E+02$	>	$-3.84E+02$
$f12$	$2.29E+02$	<	$1.79E+02$	<	$6.05E+02$	>	$5.38E+02$	>	$2.73E+02$
$f13$	$4.06E+02$	>	$2.73E+02$	<	$7.97E+02$	>	$6.19E+02$	>	$3.51E+02$
$f14$	$6.47E+03$	>	$5.33E+03$	>	$9.10E+03$	>	$8.41E+03$	>	$1.84E+03$
$f15$	$8.53E+03$	<	$1.31E+04$	>	$1.15E+04$	>	$9.62E+03$	>	$9.55E+03$
$f16$	$2.03E+02$	>	$2.03E+02$	>	$2.03E+02$	>	$2.02E+02$	=	$2.02E+02$
$f17$	$7.99E+02$	>	$6.56E+02$	>	$1.45E+03$	>	$1.32E+03$	>	$3.51E+02$
$f18$	$8.68E+02$	<	$9.56E+02$	<	$1.56E+03$	>	$1.45E+03$	>	$9.85E+02$
$f19$	$5.30E+02$	>	$5.04E+04$	>	$6.81E+02$	>	$6.25E+02$	>	$5.02E+02$
$f20$	$6.24E+02$	<	$6.22E+02$	<	$6.25E+02$	>	$6.24E+02$	=	$6.25E+02$
$f21$	$1.67E+03$	>	$2.72E+03$	>	$1.93E+03$	>	$1.65E+03$	>	$1.64E+03$
$f22$	$1.02E+04$	>	$6.75E+03$	>	$1.29E+04$	>	$1.23E+04$	>	$4.31E+03$
$f23$	$1.20E+04$	>	$1.45E+04$	>	$1.41E+04$	>	$1.30E+04$	<	$1.40E+04$
$f24$	$1.38E+03$	>	$1.35E+03$	<	$1.41E+03$	>	$1.43E+03$	>	$1.36E+03$
$f25$	$1.54E+03$	<	$1.47E+03$	<	$1.53E+03$	<	$1.52E+03$	<	$1.60E+03$
$f26$	$1.65E+03$	>	$1.61E+03$	>	$1.67E+03$	>	$1.64E+03$	>	$1.48E+03$
$f27$	$3.32E+03$	>	$3.04E+03$	<	$3.56E+03$	>	$3.60E+03$	>	$3.32E+03$
$f28$	$4.44E+03$	<	$3.49E+03$	<	$8.75E+03$	>	$8.12E+03$	>	$5.16E+03$
>/=/<	19/1/8		19/1/8		25/1/2		21/3/4		

After updating θ and σ, the pBest will be generated. As the number of iterations increases, σ gets smaller and smaller, and the ncPSO converges at the convergence position θ faster and faster.

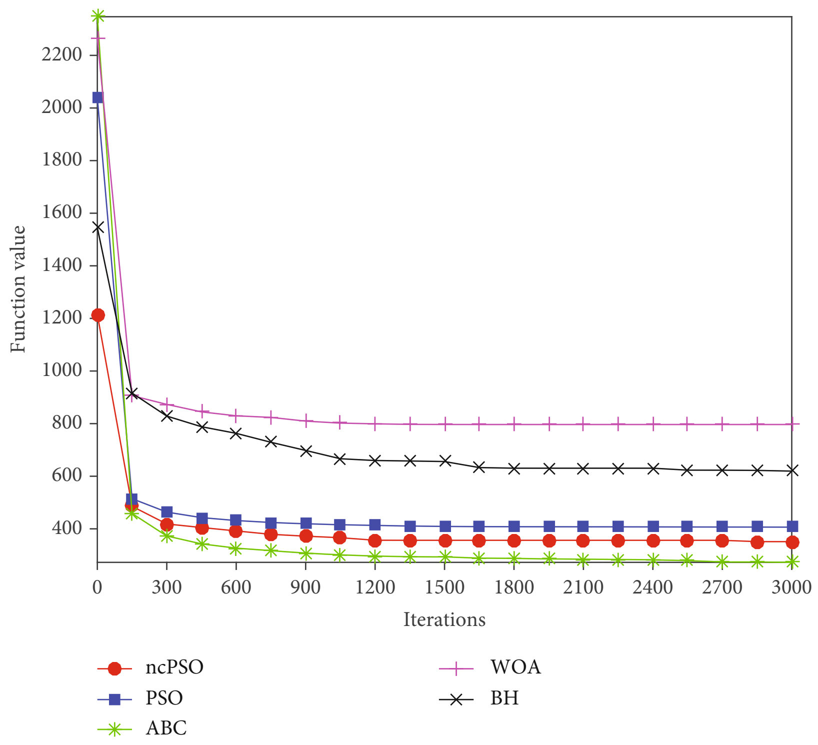
The difference between ncPSO and other algorithms is that ncPSO uses Pareto distribution of heavy-tailed distribution to describe the position of the whole particle swarm instead of Gaussian distribution. The advantage of using Pareto distribution is that it can reduce the probability of the algorithm falling into local optimum. Another difference is that ncPSO uses mathematical distribution to represent the position of particle swarm instead of storing the position of each particle, which greatly reduces the use of memory units. Taking 30 particles and 40 dimensions as an example, the traditional PSO needs $40 \times 30 = 1200$ memory units to store the position of each particle in each dimension, while ncPSO only needs $40 \times 2 = 80$ memory units to express the position of the whole particle swarm. The ncPSO has a good applicability to microdevices, and the excellent optimization performance of ncPSO is a new optimization method for 3D coverage of WSN. The pseudocode of ncPSO is shown in Algorithm 3.

4. The Performance Tests of Algorithms

We mainly compare the performance of ncPSO with conventional heuristic algorithms and other compact algorithms in this section. The CEC2013 test functions are used for performance test. The functions in CEC2013 are diverse and very convincing [41]. The 28 functions in CEC2013 are represented by $f1-f28$, respectively. $f1-f5$ are 5 unimodal functions, $f6-f20$ are 15 multimodal functions, and $f21-f28$ are 8 composition functions.

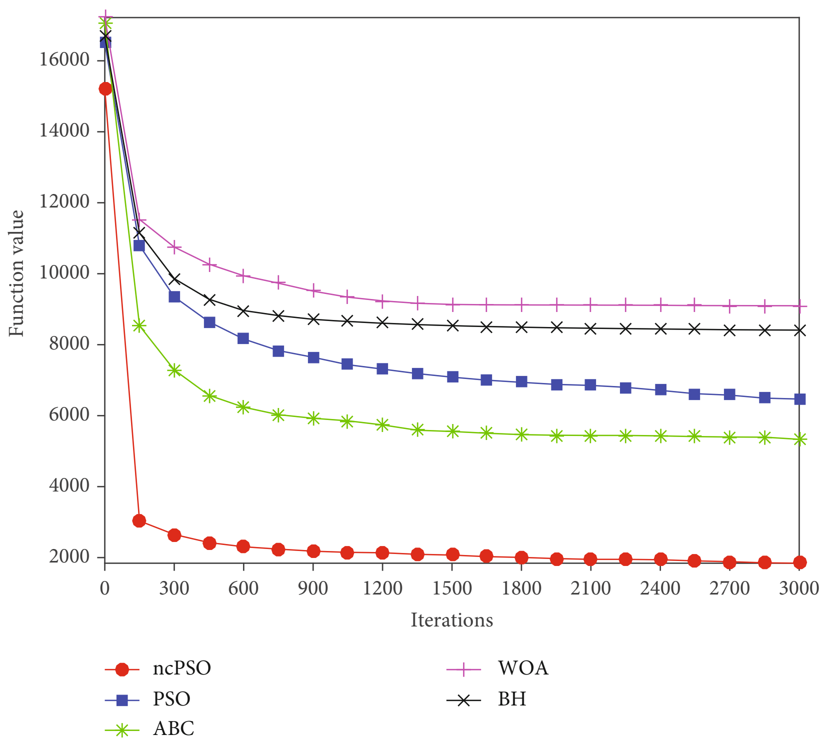


(a) $f11$

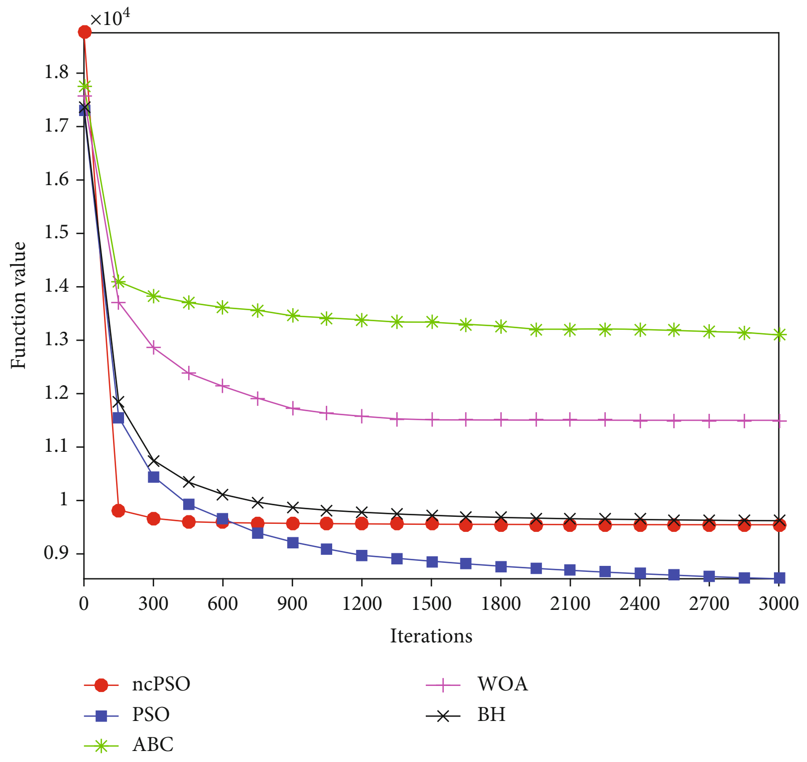


(b) $f13$

Figure 3: Continued.

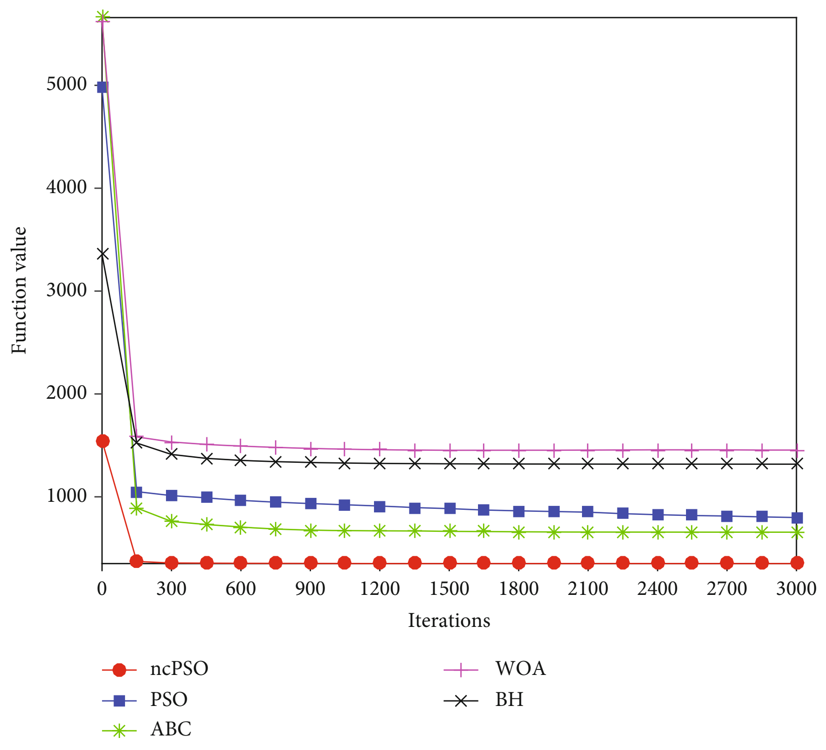


(c) *f*14

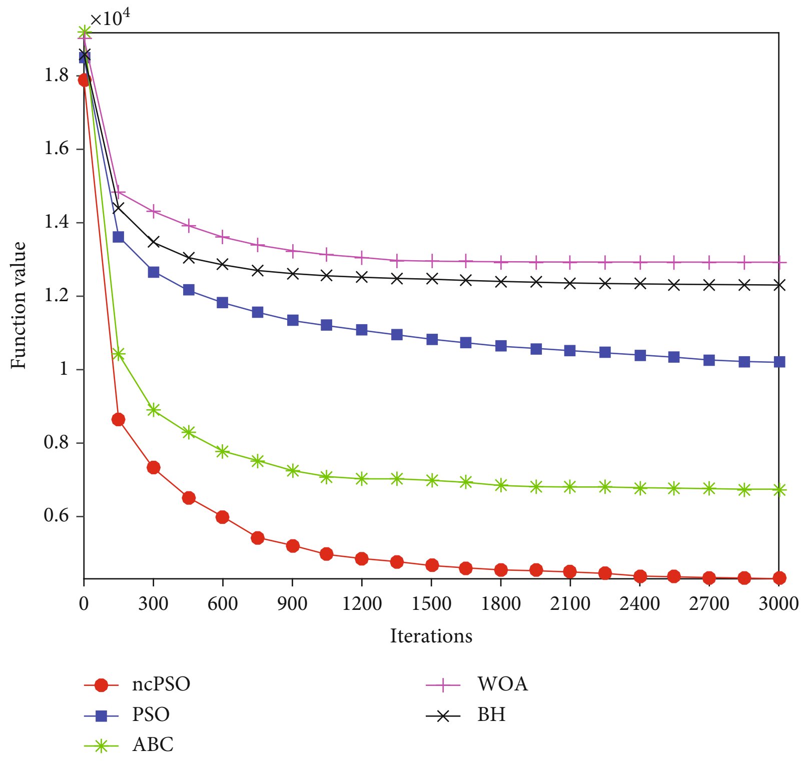


(d) *f*15

Figure 3: Continued.

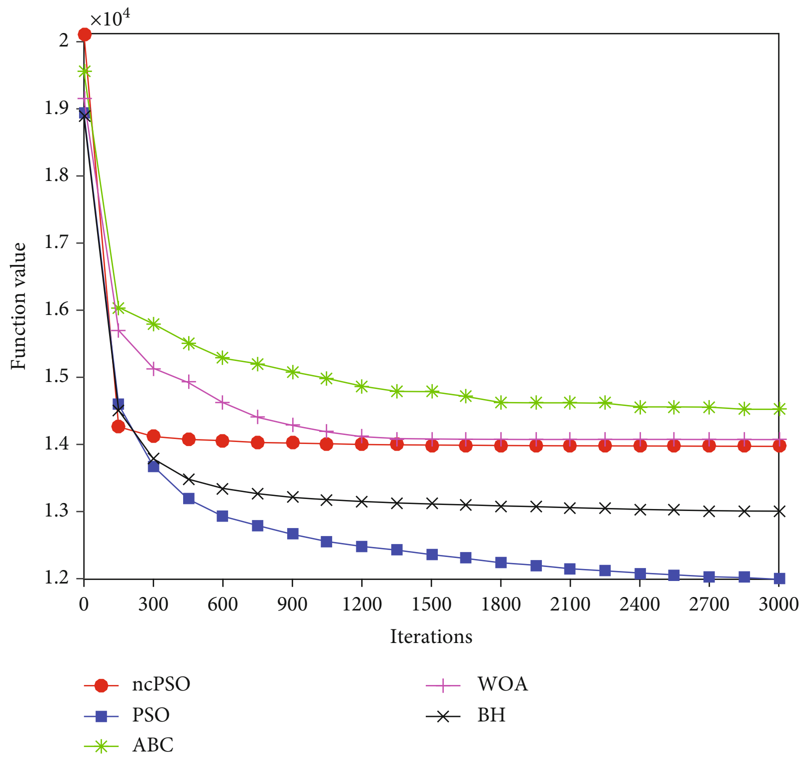


(e) $f17$

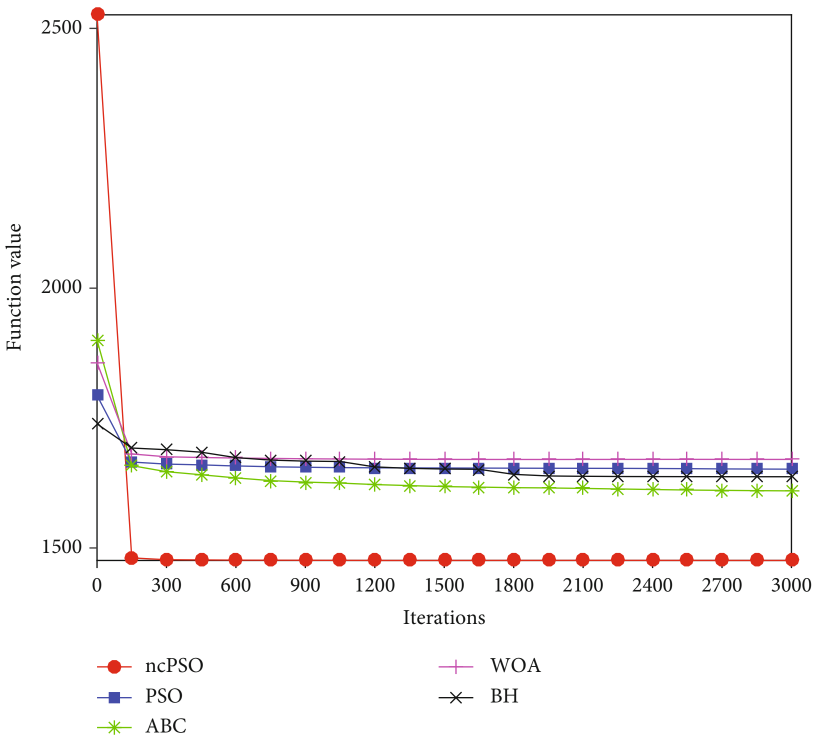


(f) $f22$

Figure 3: Continued.



(g) $f23$



(h) $f26$

Figure 3: The convergence of some test functions of conventional algorithms.

4.1. Comparison between ncPSO and Conventional Heuristic Algorithms. Firstly, we compare ncPSO with conventional heuristic algorithms. We choose PSO, ABC, WOA, and BH. All algorithms are run 20 times on each test function, and then, the average is calculated as the experiment result. Simultaneously Wilcoxon's sign rank test is taken with a significant level $\alpha = 0.05$. The memory and function calls of the algorithms are shown in Table 2.

Table 5: The parameter settings of all compact algorithms.

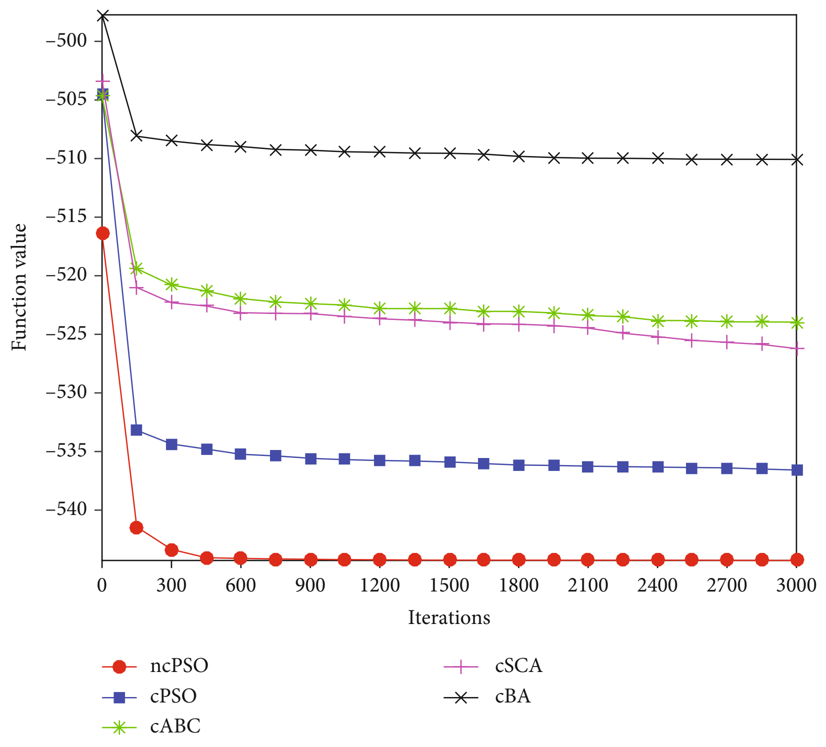
Algorithm	Parameters
cPSO	$D = 50$, virtual Np = 300, $w = 0.2$, $r_1 = -0.07$, $r_2 = 3.74$
cABC	$D = 50$, virtual Np = 300, limit = 100, food number = 3
cSCA	$D = 50$, virtual Np = 300, probability switch = 0.5
cBA	$D = 50$, virtual Np = 300, loudness rate = 0.5, pulse rate = 0.5, $f_{\min} = 1$, $f_{\max} = 2$
ncPSO	$D = 50$, virtual Np = 300, $r_1 = 2.0$, $r_2 = 2.0$

Table 6: Comparison of performance testing and Wilcoxon's sign test of compact algorithms.

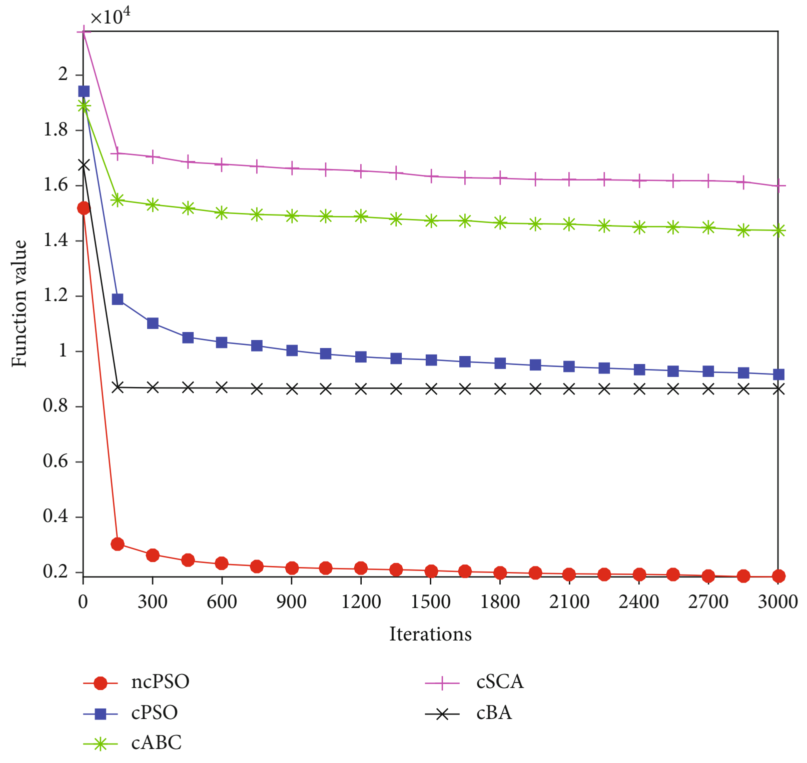
	cABC		cPSO		cBA		cSCA		ncPSO
$f1$	$2.42E+04$	>	$-1.39E+03$	>	$-1.40E+03$	=	$5.33E+04$	>	$-1.40E+03$
$f2$	$6.92E+08$	>	$2.71E+07$	>	$4.24E+06$	>	$7.31E+08$	>	$1.74E+06$
$f3$	$2.33E+11$	>	$5.53E+09$	>	$4.82E+09$	>	$1.88E+15$	>	$1.04E+09$
$f4$	$1.42E+05$	>	$4.80E+03$	<	$2.81E+05$	>	$1.81E+05$	>	$8.12E+04$
$f5$	$5.79E+03$	>	$-9.91E+02$	>	$-1.00E+03$	=	$1.43E+04$	>	$-1.00E+03$
$f6$	$1.34E+03$	>	$-8.23E+02$	=	$-8.51E+02$	<	$3.01E+03$	>	$-8.23E+02$
$f7$	$-4.44E+02$	>	$-5.90E+02$	>	$3.63E+12$	>	$1.48E+04$	>	$-6.84E+02$
$f8$	$-6.79E+02$	=	$-6.79E+02$	=	$-6.79E+02$	=	$-6.79E+02$	=	$-6.79E+02$
$f9$	$-5.24E+02$	>	$-5.37E+02$	>	$-5.10E+02$	>	$-5.26E+02$	>	$-5.44E+02$
$f10$	$4.37E+03$	>	$-4.75E+02$	>	$-4.99E+02$	=	$7.01E+03$	>	$-4.99E+02$
$f11$	$4.47E+02$	>	$3.16E+02$	>	$2.47E+03$	>	$4.60E+02$	>	$-3.84E+02$
$f12$	$7.07E+02$	>	$4.95E+02$	>	$7.91E+03$	>	$9.23E+02$	>	$2.73E+02$
$f13$	$8.51E+02$	>	$5.27E+02$	>	$6.87E+03$	>	$9.93E+02$	>	$3.51E+02$
$f14$	$1.44E+04$	>	$9.17E+03$	>	$8.67E+03$	>	$1.60E+04$	>	$1.84E+03$
$f15$	$1.51E+04$	>	$9.98E+03$	>	$1.03E+04$	>	$1.54E+04$	>	$9.55E+03$
$f16$	$2.04E+02$	>	$2.03E+02$	>	$2.01E+02$	<	$2.04E+02$	>	$2.02E+02$
$f17$	$1.95E+03$	>	$1.03E+03$	>	$1.66E+04$	>	$1.46E+03$	>	$3.51E+02$
$f18$	$2.32E+03$	>	$1.07E+03$	>	$1.58E+04$	>	$1.59E+03$	>	$9.85E+02$
$f19$	$6.32E+04$	>	$5.51E+02$	>	$8.67E+02$	>	$1.83E+04$	>	$5.02E+02$
$f20$	$6.25E+02$	=	$6.24E+02$	<	$6.25E+02$	=	$6.25E+02$	=	$6.25E+02$
$f21$	$5.69E+03$	>	$1.58E+03$	<	$1.61E+03$	<	$5.17E+03$	>	$1.64E+03$
$f22$	$1.66E+04$	>	$1.22E+04$	>	$1.11E+04$	>	$1.79E+04$	>	$4.31E+03$
$f23$	$1.71E+04$	>	$1.28E+04$	<	$1.34E+04$	<	$1.69E+04$	>	$1.40E+04$
$f24$	$1.42E+03$	>	$1.43E+03$	>	$1.67E+03$	>	$1.51E+03$	>	$1.36E+03$
$f25$	$1.59E+03$	<	$1.56E+03$	<	$1.64E+03$	>	$1.58E+03$	<	$1.60E+03$
$f26$	$1.64E+03$	>	$1.62E+03$	>	$1.40E+03$	<	$1.71E+03$	>	$1.48E+03$
$f27$	$3.66E+03$	>	$3.42E+03$	>	$5.29E+03$	>	$3.82E+03$	>	$3.32E+03$
$f28$	$7.13E+03$	>	$3.48E+03$	<	$8.61E+04$	>	$8.68E+03$	>	$5.16E+03$
>/=/<	25/2/1		20/2/6		18/5/5		25/2/1		

In Table 2, we can see that ncPSO requires less memory and function calls than other algorithms. Table 3 shows the parameter settings of all algorithms.

The test results of these algorithms on CEC2013 and Wilcoxon's sign test results are shown in Table 4. The symbol ">" indicates that the performance of ncPSO is better than other algorithms. The symbol "=" indicates that the performance of ncPSO is as good as other algorithms. The symbol "<" indicates that the performance of ncPSO is worse than other algorithms.

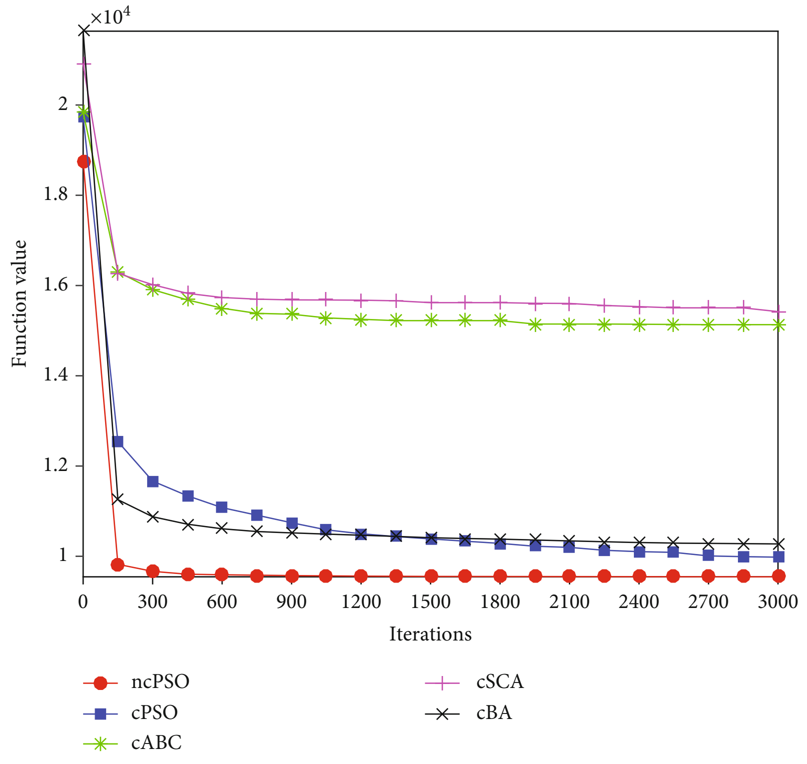


(a) *f*9

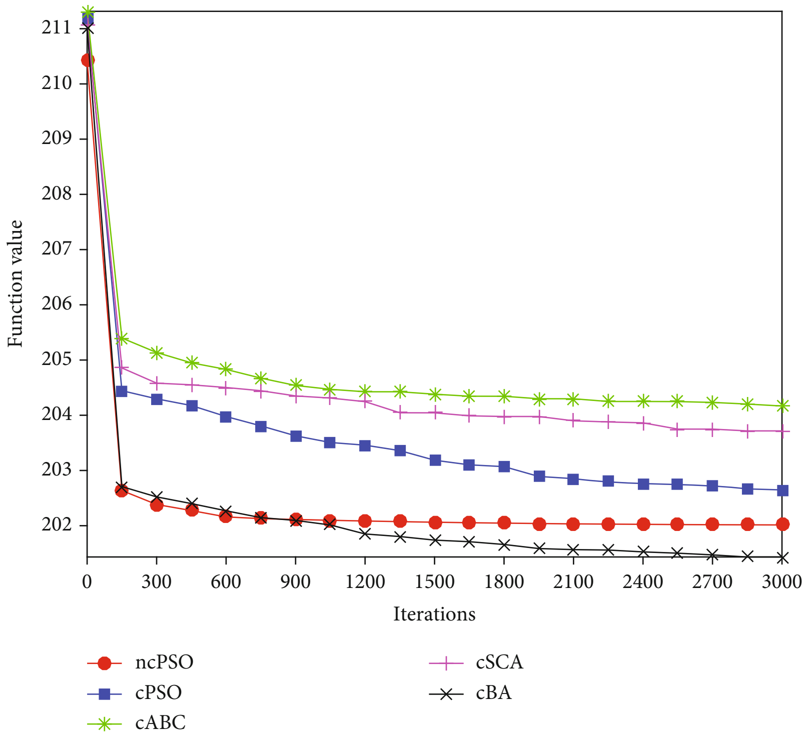


(b) *f*14

FIGURE 4: Continued.

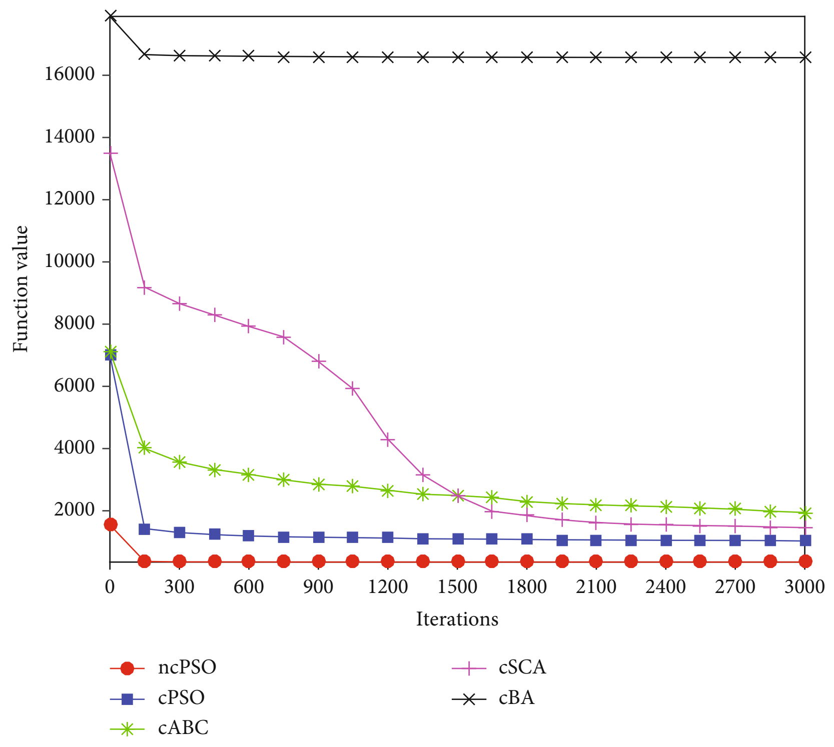


(c) *f*15

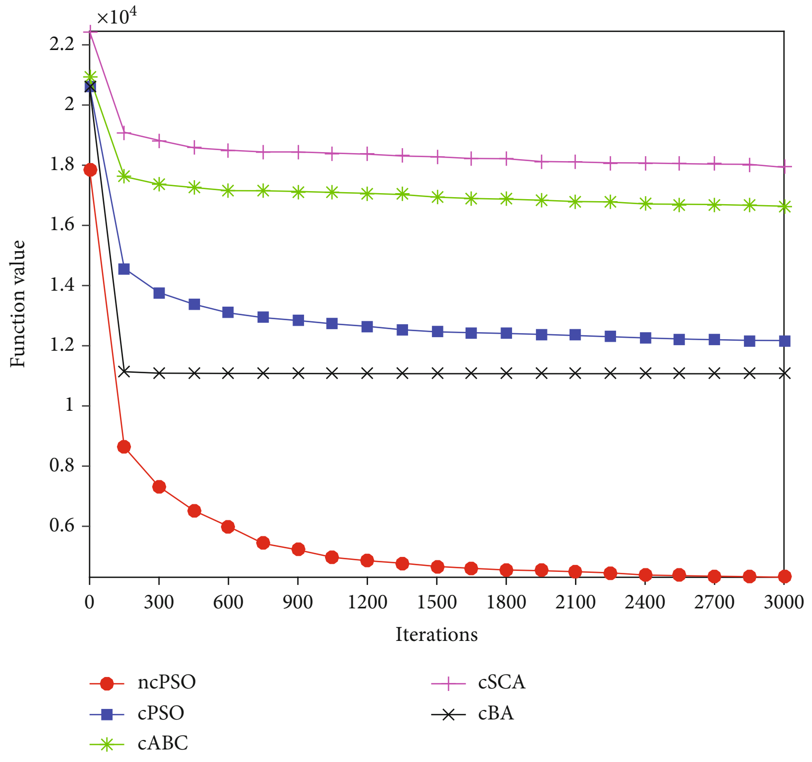


(d) *f*16

Figure 4: Continued.

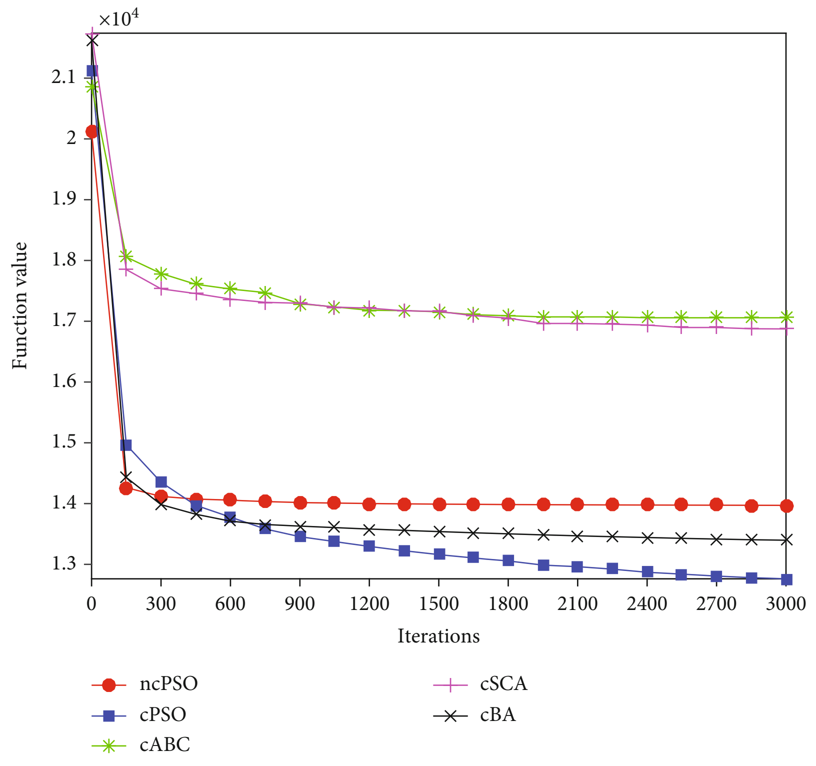


(e) $f17$

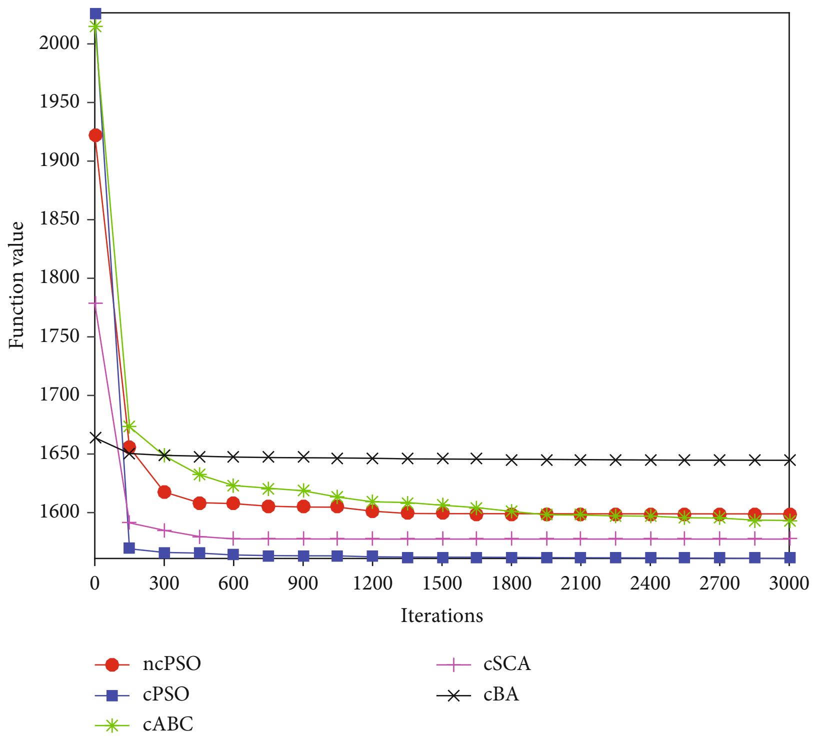


(f) $f22$

Figure 4: Continued.



(g) $f23$



(h) $f25$

FIGURE 4: The convergence of some test functions of compact algorithms.

According to the data from Table 4, we can see that the ncPSO has better performance than PSO on 19 functions. And compared with ABC, the ncPSO achieves better results on 19 functions too. The WOA is better than ncPSO in only two functions. The ncPSO performs better than BH on 21 functions. Because some images have very

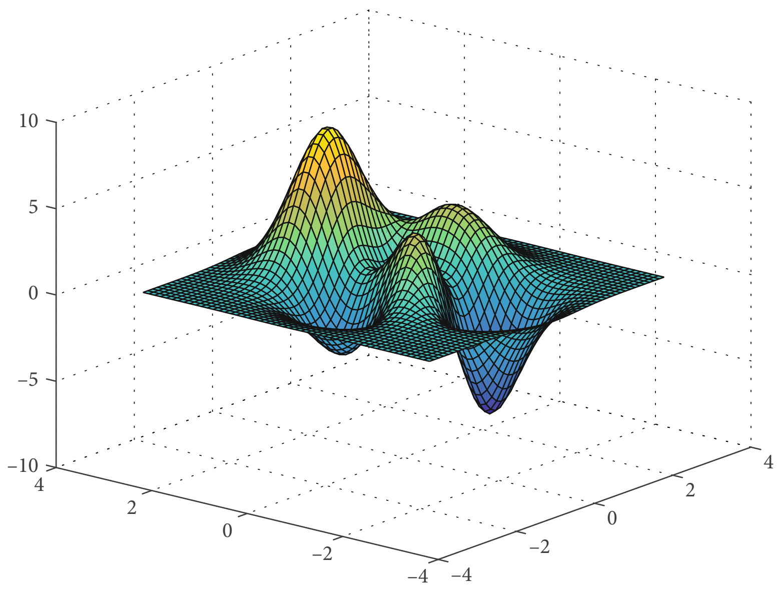


FIGURE 5: The virtual environment in 3D for simulation experiments.

TABLE 7: The form of particle to represent position of nodes.

Dimension	1	2	......	$2N-1$	$2N$
Sensor	$\text{Sensor}_{1,1}$	$\text{Sensor}_{1,2}$	......	$\text{Sensor}_{N,1}$	$\text{Sensor}_{N,2}$

little discrimination, we select a part of representative images of the convergence process for display. The convergence of some test functions is shown in Figure 3.

We can see that ncPSO has faster convergence speed and better ability to find optimization on $f11, f14, f17, f22, f26$. The ncPSO does not have a better optimization ability than ABC on $f13$. Although the optimization ability of ncPSO is not as good as that of PSO, the convergence speed of ncPSO is faster on $f15$ and $f23$. The BH performs better than ncPSO on $f23$.

4.2. Comparison between ncPSO and Other Compact Algorithms. We compare ncPSO with other compact algorithms in this subsection. We choose cPSO, cABC, cSCA, and cBA. We also performed 20 experiments for each test function and also take Wilcoxon's sign rank test. The memory and function calls are the same for all these compact algorithms. Table 5 shows the parameter settings of all algorithms.

Table 6 shows the test results of the algorithm on CEC2013 and Wilcoxon's sign test results. According to Table 6, we can find that ncPSO performs better than cPSO on 20 functions. The cABC and cSCA are only better than ncPSO on $f25$. The ncPSO has better performance than cBA on 18 functions and has same performance with cBA on 5 functions.

We also select some representative images of the convergence process for display in Figure 4. According to the figures from Figure 4, we can find that ncPSO has faster convergence speed and better ability to find optimization on $f9, f14, f15, f17, f22$. The convergence ability of ncPSO is not as good as cBA in $f16$, but it is better than cABC, cSCA, and cPSO. On the contrary, the convergence ability of ncPSO is better than cBA on $f25$, but not as good as cABC, cSCA, and cPSO. On $f23$, the convergence ability of ncPSO is worse than that of cPSO and cBA, but better than that of cSCA and cABC.

5. Application in Optimizing Coverage of 3D in Wireless Sensor Network

The problem of sensor coverage in 3D environment is more complicated than that in 2D. Compared with 2D coverage, 3D coverage needs to consider the influence of terrain on coverage effect. The complicated terrain makes it difficult for sensors to collect information. We use the 3D model generated by "peak" function in MATLAB as the environment for simulation experiments. The control variables of the experiment in this section are the 3D simulation environment and the communication radius of sensors. The simulation environment is shown in Figure 5. The communication radius of sensors are set to 5 m.

In this simulation environment, as long as the first and second coordinates are known, the height in the 3D environment can be known, and then, the position of the sensor can be known. The ncPSO sets the dimension of particle swarm to twice the number of sensors. Two dimensions represent two coordinates of a sensor, and then, the value of the third dimension of the sensor in the terrain can be calculated by

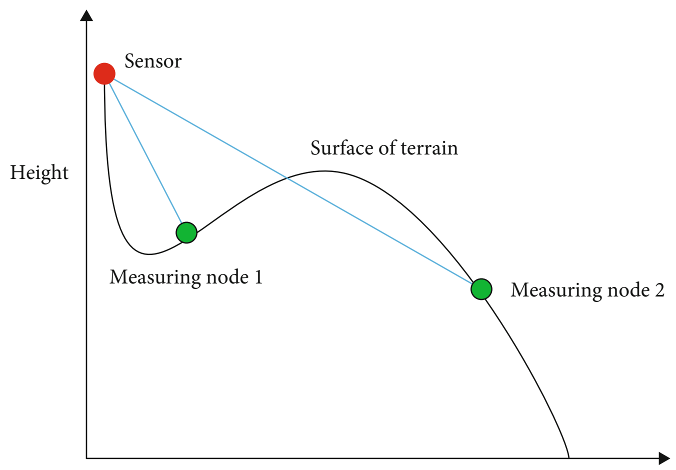


Figure 6: Obstacle model in sensor networks.

Table 8: The simulation results of maximum coverage rate.

Sensor number	30		40		50		60	
	Average	std	Average	std	Average	std	Average	std
PSO	47.53%	*0.0073*	56.87%	0.0140	63.94%	*0.0095*	70.44%	0.0079
GA	46.64%	0.0077	55.94%	0.0078	64.44%	0.0107	70.24%	*0.0061*
WOA	48.04%	0.0086	57.56%	0.0139	65.74%	0.0132	71.44%	0.0124
BH	49.25%	0.0135	58.43%	*0.0070*	66.16%	0.0128	72.47%	0.0144
ncPSO	*55.07%*	0.0111	*65.46%*	0.0125	*72.65%*	0.0118	*79.02%*	0.0154
cPSO	48.86%	0.0112	58.87%	0.0098	66.50%	0.0124	72.83%	0.0116
cABC	47.08%	0.0114	56.38%	0.0085	64.42%	0.0100	70.31%	0.0085
cBA	44.06%	0.0146	53.76%	0.0133	61.05%	0.0220	68.28%	0.0226
cSCA	43.32%	0.0096	51.97%	0.0098	61.34%	0.0207	66.42%	0.0150

the information of the topographic map, so that the position of the sensor can be obtained. The particle of ncPSO can be represented in the form of Table 7.

There is a monitoring obstacle problem in the 3D problem. The existence of obstacles makes it impossible for the sensor to pass through the obstacles to collect information about the location to be measured. If the terrain between the sensor and the monitoring node is higher than the line connecting the two points, the signal will be blocked and the monitoring point will not be monitored. The obstacle model in WSN is shown in Figure 6.

We suppose that nodes 1 and 2 are both in the radius of sensor in Figure 6; it can been seen that node 1 could be covered but node 2 could not be covered due to the obstacle. The judgment formula for whether the monitoring point is covered is shown in Equation (12). The formula for the coverage rate of the whole environment is shown in Equation (13).

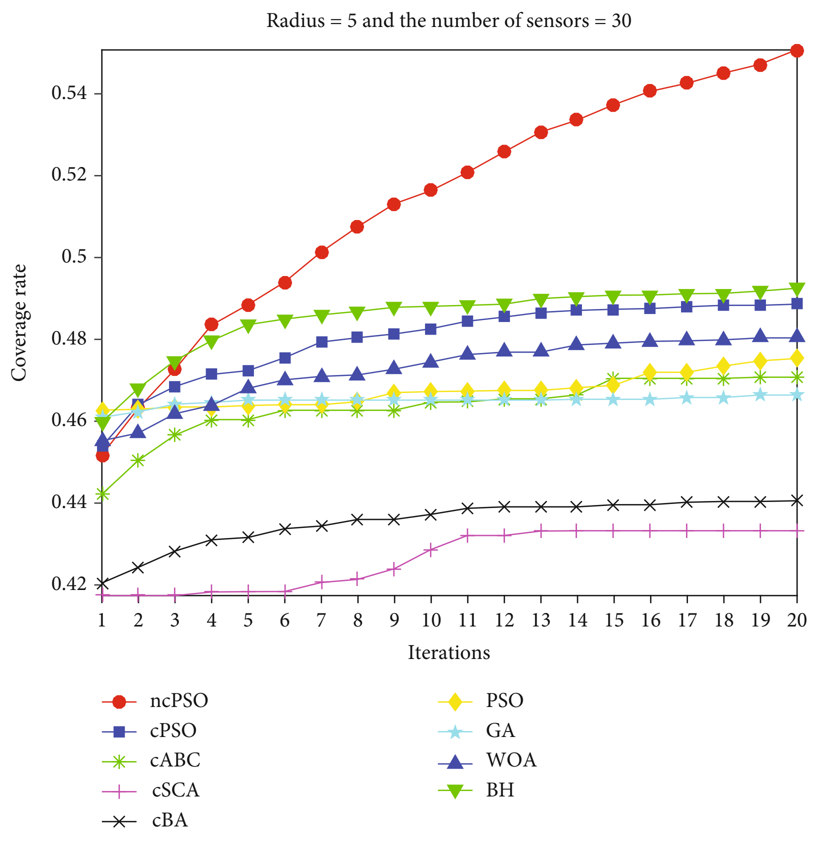
$$\text{Coverage}(s, n) = \begin{cases} 1, & \text{distance}(s, n) \leq \text{radius and there no obstacle,} \\ 0, & \text{distance}(s, n) > \text{radius or there are obstacles,} \end{cases} \tag{12}$$

$$\text{Coverage rate} = \frac{1}{M} \cdot \sum_{i=1}^{M} \left(\sum_{j=1}^{N} \text{Coverage}\left(s_j, n_i\right) \right), \tag{13}$$

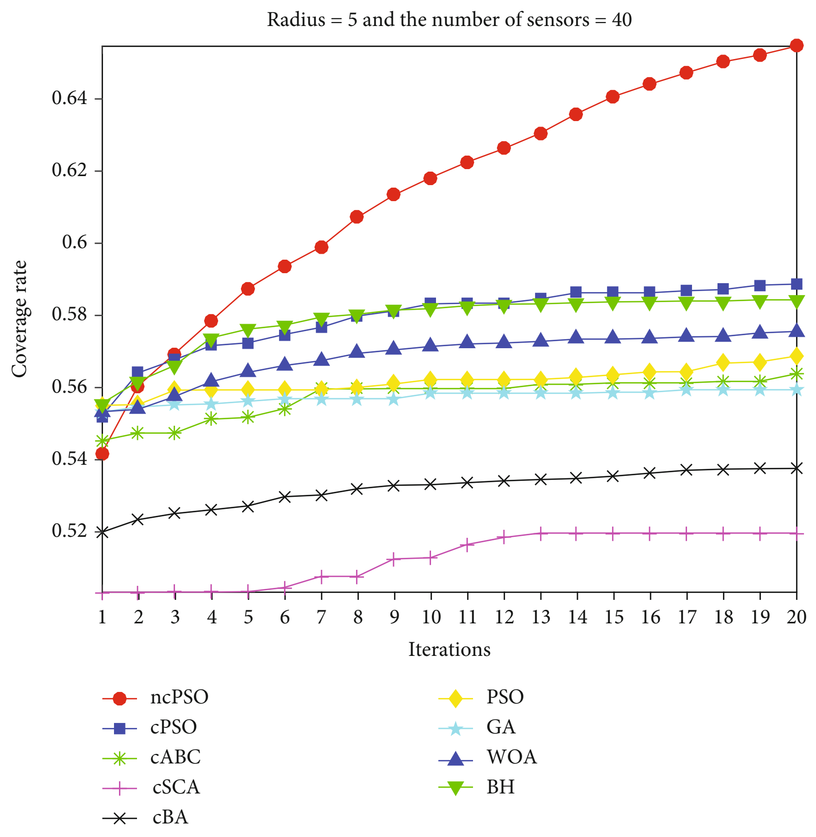
where s represents the sensor and n represents the measured node, N represents the total number of sensors, and M represents the total measured nodes. Then, we will carry out simulation experiments on the two problems of achieving the maximum coverage under a certain node and the minimum number of sensors to achieve a certain coverage.

5.1. The Maximum Coverage Rate under a Certain Number of Sensor Nodes

Different algorithms are used to optimize the problem of maximum coverage rate under a certain number of sensor nodes in this subsection. The radius of sensors is set to 5 m, and the number of sensors is set to 30, 40, 50, and 60, respectively. When the number of sensors is the same, different algorithms are used to optimize the coverage rate. The larger the coverage rate, the better the solution effect of the algorithm. Each group of experiments is carried out 10 times to ensure the accuracy of experiments, and the mean value is used as the experimental result. At the same time, we also calculate the standard deviation. Table 8 shows the results of simulation experiments.

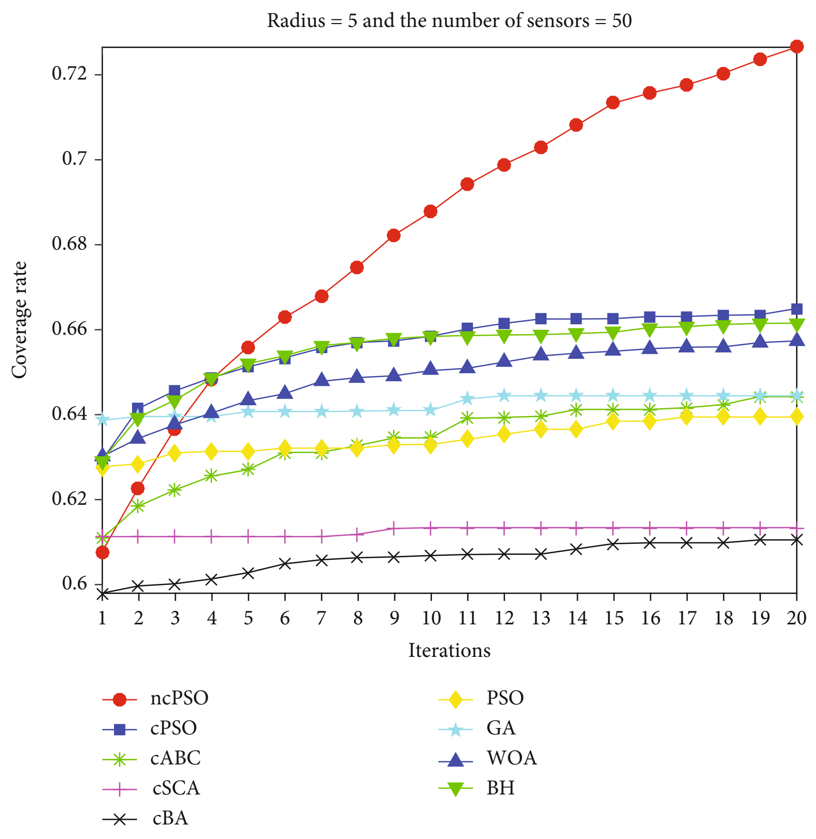


(a) Sensor number = 30

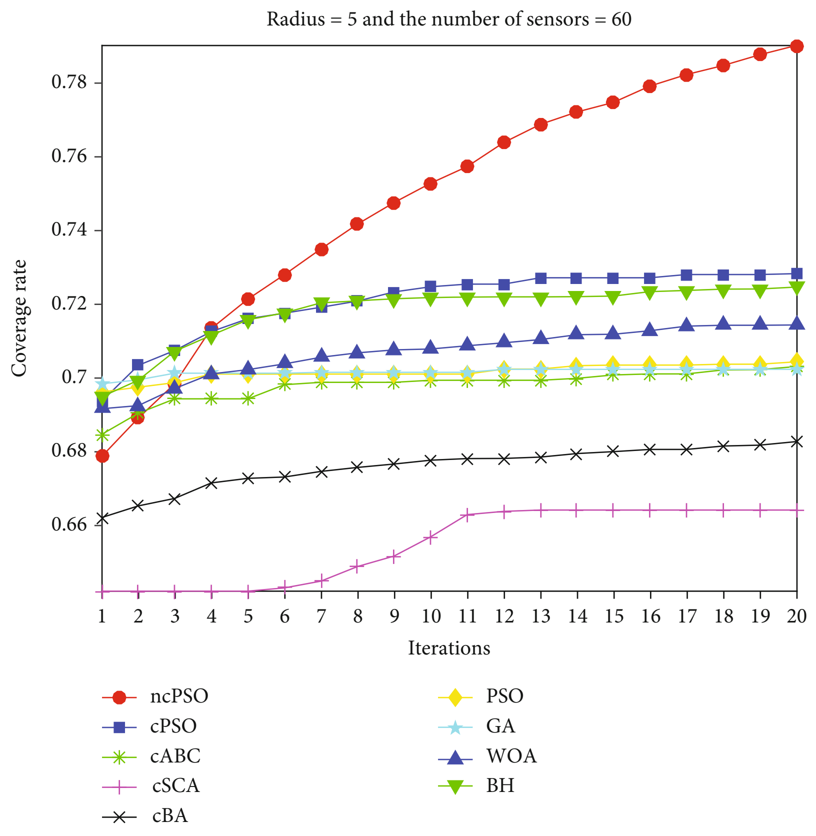


(b) Sensor number = 40

Figure 7: Continued.



(c) Sensor number = 50



(d) Sensor number = 60

Figure 7: The convergence process of algorithms.

Table 9: The simulation results of maximum coverage rate.

Radius = 5	70%		80%	
	Average	std	Average	std
PSO	60.00	*1.05*	81.60	3.10
GA	59.90	1.37	80.80	*1.55*
WOA	58.20	1.81	79.30	3.56
BH	59.00	1.83	78.20	2.90
ncPSO	58.20	1.75	*76.40*	3.27
cPSO	*57.40*	1.96	78.50	3.14

The maximum coverage rate and the minimum standard deviation of all algorithms are set in italics in Table 8. When the number of sensors is 30, 40, 50, and 60, respectively, the coverage of ncPSO to the simulation environment can reach 55.07%, 65.46%, 72.65%, and 79.02%, respectively. Table 8 shows that although the standard deviation of ncPSO is not the smallest, the optimization effect of ncPSO is much better than that of other algorithms. In other words, the ncPSO has the best optimization effect on 3D coverage because of its excellent optimization ability. It can be seen from Table 8 that as the number of sensors increases, the coverage rate of simulation environment also increases. In order to show the operation process of the algorithm, the optimization processes are shown in Figure 7.

Figure 7 shows that algorithms constantly update the positions of sensors, so that the coverage rates constantly increase. Compared with other algorithms, ncPSO has a greater improvement in coverage rate with different number of sensors. In the whole process, ncPSO has a strong ability to jump out of the local optimum, so it improves the coverage rate greatly compared with other algorithms.

5.2. The Minimum Number of Sensor Nodes to Achieve a Certain Coverage Rate. How to use the minimum number of sensors to achieve a certain coverage rate will be discussed in this subsection. When the number of sensors is limited, it is necessary to consider how to arrange the sensor positions to minimize the number of sensors to achieve the specified coverage. The radius of sensors is set to 5 m, and the coverage rates are set to 70% and 80%, respectively. We also conduct 10 simulation experiments per group to ensure the accuracy of the data. The number of sensors required by different algorithms is tested by simulation experiments. Table 9 shows the results of simulation experiments.

The minimum number of sensors to achieve the specified coverage and the minimum standard deviation are set in italics in Table 9. It can be seen from Table 9 that in order to achieve 70% coverage, ncPSO needs 58.2 sensors on average and cPSO needs 57.4 sensors on average. Compared with other algorithms, ncPSO requires fewer sensors to achieve 70% coverage. Because of good optimization ability of ncPSO, ncPSO requires fewer sensors than other algorithms to achieve 80% coverage.

6. Conclusion

We propose the ncPSO in this paper. This algorithm reduces memory usage and also improves the problem that traditional compact algorithms tend to fall into local optima. This novel algorithm uses a Pareto distribution of the long-tailed distribution to describe the position of particle swarms. The ncPSO uses Pareto distribution and Gaussian perturbation to avoid feasible solutions falling into local optima. Then, we conduct the performance test of the proposed ncPSO on CEC2013 and compare it with the conventional heuristic and compact algorithms. The results show that the ncPSO has good optimization ability in most cases compared with other algorithms. Finally, we apply the ncPSO to the 3D sensor coverage problem and compare it with other algorithms. The simulation results show that the ncPSO can achieve better results in solving these problems.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare no conflict of interest.

Acknowledgments

This project is funded by the National Natural Science Foundation of China (No. 619320).

References

[1] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-international conference on neural networks*, pp. 1942–1948, Perth, WA, Australia, 1995.

[2] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*, MIT press, 1992.

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

[4] A. Hatamlou, "Black hole: a new heuristic optimization approach for data clustering," *Information Sciences*, vol. 222, pp. 175–184, 2013.

[5] D. Karaboga, "Artificial bee colony algorithm," *Scholarpedia*, vol. 5, no. 3, p. 6915, 2010.

[6] S. Mirjalili, "SCA: a sine cosine algorithm for solving optimization problems," *Knowledge-Based Systems*, vol. 96, pp. 120–133, 2016.

[7] X.-S. Yang and A. H. Gandomi, "Bat algorithm: a novel approach for global engineering optimization," *Engineering Computations*, vol. 29, no. 5, pp. 464–483, 2012.

[8] T. Joyce and J. M. Herrmann, "A review of no free lunch theorems, and their implications for metaheuristic optimisation," in *Nature-Inspired Algorithms and Applied Optimization*, pp. 27–51, Springer, Cham, 2018.

[9] J.-S. Pan, X.-X. Sun, S.-C. Chu, A. Abraham, and B. Yan, "Digital watermarking with improved SMS applied for qr code," *Engineering Applications of Artificial Intelligence*, vol. 97, article 104049, 2021.

[10] A. D. Boursianis, M. S. Papadopoulou, A. Gotsis et al., "Smart irrigation system for precision agriculture—The AREThOU5A IoT Platform," *IEEE Sensors Journal*, vol. 21, no. 16, pp. 17539–17547, 2021.

[11] C. Chen, J. Jiang, Y. Zhou, N. Lv, X. Liang, and S. Wan, "An edge intelligence empowered flooding process prediction using internet of things in smart city," *Journal of Parallel and Distributed Computing*, vol. 165, pp. 66–78, 2022.

[12] W. Wei, R. Yang, H. Gu, W. Zhao, C. Chen, and S. Wan, "Multi-objective optimization for resource allocation in vehicular cloud computing networks," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–10, 2021.

[13] J. Huang, H. Gao, S. Wan, and Y. Chen, "AoI-aware energy control and computation offloading for industrial IoT," *Future Generation Computer Systems*, vol. 139, pp. 29–37, 2023.

[14] X. Ji, Y. Zhang, D. Gong, X. Sun, and Y. Guo, "Multisurrogate-assisted multitasking particle swarm optimization for expensive multimodal problems," *Cybernetics*, pp. 1–15, 2021.

[15] X.-F. Song, Y. Zhang, Y.-N. Guo, X.-Y. Sun, and Y.-L. Wang, "Variable-size cooperative coevolutionary particle swarm optimization for feature selection on high-dimensional data," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 5, pp. 882–895, 2020.

[16] F. Wang, H. Zhang, and A. Zhou, "A particle swarm optimization algorithm for mixed-variable optimization problems," *Swarm and Evolutionary Computation*, vol. 60, article 100808, 2021.

[17] Y. Fan, P. Wang, A. A. Heidari, H. Chen, and M. Mafarja, "Random reselection particle swarm optimization for optimal design of solar photovoltaic modules," *Energy*, vol. 239, article 121865, 2022.

[18] T.-K. Dao, S.-C. Chu, T.-T. Nguyen, C.-S. Shieh, and M.-F. Horng, "Compact artificial bee colony," in *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, pp. 96–105, Springer, Cham, 2014.

[19] F. Neri, E. Mininno, and G. Iacca, "Compact particle swarm optimization," *Information Sciences*, vol. 239, pp. 96–121, 2013.

[20] J.-S. Pan, Q.-y. Yang, S.-C. Chu, and K.-C. Chang, "Compact sine cosine algorithm applied in vehicle routing problem with time window," *Telecommunication Systems*, vol. 78, no. 4, pp. 609–628, 2021.

[21] T.-K. Dao, J.-S. Pan, T.-T. Nguyen, S.-C. Chu, and C.-S. Shieh, "Compact bat algorithm," in *Intelligent Data Analysis and Its Applications*, pp. 57–68, Springer, Cham, 2014.

[22] W. Che-I, H.-Y. Kung, C.-H. Chen, and L.-C. Kuo, "An intelligent slope disaster prediction and monitoring system based on WSN and ANP," *Expert Systems with Applications*, vol. 41, no. 10, pp. 4554–4562, 2014.

[23] L. Muduli, D. P. Mishra, and P. K. Jana, "Application of wireless sensor network for environmental monitoring in underground coal mines: a systematic review," *Journal of Network and Computer Applications*, vol. 106, pp. 48–67, 2018.

[24] J. Wang, Y. Cao, B. Li, H. J. Kim, and S. Lee, "Particle swarm optimization based clustering algorithm with mobile sink for WSNs," *Future Generation Computer Systems*, vol. 76, pp. 452–457, 2017.

[25] A. Z. A. Aqeel-ur-Rehman, N. Islam, and Z. A. Shaikh, "A review of wireless sensors and networks applications in agriculture," *Computer Standards & Interfaces*, vol. 36, no. 2, pp. 263–270, 2014.

[26] B. Wang, "Coverage problems in sensor networks," *ACM Computing Surveys*, vol. 43, no. 4, pp. 1–53, 2011.

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

[28] A. Maheshwari and N. Chand, "A survey on wireless sensor networks coverage problems," in *Proceedings of 2nd International Conference on Communication, Computing and Networking*, pp. 153–164, Springer, Singapore, 2019.

[29] Y. Yoon and Y.-H. Kim, "An efficient genetic algorithm for maximum coverage deployment in wireless sensor networks," *IEEE Transactions on Cybernetics*, vol. 43, no. 5, pp. 1473–1483, 2013.

[30] A. Anurag, R. Priyadarshi, A. Goel, and B. Gupta, "2-d coverage optimization in wsn using a novel variant of particle swarm optimisation," in *2020 7th International Conference on Signal Processing and Integrated Networks*, pp. 663–668, Noida, India, 2020.

[31] N. T. Hanh, H. T. T. Binh, N. X. Hoai, and M. S. Palaniswami, "An efficient genetic algorithm for maximizing area coverage in wireless sensor networks," *Information Sciences*, vol. 488, pp. 58–75, 2019.

[32] H. T. T. Binh, N. T. Hanh, L. Van Quan, and N. Dey, "Improved cuckoo search and chaotic flower pollination optimization algorithm for maximizing area coverage in wireless sensor networks," *Neural Computing and Applications*, vol. 30, no. 7, pp. 2305–2317, 2018.

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

[34] J. Akram, H. Munawar, A. Kouzani, and M. Mahmud, "Using adaptive sensors for optimised target coverage in wireless sensor networks," *Sensors*, vol. 22, no. 3, p. 1083, 2022.

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

[36] J.-S. Pan, Q.-W. Chai, S.-C. Chu, and W. Ning, "3-d terrain node coverage of wireless sensor network using enhanced black hole algorithm," *Sensors*, vol. 20, no. 8, p. 2411, 2020.

[37] R. Elhabyan, W. Shi, and M. St-Hilaire, "A full area coverage guaranteed, energy efficient network configuration strategy for 3d wireless sensor networks," in *2018 IEEE Canadian Conference on Electrical & Computer Engineering*, pp. 1–6, Quebec, QC, Canada, 2018.

[38] W. Wang, H. Huang, F. He, F. Xiao, X. Jiang, and C. Sha, “An enhanced virtual force algorithm for diverse k-coverage deployment of 3d underwater wireless sensor networks,” *Sensors*, vol. 19, no. 16, p. 3496, 2019.

[39] B. Cao, J. Zhao, Z. Lv, X. Liu, X. Kang, and S. Yang, “Deployment optimization for 3d industrial wireless sensor networks based on particle swarm optimizers with distributed parallelism,” *Journal of Network and Computer Applications*, vol. 103, pp. 225–238, 2018.

[40] W. Bentz, T. Hoang, E. Bayasgalan, and D. Panagou, “Complete 3-d dynamic coverage in energy-constrained multi-uav sensor networks,” *Autonomous Robots*, vol. 42, no. 4, pp. 825–851, 2018.

[41] J. J. Liang, B. Y. Qu, P. N. Suganthan, and A. G. Hernández-Díaz, “Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization,” *Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, Technical Report*, vol. 201212, no. 34, pp. 281–295, 2013.