Telecommunication Systems
https://doi.org/10.1007/s11235-021-00804-y

A parallel compact sine cosine algorithm for TDOA localization of wireless sensor network

Siqi Zhang[1] · Fang Fan[2] · Wei Li[3] · Shu-Chuan Chu[1] · Jeng-Shyang Pan[4]

Accepted: 10 May 2021

Abstract
A Parallel and Compact version of the Sine Cosine Algorithm (PCSCA) is proposed in this article. Parallel method can effectively improve search ability and increase the diversity of solutions. We develop three communication strategies based on parallelism idea to serve different types of optimization function to achieve the best performance. Furthermore, compact method uses statistical distribution to represent the solutions, which can save memory space and energy of the digital device. To check the optimization effect of the proposed PCSCA algorithm, it is tested on the CEC2013 benchmark function set and compared to SCA, parallel compact Cuckoo Search (PCCS) algorithms. The empirical study demonstrates that PCSCA has improved by 50.1% and 5.6%, compared to SCA and PCCS, respectively. Finally, we apply PCSCA to optimize the position accuracy of sensor node deployed in 3D actual terrain. Experimental results show that PCSCA can achieve lower localization error via Time Difference of Arrival method.

Keywords Sine cosine algorithm · Parallel communication strategy · Compact strategy · 3D localization

1 Introduction

Optimization is the process of obtaining the optimal value from a given problem. Through optimization, an optimal solution can be found from a series of available solutions. Obviously, optimization problems exist in many fields. More and more optimization algorithms are applied to scientific research, engineering applications, economics and so on.

Along with the sustained development of optimization algorithm, many excellent optimization algorithms have been proposed such as Whale Optimization Algorithm (WOA) [1, 2], Cuckoo Search (CS) [3,4], Particle Swarm Optimization (PSO) ([5,6], Pigeon-inspired Optimization (PIO) [7,8], Artificial Bee Colony (ABC) [9,10], Flower Pollination Algorithm (FPA) [11,12], Bat Algorithm (BA) [13,14]), Ant Colony Optimization (ACO) [15–17], etc.

In 2016, Mirjalili proposed Sine Cosine Algorithm (SCA) [24] to solve the optimization problem. Initially, a series of random solutions are initialized. The objective function will evaluate the solution in each iteration, and these solutions will be iterated according to the update formula. As the number of iterations increases, there is a high probability of finding the global optimal solution. Sine Cosine Algorithm as one of the stochastic optimization algorithms has few requirements for the objective function, and the objective function can be regarded as a black box, which means that you only need to change the input and monitor the system output to maximize or minimize the output. The global optimal values of given problems can be found only by using the iterations of sine and cosine functions. Besides, the algorithm has a simple

✉ Jeng-Shyang Pan
jengshyangpan@gmail.com

Siqi Zhang
zhangsiqichn@gmail.com

Fang Fan
fangfan@sdust.edu.cn

Wei Li
wei.li@hrbeu.edu.cn

Shu-Chuan Chu
scchu@gmail.com

1 College of Computer Science and Engineering Shandong University of Science and Technology, Qingdao 266590, China

2 College of Intelligent Equipment, Shandong University of Science and Technology, Taian 271019, Shandong, China

3 College of Computer Science and Technology, Harbin Engineering University, Harbin 150001, China

4 Department of Information Management, Chaoyang University of Technology, Taichung, Taiwan

Published online: 02 June 2021

structure, few controlled parameters ,and is easy to realize. It has better optimization performance. However, in complex nonlinear optimization problems, Sine Cosine Algorithm still has the problem of poor global search ability and easy to fall into local optimum. To avoid the algorithm trapped in the local optimal, an improved SCA based on Levy flight [38] was proposed. The algorithm combines the current rank of the fitness value of solutions and its historical fitness value to mark the solutions which may fall into the local optimum. This improvement of Levy flight enhances the global searching ability in the exploration period of the algorithm, but it did not perform meticulous exploitation. The parallel Sine Cosine Algorithm uses parallel processing to increase the efficiency of optimization, but it should record the position and fitness value of the whole population, which takes many memories of devices.

Our approaches and contributions Based on the above discussions, it is a key to effectively deal with lower global search ability and memory usage. Unlike the existing SCA methods, we combine the parallel strategy with compact method to improve the Sine Cosine Algorithm. The main contributions of the paper are listed as follows:

- We develop a parallel and compact version Sine Cosine Algorithm for effective optimization and memory usage problems. PCSCA can not only increase the efficiency of optimization and the varieties of solutions, but also save the memory of devices effectively.
- We propose three parallel strategies by designing the arrangement structure of subgroups and utilizing compare and disturb operations for unimodal problems, multimodal problems, and complex nonlinear optimization problems.
- We extend the PCSCA algorithm on node location problem in 3D actual terrain.

We conduct extensive empirical studies on the CEC 2013 benchmark in Sect. 4. The empirical studies confirm that our new approach significantly outperforms the state-of-the-art approaches.

Organization The rest of the paper is organized as follows. Section 2 gives a brief retrospect to SCA algorithm and the basic principles of compact. Section 3 formally elaborate on the parallel improvement and compact improvement methods of SCA, and propose three parallel communication strategies. In Sect. 4, the results of numerical experiments are presented and discussed. Section 5 introduces the application of this algorithm to improve the accuracy of WSN node localization in 3D actual terrain. Finally, the Sect. 6 gives a conclusion.

2 Related works

In this section, we will introduce the basic idea of the SCA and the specific implementation process. Then,we discuss in detail some of the shortcomings of the SCA algorithm, and how to improve these shortcomings thro-ugh our improvements.

2.1 Sine cosine algorithm

The SCA has few requirements for the objective function, and the objective function can be regarded as a black box, which means that you only need to know the input and the input, without knowing the specific details of the objective function. The SCA is not the only novel in thought but also refined in structure. Initially, a series of random solutions are initialized. The objective function will evaluate the solution in each iteration, and these solutions will be iterated according to the update formula. As the number of iterations increases, there is a high probability of finding the global optimal solution.

The SCA algorithm includes two processes: exploration and exploitation. In the exploration process, the optimization algorithm will find a high probability area in the search space through a set of random solutions. During the exploitation process, the high probability area will be developed more carefully to find the optimal solution. At the beginning of optimization, a set of N random solutions of dimension d $X_i(i = 1, 2, \ldots, d)$ are generated. The fitness function is then used to calculate the fitness of each solution at each iteration. Find the solution with the best fitness. The current solution and the optimal solution are used to update the position of the solution. After a certain number of iterations, the final global optimal solution is found. The update formula of the solution is as follows:

$$X_i^{t+1} = \begin{cases} X_i^t + r_1 \times \sin(r_2) \times \left| r_3 P_i^t - X_i^t \right|, r_4 < 0.5 \\ X_i^t + r_1 \times \cos(r_2) \times \left| r_3 P_i^t - X_i^t \right|, r_4 \geq 0.5 \end{cases} \quad (1)$$

where t is the number of current iteration. X_i^t is the value of solution at the tth iteration and in the ith dimension r_1, r_2, and r_3 are three random number. P_i^t is the current best solution. $| \, |$ is the absolute symbol. The next generation position of X_i^t is X_i^{t+1}. The r_1 decreases linearly with the number of iterations from a to 0. The calculation formula are described as follows:

$$r_1 = a - a\frac{t}{T} \quad (2)$$

where a is a constant value. T is the number of maximum iteration. t represents the current number of iteration. r_1 can guide SCA to explore or exploit. When $r_1 > 1$, X_i^{t+1} will move away from the current best solution P_i^t. When $r_1 \leq 1$,

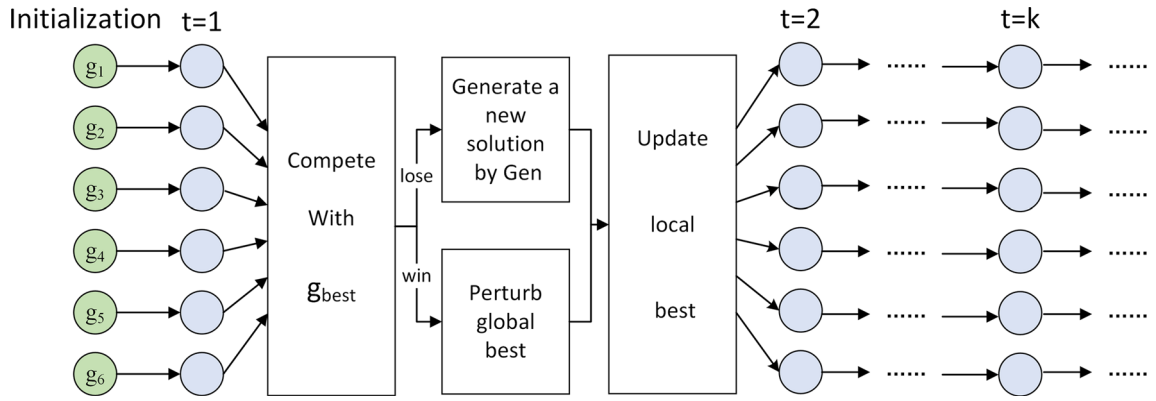

Fig. 1 Communication Strategy 1

X_i^{t+1} will move closer to the current best solution P_i^t. The parameter $r_3 \in [0, 2]$ is the weight of P_i^t. The parameter $r_4 \in [0, 1]$ is used to select the location update function. When $r_4 \geq 0.5$, the algorithm will choose to update the cosine funtion, as in (1). Otherwise, the sine function will be selected.

3 The proposed method

In this section, we propose a parallel communication approach to avoid local optimum issue existing in SCA in Sect. 3.2 and a compact technique to reduce memory consumption in SCA in Sect. 3.3, based on which we then develop a new optimization approach, denoted PCSCA, in Algorithm 1.

3.1 Background

This article introduces the concept of parallel processing [39] into SCA and designs three communication strategies for this purpose. There is an issue with the raw SCA algorithm. It is likely to be caught in the local optimal solution prematurely and population diversity declines prematurely. In order to mitigate the impact of this shortcoming, parallel communication strategies can be adopted to avoid the situation where the entire population falls into the local optimum and cannot escape. The parallel method is to divide the entire population into designed subpopulations, and each subpopulation independently searches in its own search space [40]. Combining it with SCA method, the specific scheme is as follows. Suppose there are 30 individuals in the entire population, and each individual is numbered 1–30. If the population is divided into 6 sub-groups, we take 1–5 as the first group, 6–10 as the second group, and so on. The grouping is completed. Then each sub-population evolves independently according to the iterative rules. The evaluation of the solution is based on the fitness value. The maximum or minimum value is the local optimal in the subgroup. After triggering a certain inter-population communication scheme, replace inappropriate individuals in the population with corresponding strategies, and exploit or explore the promising area. When the number of iterations reaches the maximum, the search ends.

3.2 Three proposed parallel communication strategies

Because of the existence of Unimodal problems, multimodal problems, and complex non-linear optimization problems, different strategies can be used for different types of problems.

Based on this idea, this section designs three communication strategies adapted to different function types.

- Strategy 1 solves the unimodal problem, focusing on the communication between the group and the global optimal, reducing the optimal exchange between groups.
- Strategy 2 to solve the multimodal problem with multiple peaks and valleys, focusing on the communication between groups, can very well help the group trapped in the local optimal to jump out of the local optimal.
- Strategy 3 solves complex non-linear optimization problems.

The workflow of Strategy 1 is shown in Fig. 1, which first computes the local optimal (denoted as l_{best}) and the global optimal (denoted as g_{best}) within the group, and then compares them. Second, if l_{best} is worse than g_{best}, we use l_{best} and g_{best} to generate the genome $Gen(l_{b1}, g_{b1}, l_{b2}, g_{b2}, \ldots, l_{gn}, g_{gn})$. Then a new solution is randomly generated from Gen. In contrast, if l_{best} is better than g_{best}, then we perform an disturbing operation on g_{best} to get a new solution. Third, the new solution obtained from the second step will be compared with l_{best}, and the winner will replace l_{best}. At the end of each generation, Strategy 1 will be executed once.

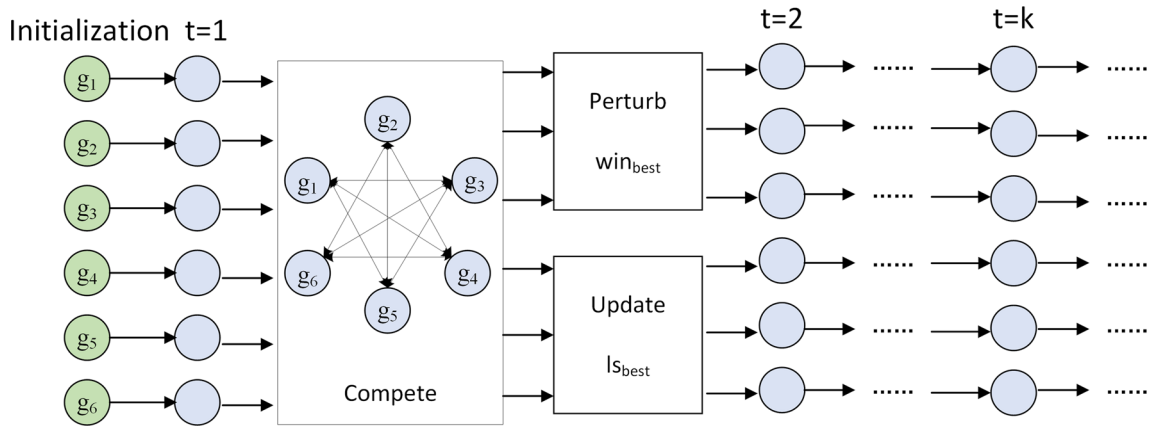

Fig. 2 Communication Strategy 2

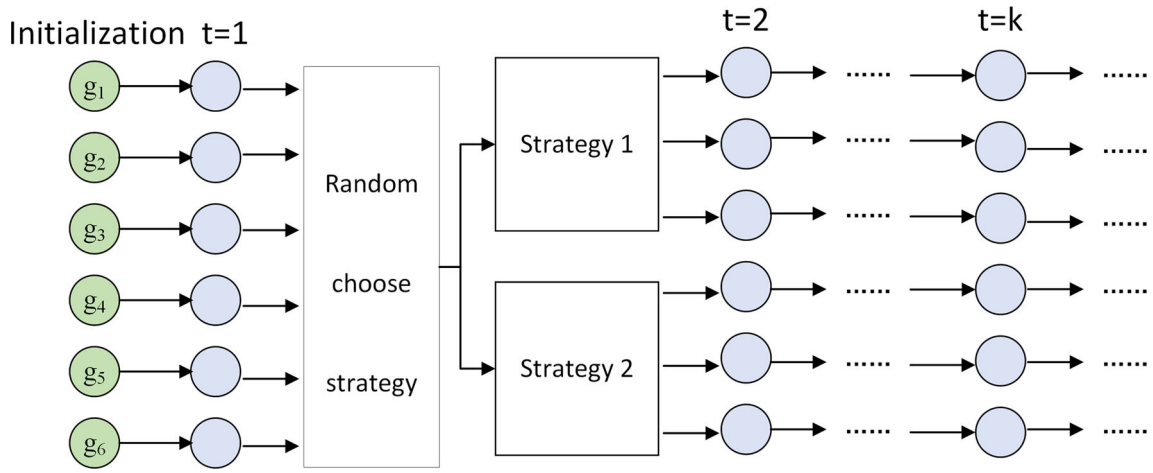

Fig. 3 Communication Strategy 3

Strategy 2 (as shown in Fig. 2) encloses the subgroups in the a circle. The l_{best} of the subgroup in the circle will be compared with the l_{best} of the non-adjacent group. For example, g_1 is compared with the g_3, g_4, and g_5. According to the fitness value returned by the finesse function, distinguish the better value group and worse value group. Here, we use win_{best} to represent the better group and ls_{best} represents the bad group. Next, replacing ls_{best} with win_{best}. Finally, we perform a disturb operation on win_{best} to generate a new solution. Then compare the new solution with win_{best}. If the fitness value of new solution is better than win_{best}, then replace win_{best}. At the end of each generation, Strategy 2 will be executed once.

In Strategy 3, random number r will be generated at the beginning of the communication. When $r < 0.5$ Strategy 2 will be executed. If $r \geq 0.5$, Strategy 1 will be executed. Strategy 3 as shown in Fig. 3. Strategy 3, as a hybrid strategy, has greater randomness and probability of finding the global optimal solution when dealing with complex problems.

3.3 Our PCSCA approach

Following the proposed three parallel communication strategies, in this section, we present our PCSCA approach. For convenience, this article uses Strategy 3 to demonstrate the PCSCA algorithm. The pseudocode is shown in Algorithm 1.

The compact method [25] is a population-less method, which is essentially based on the probability model of the estimated distribution algorithm (EDA). In the process of algorithm execution, a distribution model can be used instead of the population to perform related operations. The continuous search of the decision space is realized by updating the distribution model.

We initially receive the input parameter. The virtual population size is the number of virtual solutions. Max iteration is the maximum number of iterations and the loop termination condition. Fitness function is the problem, which we need to optimize and approach the optimal value of it. Dimension d is the solutions dimension of a fitness function. Upper and lower bounds are the bounds of the dimension of decision

space. Then initializing each group, there are 6 groups in this article. Each group contains $G(i).PV$, $G(i).best$ and $G(i).Fmin$. The $G(i).PV$ is a distribution model to represent the characteristics of the population. Perturbation Vector (PV) is a data structure used to describe the macroscopic probability distribution of a population. PV is composed of a $n \times 2$ matrix: $PV_t = [\mu_t, \delta_t]$, where μ and δ are the mean and standard deviation of PV. t is the number of current iterations. Each pair of average and standard deviation in PV constitutes a Probability Density Function (PDF). When initializing the virtual population, we usually set $\mu_t[i] = 0, \delta_t[i] = \lambda$, where λ is a constant with a value of 10. $G(i).best$ and $G(i).Fmin$ is respectively the position and fitness value of the best optimal solution found during the iteration in group i. $GlobalBest$ and $GlobalFmin$ is position and fitness value of global best solution.

Then, we iteratively update each groups. For each i in range $[1, groups]$, we first use CDF inverse function equation to generate a new solution X_1. PDF is truncated at $[-1, 1]$ and PDF normalizes the area of the amplitude to 1. By constructing Chebyshev polynomial, a Cumulative Distribution Function (CDF) with a range of [0, 1] can be obtained from PDF.The CDF can be written as

$$\begin{aligned} CDF &= \int_{-1}^{x} PDFdx \\ &= \int_{-1}^{x} \frac{\sqrt{\frac{2}{\pi}} e^{-\frac{(x-\mu)^2}{2\delta^2}}}{\delta\left(\text{erf}\left(\frac{\mu+1}{\sqrt{2}\delta}\right) - \text{erf}\left(\frac{\mu-1}{\sqrt{2}\delta}\right)\right)} dx \end{aligned} \tag{3}$$

where x is a random number and the range of x is in range $[-1, 1]$. μ and δ are the mean value and standard deviation of PV. erf is the error function. Using the inverse function of
CDF can randomly generate an actual solution of PV. CDF inverse function as follows:

$$\begin{aligned} y = \sqrt{2}\delta erf^{-1}\Bigg(&-\text{erf}\left(\frac{\mu+1}{\sqrt{2}\delta}\right) - xerf\left(\frac{\mu-1}{\sqrt{2}\delta}\right) \\ &+ xerf\left(\frac{\mu+1}{\sqrt{2}\delta}\right)\Bigg) + \mu \end{aligned} \tag{4}$$

The value range of y is in range $[-1, 1]$. erf^{-1} is the inverse function of erf. The parameter $x \in [0, 1]$, as a random number. The solution y needs to be mapped to the actual decision space. The mapping formula is:

$$y_{ds} = y \times \frac{1}{2}(ub - lb) + \frac{1}{2}(ub + lb) \tag{5}$$

ub is upper bounds and lb is lower bounds. y is the solution generated by the inverse CDF function. y_{ds} is a solution in the actual solution space. Record the obtained y_{ds} as the solution X_1, then we perform the SCA update operation Eq. (1) on X_1 to obtain the solution X_2. Comparing the fitness values of X_1 with X_2, which are obtained from the fitness function. The better one and bad one are denoted as $winner$ and $loser$, respectively. For each i in range $[1, d]$, we first update PV according to Eqs. (6) and (7) (Line 9). The update formula of PV for the mean is:

$$\mu_i^{t+1} = \mu_i^t + \frac{1}{N_p}(winner_i - loser_i) \tag{6}$$

μ_i^{t+1} is the updated mean value. μ_i^t is the currnet mean value. N_p is the number of solutions in the virtual population. The update formula for standard deviation is:

$$\delta_i^{t+1} = \sqrt{(\delta_i^t)^2 + (\mu_i^t)^2 - (\mu_i^{t+1})^2 + \frac{1}{N_p}(winner_i - loser_i)^2} \tag{7}$$

After that, if fit_{winner} is less than the given threshold, the $best$ and $Fmin$ values will be updated to winner and fit_{winner}, respectively. When the updating population iteration end, $GlobalBest$ and $GlobalFmin$ are updated (Line 15).

Finally, we use Strategy 3 to help the algorithm achieve better solutions. For each i in range $[1, groups]$, there will generate a randum number $rand$ between 0 and 1. When $rand$ is less than 0.5, then a group k will be selected. Comparing $G(i).Fmin$ with $G(k).Fmin$, the winner group is denoted as w and the loser group is denoted as l. The $G(l).Fmin$ will be replaced by $G(w).Fmin$. The $G(w).best$ then operate disturb operation to replace the current position. If $rand$ is greater than 0.5, we use $G(i).best$ and $GlobalBest$ to generate the genome Gen which is utilized to get a new solution x. Then, $G(i).best$ is updated by x.

4 Experiments

We conduct extensive empirical studies to evaluate the efficiency and effectiveness of our proposed algorithms for unimodal problems, multimodal problems and composite problems.

4.1 Setup

In specific, we evaluate the following algorithms:

- SCA: The brief review of SCA is discussed in Sect. 2.1.
- PCCS: the algorithm in [29], which firstly implemented the compact CS with communication strategy.

Algorithm 1 PCSCA(Use Strategy 3)

Require: Virtual population size, Max iteration, Fitness function, Dimension d, Upper and Lower bounds
Ensure: Global best value $GlobalBest$

```
Set the number of groups g, each group is G(i), (i ≤ g)
Initialize the G(i).PV, G(i).best and G(i).Fmin of each group
GlobalBest = G(1).best; GlobalFmin = G(1).Fmin
while (t < Max Iteration) do
    for i = 1 : groups do
        Get X1,X2 via Eq. (4) Eq. (5) Eq. (1).
        [winner, loser, fit_winner] = compete(X1,X2)
        for j = 1 : d do
            Update G(j).PV via Eq. (6) Eq. (7)
        end for
        if fit_winner < G(i).Fmin then
            G(i).best = winenr
            G(i).Fmin = fit_winner
        end if
        Update GlobalBest and GlobalFmin
    end for
    for i = 1 : groups do
        if rand < 0.5 then
            Select a group k ≠ i
            Compete G(i).Fmin and G(k).Fmin to
              get winner group (w) loser group (l)
            Set G(l).Fmin = G(w).Fmin
            Disturb and update G(w).best
        else
            Generate Gen(l_b1, g_b1, l_b2, g_b2, ..., l_bn, g_bn)
            Use Gen generate a new solution x
            Update G(i).best by x
        end if
    end for
end while
```

- PCSCAS1: the approach discussed in Sect. 3 with parallel communication Strategy 1.
- PCSCAS2: the approach discussed in Sect. 3 with parallel communication Strategy 2.
- PCSCAS3: the approach discussed in Sect. 3 with parallel communication Strategy 3.

All algorithms are implemented in Matlab; the source code of SCA and PCCS are obtained from the authors in and while all other algorithms are implemented by us. All experiments are conducted on a machine with an Intel(R) Core(TM) i3-8100 3.60GHz CPU and 8GB memory running b64 bits Windows 10. We evaluate the performance of all algorithms on the CEC2013 [28] benchmark functions as follows: CEC2013 has 28 test functions in total. Among them, F1–F5 are unimodal functions. F6–F20 is a multimodal function, used to test the algorithm's ability to avoid falling into local optimal solutions. In addition, F21–F28 are complex functions.

4.2 Effectiveness evaluation

We compare our proposed algorithms, PCSCA, with the existing algorithms, SCA, and PCCS. In Fig. 4, the convergence curve of each algorithm is gradually decreasing and tending to be flat. The faster the convergence curve drops, the stronger the search capability of the algorithm, the faster the convergence speed, and the fewer iterations used. This advantage can greatly improve the search efficiency of the algorithm and expand the scope of exploration in the solution space. The optimal value of the test function in CEC2013 is as small as possible. In order to achieve fair competition, there are 1000 iterations in total and take the average value of 30 tests.

According to Fig. 4, the experimental results show that Strategy 1 is the best for unimodal function optimization. When the line of fitness value gradually becomes flat, it means that the algorithm has converged to the global optimal solution. It can be seen that in functions F1, F2 and F5, PCSCAS1 has a very fast convergence rate compared to SCA and PCCS and converge in about 100, 200 and 175 generations respectively, This means that PCSCAS1 can find the optimal solution in a shorter iteration on the unimodal function. It can be seen from Table 1 that the average value of PCSCAS1 is lower than that of SCA and PCCS on F1, F3, and F5. This shows that the stability of this strategy is also excellent. Due to PCSCA with Strategy 2 has more frequent comparisons and exchanges, it will also perform well when dealing with simple unimodal functions. But the time cost is higher than Strategy1, so it is not recommended to use Strategy2 to solve the unimodal problem.

PCSCAS2 is excellent when solving multimodal functions. The convergence curve of PCSCAS2, as shown in Fig. 4, achieved the lowest fitness value. Compared with other algorithms, the convergence speed has no obvious difference. PCSCAS2 has converged around 200 generations on average. In the Competing stage of Strategy2, each group will be frequently compared with non-adjacent groups to increase sufficient communication between groups, making it easier for groups that fall into the local optimum to jump out of their current position. After that, performing perturb and update operations can increase the diversity of solutions. Strategy 2 can greatly improve the performance of PCSCA to solve multimodal problems. In Table 1, We can see that from F6-F20, for all multimodal functions, PCSCAS2 all achieved the minimum value. Compared with other algorithms, it shows that PCSCA2 has good stability in solving multimodal function distance.

Strategy 3 is designed to optimize complex nonlinear problems. It can be seen from the convergence curves of PCSCA3 in functions F23, F27, and F28. PCSCAS3 has a fast convergence rate before 200 generations and will continue to explore and exploit the solution space in subsequent

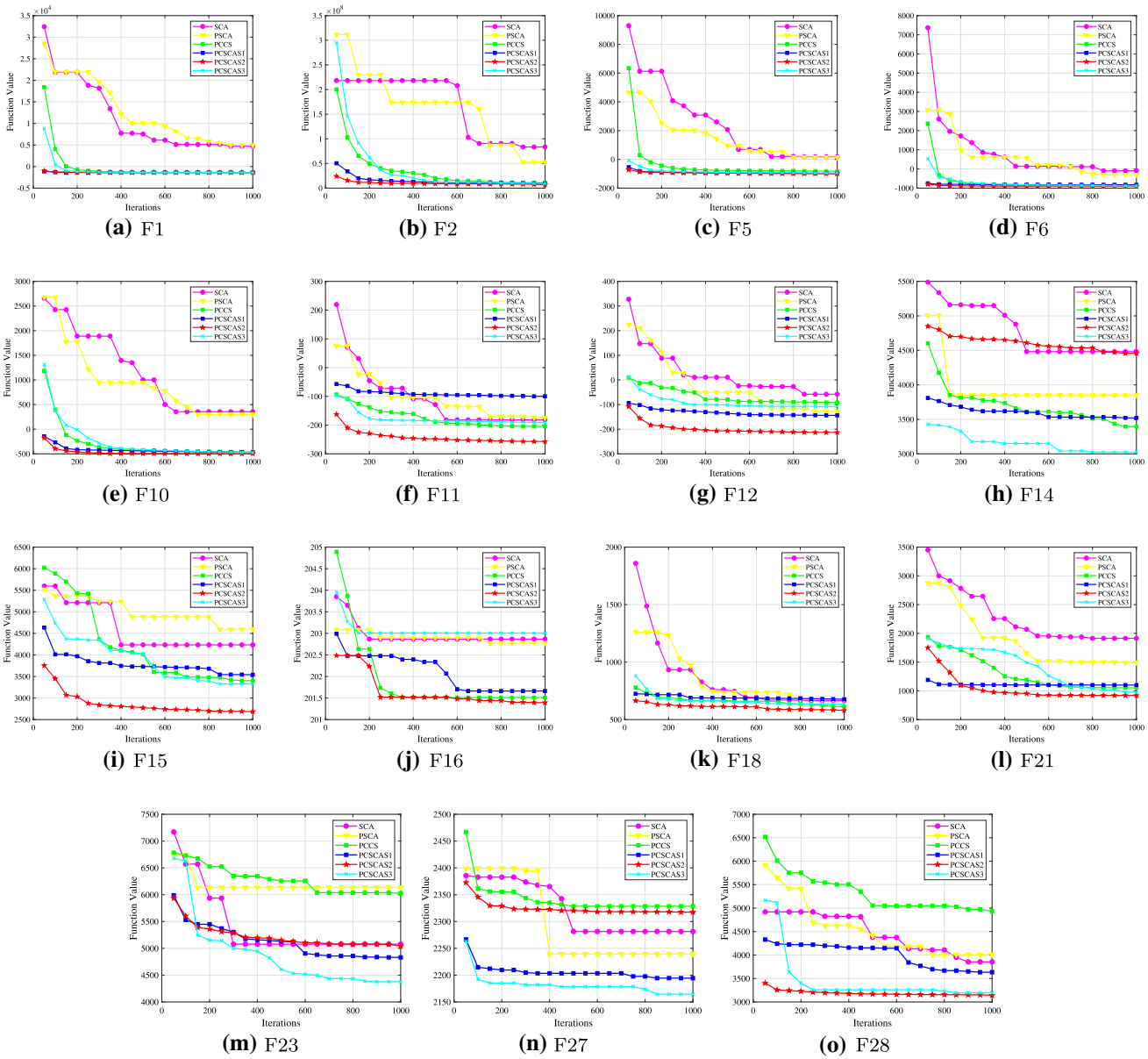

Fig. 4 Simulation results of CEC 2013

iterations to find the optimal solution. Since Strategy 3 will randomly select Strategy 1 or Strategy 2 during parallel communication, this can greatly improve the diversity and randomness of the solution. In Table 1, we can see that F23, F26, and F27 all achieved the best results. The above test results show that PCSCAS3 is extremely stable and can effectively solve complex nonlinear problems.

5 Results

The PCSCA algorithm is applied to WSN node localization in 3D actual terrain. The 3D actual terrain is Xiaozhu mountain next to Shandong University of Science and Technology. It has a complex topography, with peaks, ridges ,and ravines, and is very suitable for localization as a test node for actual terrain. The 3D terrain used in this article is formed by Matlab, as shown in Fig. 5. When a node in the WSN exchanges information with other nodes in the 3D actual terrain, the signal may be blocked by the mountain. If the node is not connected with other nodes, the node cannot send out the information collected by itself. So it is very important to check whether the nodes are connected. For obstacle judgment, you can use the line-of-sight (LOS) [30] algorithm to judge. Because Bresenham LOS algorithm [31] has a faster calculation speed, and it does not need to perform interpo-

Table 1 Performance comparison of SCA, PCCS, PCSCA (including Strategy 1, Strategy 2, and Strategy 3) for functions (F1–F28)

Functions	SCA	PCCS	PCSCAS1	PCSCAS2	PCSCAS3
F1	4.30E+03	− 1.39E+03	**−1.40E+03**	**−1.40E+03**	**−1.40E+03**
F2	6.40E+07	1.29E+07	1.40E+07	**8.04E+06**	2.43E+07
F3	4.86E+10	2.28E+09	1.35E+09	**3.71E+08**	4.23E+09
F4	3.09E+04	4.46E+04	3.30E+04	**1.72E+04**	4.16E+04
F5	2.20E+02	−9.16E+02	−9.73E+02	**−9.94E+02**	−9.25E+02
F6	−2.46E+02	−8.61E+02	−8.56E+02	**−8.63E+02**	−2.46E+02
F7	−6.39E+02	−6.96E+02	−6.70E+02	**−7.22E+02**	−6.39E+02
F8	**−6.79E+02**	**−6.79E+02**	**−6.79E+02**	**−6.79E+02**	**−6.79E+02**
F9	− 5.75E+02	**−5.80E+02**	**−5.80E+02**	**−5.80E+02**	− 5.75E+02
F10	3.11E+02	− 4.70E+02	− 4.84E+02	**−4.94E+02**	3.11E+02
F11	− 1.89E+02	− 1.86E+02	− 1.74E+02	**−2.19E+02**	− 1.89E+02
F12	− 6.48E+01	**−1.27E+02**	− 1.11E+02	**−1.27E+02**	− 6.48E+01
F13	3.12E+01	− 1.65E+01	− 1.90E+01	**-3.42E+01**	3.12E+01
F14	4.26E+03	2.94E+03	3.01E+03	**2.77E+03**	4.26E+03
F15	4.53E+03	3.24E+03	3.03E+03	**2.74E+03**	4.53E+03
F16	2.02E+02	2.02E+02	**2.01E+02**	**2.01E+02**	2.02E+02
F17	5.81E+02	5.06E+02	5.15E+02	**4.74E+02**	5.81E+02
F18	7.01E+02	6.20E+02	5.73E+02	**5.60E+02**	7.01E+02
F19	1.79E+03	5.15E+02	5.14E+02	**5.11E+02**	1.79E+03
F20	6.10E+02	6.10E+02	6.10E+02	**6.09E+02**	6.10E+02
F21	1.69E+03	1.09E+03	1.04E+03	**1.02E+03**	**1.02E+03**
F22	5.49E+03	4.74E+03	4.77E+03	**4.44E+03**	4.57E+03
F23	5.80E+03	5.93E+03	4.80E+03	4.74E+03	**4.47E+03**
F24	1.28E+03	**1.26E+03**	1.27E+03	1.27E+03	1.27E+03
F25	1.38E+03	**1.36E+03**	1.37E+03	1.38E+03	1.37E+03
F26	1.41E+03	1.60E+03	1.40E+03	**1.40E+03**	**1.40E+03**
F27	2.29E+03	2.20E+03	2.19E+03	2.19E+03	**2.18E+03**
F28	4.08E+03	5.45E+03	3.94E+03	**3.60E+03**	4.03E+03

We evaluate the performance of all algorithms on the CEC2013 benchmark functions. The optimal value of benchmark functions is as small as possible. Therefore, in order to highlight the optimal value obtained by the same test function on different algorithms, the optimal value is bolded

lation calculation, and there is no strict requirement on the number of calculation points. Therefore, this article uses the Bresenham LOS algorithm to determine whether a point in the terrain is blocked by obstacles.

Suppose there are two nodes A and B and we are going to send a message from point A to point B. Make a straight-line L connecting AB. If the height of a point C between AB is higher than the height of the straight-line L at the same coordinate, then point C is considered to be an obstacle between points AB. Conversely, if the height of C is lower than the height of the straight-line L at the same coordinate, point C is not considered as an obstacle between points AB, which is illustrated in Fig. 6.

Because the communication range of the sensor node in 3D space is a sphere, if the Euclidean distance D of the two nodes is greater than the detection radius, it means that the two nodes do not cover each other. This paper uses a con-

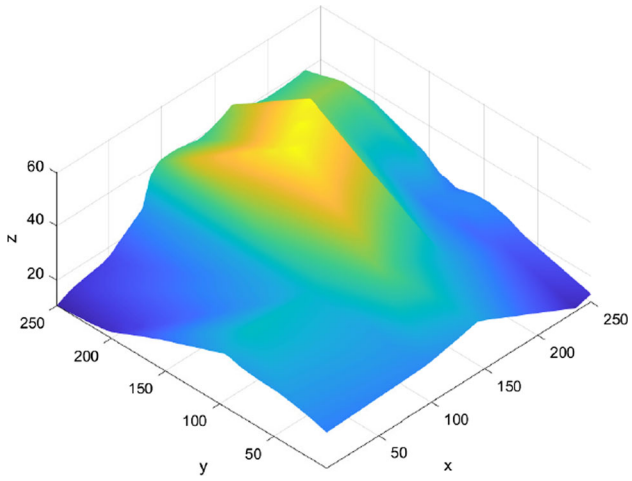

Fig. 5 The 3D terrain which sensor nodes deployed

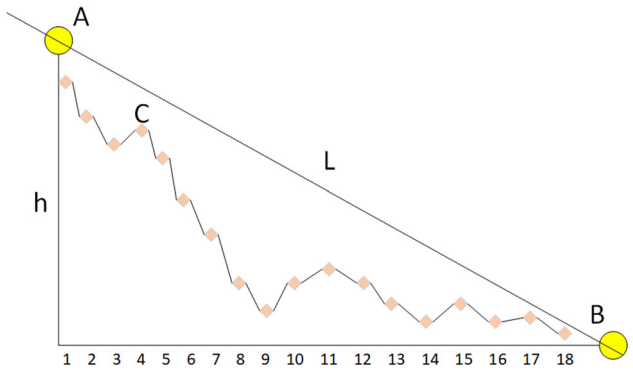

Fig. 6 Determination of LOS between A and B

nected matrix to represent the connectivity between nodes, which is expressed as follows:

$$C(s,r) = \begin{cases} 1, & D(s,r) \leq R \text{ and there no obstacles} \\ 0, & D(s,r) > R \text{ and there has obstacles} \end{cases} \tag{8}$$

where C is the connectivity matrix, s and r are nodes. When the Euclidean distance D between s and r is less than or equal to detection radius of node and there is no obstacle, it means that the two nodes are connected marked as 1. Otherwise, it is recorded as 0.

Intelligence computing has the advantage of solving optimization problems effectively and some work has applied it to locate the unknown nodes of WSN on 2D surface. In this article, the PCSCA is applied to reduce the locate error based TDOA and the error can be calculated as following:

$$error = \sum_{i=2}^{n} (r_i - d_i)^2 \tag{9}$$

$$Accuracy = \frac{error}{radius} \tag{10}$$

where r_i and d_i are calculated by Eq. (4). r_i is the distance difference of the estimated node by the algorithm between the anchor node i and anchor node 1. d_i are the distance difference of the unknown node between the anchor node i and anchor node 1. There are n anchor nodes. Because anchor node 1 is used to calculate as the bench, so i start from 2. $error$ is the total distance error. Radius is the detection radius of the sensor node. In order to further reduce the localization error, this paper proposes a fitness function calculation formula as follows:

$$f(x,y) = min\left(\left(\frac{1}{T_n}\right)\sum_{i=2}^{n}(r_i - d_i)^2\right) \tag{11}$$

where T_n is the number of anchor nodes connected with unknown nodes. PCSCA can use the fitness value returned

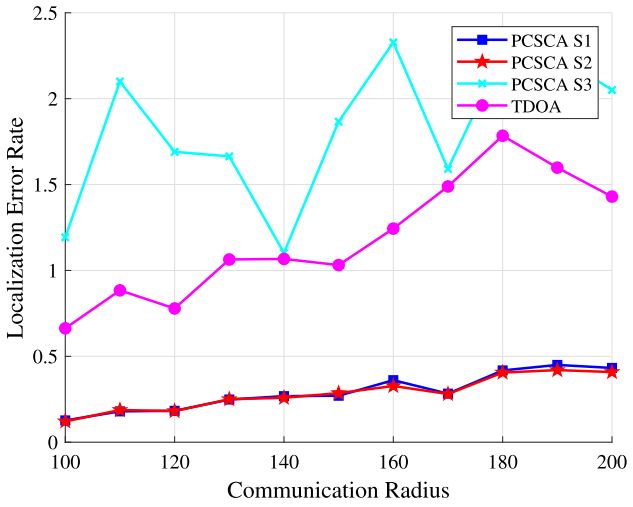

Fig. 7 The influence of communication radius on accuracy

by this fitness function to find the evaluation position with the smallest error. The main work of this application can be divided into the following steps: (1) Randomly deploy nodes in the 3D terrain as shown in Fig. 5. (2) Determine the connectivity between the unknown node and the anchor node through the $C(s,r)$. (3) Calculate the distance difference between different anchor nodes. (4) The algorithm is used to finally determine the location of the unknown node according to the distance difference.

The nodes will be randomly arranged in the 3D terrain. The terrain size is 60 × 250 × 250 meters. For comprehensively testing the performance of PCSCA in TDOA, two tests are provided in this section. The nodes in both tests are generated independently. The first test is to test the change of localization error as the detection radius increases when the number of anchor nodes is fixed. Each group has got different localization errors under different detection radius.

As shown in Fig. 7, The abscissa in the Fig. is the detection radius of the node. The detection radius is an important parameter for communication between nodes. If the detection radius is too large, the more power is needed to send the signal to anchor nodes. For the remote anchor nodes that have established a connection, the time error of signal transmission will affect the final localization accuracy, so the detection radius is not as large as possible. If the detection radius is too small, then this node will not be able to connect with any anchor nodes, and it can not send the data which it collected to the server. The ordinate is Localtion error rate calculated by the formula Eq. (10). It is used to measure the error between the estimated position of the algorithm and the actual position in the simulation localization process. It can be seen from the figure that as the detection radius increases from 100 meters to 200 meters, the strategy 1 and strategy 2 curves basically coincide. When the detection radius is doubled, the highest point of localization error does not exceed 0.5, indicating the

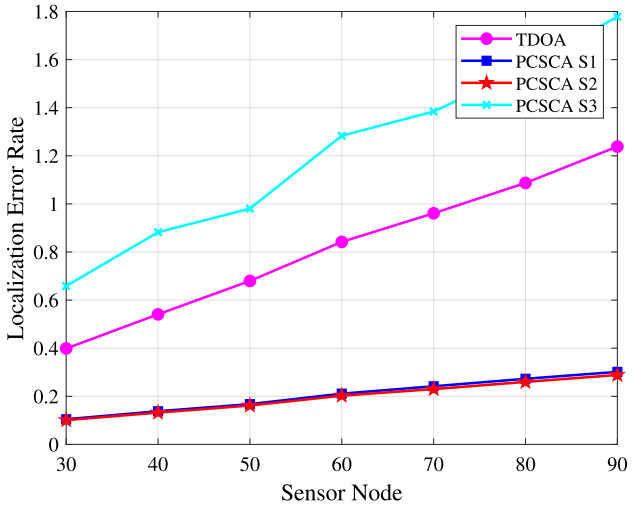

Fig. 8 The influence of sensor node number on accuracy

node position evaluated by the algorithm , Within half of the detection radius of the node. The curve of strategy 3 shows great volatility, which shows that its stability in the localization process is extremely poor, and it is not suitable for solving simple fitness functions. The TDOA method is relatively stable before the detection radius is 150 meters, and continues to rise after 160 meters. It reaches the highest point when the detection radius is 180, but then starts to decline, indicating that TDOA has a better effect in small radius, and it has a better effect when the radius exceeds 150 meters. Errors will become unacceptable.

Usually, when arranging nodes, only one node is not arranged. Therefore, when multiple nodes are arranged, the average localization error of the nodes fluctuates to the stability of the localization optimization algorithm. A stable localization optimization algorithm will not show the jitter of localization error , which is exactly what we want. Therefore, in this paper, the simulation results of deploying different numbers of nodes with the
same detection radius are tested. The number of anchor nodes is 25, evenly distributed in the actual terrain.

As shown in Fig. 8, the abscissa is the number of sensor nodes, and the ordinate is the Location error rate. It can be seen from the figure that as the number of nodes increases from 30 to 90, which is a three-fold increase, the average localization errors of strategy 1 and strategy 2 both only slightly increase. This is related to the stable update of the optimal in the group during the execution of strategy 1 and strategy 2. It can be seen from the curves of S1 and S2 that the abscissa is 30–90 and the ordinate only increases the average localization error of about 0.2, indicating that as the number of nodes increases by 3 times, the stability of S1 and S2 is extremely excellent. Strategy S3 is to mix S1 with strategy S2. Although it has higher randomness, it also brings a certain degree of instability.

6 Conclusion

In this paper, we developed a parallel and compact-based SCA algorithm, PCSCA, which can reduce the consumption of memory and energy and improve the global search ability. Extensive empirical studies on 28 CEC2013 benchmark functions demonstrate the effectiveness of our approach and the efficiency of our techniques. Compared to the existing SCA and PCCS, our PCSCA has faster convergence speed and better global optimal value. Furthermore, applying PCSCA algorithm to the WSN localization problem in 3D actual terrain can get a more accurate position than that of TDOA method and reduce the calculate energy consumption and memory usage, which further testify the effectiveness and superiority of the proposed PCSCA algorithm. In the next step, we will make more improvements on the basis of SCA by combining some promising techniques [32–37].

Declarations

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

References

1. Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software*, *95*, 51–67.
2. Sayed, G. I., Darwish, A., Hassanien, A. E., & Pan, J. S. (2016). Breast cancer diagnosis approach based on meta-heuristic optimization algorithm inspired by the bubble-net hunting strategy of whales. In *International conference on genetic and evolutionary computing* (pp. 306–313). Cham: Springer.
3. Gandomi, A. H., Yang, X. S., & Alavi, A. H. (2013). Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems. *Engineering with Computers*, *29*(1), 17–35.
4. Pan, J. S., Song, P. C., Chu, S. C., & Peng, Y. J. (2020). Improved compact cuckoo search algorithm applied to location of drone logistics hub. *Mathematics*, *8*(3), 333.
5. Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks* (Vol. 4, pp. 1942-1948). IEEE.
6. Wang, H., Sun, H., Li, C., Rahnamayan, S., & Pan, J. S. (2013). Diversity enhanced particle swarm optimization with neighborhood search. *Information Sciences*, *223*, 119–135.
7. Duan, H., & Qiao, P. (2014). Pigeon-inspired optimization: a new swarm intelligence optimizer for air robot path planning. International Journal of Intelligent Computing and Cybernetics.
8. Tian, A. Q., Chu, S. C., Pan, J. S., Cui, H., & Zheng, W. M. (2020). A compact pigeon-inspired optimization for maximum short-term generation mode in cascade hydroelectric power station. *Sustainability*, *12*(3), 767.
9. Karaboga, D. (2005). *An idea based on honey bee swarm for numerical optimization* (Vol. 200, pp. 1–10). Technical report-tr06, Erciyes University, Engineering Faculty, Computer Engineering Department.
10. TSai, P. W., Pan, J. S., Liao, B. Y., & Chu, S. C. (2009). Enhanced artificial bee colony optimization. *International Journal of Innovative Computing, Information and Control, 5*(12), 5081–5092.

11. Yang, X. S. (2012). Flower pollination algorithm for global optimization. In *International conference on unconventional computing and natural computation* (pp. 240-249). Berlin: Springer.
12. Zhuang, J., Luo, H., Pan, T. S., & Pan, J. S. Improved flower pollination algorithm for the capacitated vehicle routing problem.
13. YYang, X. S., & Gandomi, A. H. (2012). Bat algorithm: A novel approach for global engineering optimization. Engineering Computations.
14. Dao, T. K., Pan, J. S., Chu, S. C., & Shieh, C. S. (2014). Compact bat algorithm. In *Intelligent data analysis and its applications* (Vol. II, pp. 57-68). Cham: Springer.
15. Parpinelli, R. S., Lopes, H. S., & Freitas, A. A. (2002). Data mining with an ant colony optimization algorithm. *IEEE Transactions on Evolutionary Computation*, *6*(4), 321–332.
16. Chu, S. C., Roddick, J. F., & Pan, J. S. (2004). Ant colony system with communication strategies. *Information Sciences*, *167*(1–4), 63–76.
17. Chu, S. C., Roddick, J. F., Su, C. J., & Pan, J. S. (2004). Constrained ant colony optimization for data clustering. In *Pacific Rim international conference on artificial intelligence* (pp. 534–543). Berlin: Springer.
18. Mininno, E., Neri, F., Cupertino, F., & Naso, D. (2010). Compact differential evolution. *IEEE Transactions on Evolutionary Computation*, *15*(1), 32–54.
19. Neri, F., Mininno, E., & Iacca, G. (2013). Compact particle swarm optimization. *Information Sciences*, *239*, 96–121.
20. Hofmann-Wellenhof, B., Lichtenegger, H., & Collins, J. (2012). *Global positioning system: Theory and practice*. Berlin: Springer.
21. Wiley, W. C., & McLaren, I. H. (1955). Time-of-flight mass spectrometer with improved resolution. *Review of Scientific Instruments*, *26*(12), 1150–1157.
22. Liu, N., & Pan, J. S. (2019). A bi-population QUasi-Affine TRansformation evolution algorithm for global optimization and its application to dynamic deployment in wireless sensor networks. *EURASIP Journal on Wireless Communications and Networking*, *2019*(1), 175.
23. Chan, Y. T., Tsui, W. Y., So, H. C., & Ching, P. C. (2006). Time-of-arrival based localization under NLOS conditions. *IEEE Transactions on Vehicular Technology*, *55*(1), 17–24.
24. Mirjalili, S. (2016). SCA: A sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems*, *96*, 120–133.
25. Larrañaga, P., & Lozano, J. A. (Eds.). (2001). *Estimation of distribution algorithms: A new tool for evolutionary computation*, (Vol. 2). Berlin: Springer.
26. Chu, S. C. (2015). A compact artificial bee colony optimization for topology control scheme in wireless sensor networks.
27. Pan, J. S., & Dao, T. K. (2019). A compact bat algorithm for unequal clustering in wireless sensor networks. *Applied Sciences*, *9*(10), 1973.
28. Liang, J. J., Qu, B. Y., Suganthan, P. N., & Hernández-Díaz, A. G. (2013). 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*, *201212*(34), 281–295.
29. Song, P. C., Pan, J. S., & Chu, S. C. (2020). A parallel compact cuckoo search algorithm for three-dimensional path planning. *Applied Soft Computing*, 106443.
30. Seljak, U., & Zaldarriaga, M. (1996). A line of sight approach to cosmic microwave background anisotropies. arXiv preprint astro-ph/9603033.
31. Topcuoglu, H. R., Ermis, M., & Sifyan, M. (2010). Positioning and utilizing sensors on a 3-D terrain part I-Theory and modeling. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 41*(3), 376–382.
32. Sun, C., Jin, Y., Cheng, R., Ding, J., & Zeng, J. (2017). Surrogate-assisted cooperative swarm optimization of high-dimensional expensive problems. *IEEE Transactions on Evolutionary Computation*, *21*(4), 644–660.
33. Pan, J. S., Hu, P., & Chu, S. C. (2019). Novel parallel heterogeneous meta-heuristic and its communication strategies for the prediction of wind power. *Processes*, *7*(11), 845.
34. Pan, J. S., Kong, L., Sung, T. W., Tsai, P. W., & Snížel, V. . (2018). A clustering scheme for wireless sensor networks based on genetic algorithm and dominating set. *Journal of Internet Technology, 19*(4), 1111–1118.
35. Shieh, C. S., Sai, V. O., Lee, T. F., Le, Q. D., Lin, Y. C., & Nguyen, Trong-The. (2017). Node localization in WSN using heuristic optimization approaches. *Journal of Network Intelligence*, *2*(3), 275–286.
36. Tang, Z., Xue, X., Wang, J., & Hang, Z. *The logic sense request of WSN and its analysis model*.
37. Meng, Z., Pan, J. S., & Tseng, K. K. (2019). PaDE: An enhanced differential evolution algorithm with novel control parameter adaptation schemes for numerical optimization. *Knowledge-Based Systems*, *168*, 80–99.
38. Li, N., Li, G., & Deng, Z. (2017, July). An improved sine cosine algorithm based on levy flight. In *Ninth international conference on digital image processing (ICDIP 2017)* (Vol. 10420, p. 104204R). International Society for Optics and Photonics.
39. Censor, Y., & Zenios, S. A. (1997). *Parallel optimization: Theory, algorithms, and applications*. Oxford: Oxford University Press on Demand.
40. Bäck, T. (1994, October). Parallel optimization of evolutionary algorithms. In *International conference on parallel problem solving from nature* (pp. 418–427). Berlin: Springer.