Wireless Networks
https://doi.org/10.1007/s11276-022-03048-z

ORIGINAL PAPER

Improved DV-Hop based on parallel and compact whale optimization algorithm for localization in wireless sensor networks

Ruo-Bin Wang[1,2] · Wei-Feng Wang[1] · Lin Xu[3] · Jeng-Shyang Pan[4] · Shu-Chuan Chu[4,5]

Accepted: 31 May 2022
© The Author(s) 2022

Abstract
Improving localization performance is one of the critical issues in Wireless Sensor Networks (WSN). As a range-free localization algorithm, Distance Vector-Hop(DV-Hop) is well-known for its simplicity but is hindered by its low accuracy and poor stability. Therefore, it is necessary to improve DV-Hop to achieve a competitive performance. However, the comprehensive performance of WSN is limited by computing and storage capabilities of sensor nodes. In this paper, we propose an algorithm with parallel and compact techniques based on Whale Optimization Algorithm (PCWOA) to improve DV-Hop performance. The compact technique saves memory consumption by reducing the original population. The parallel techniques enhance the ability to jump out of local optimization and improve the solution accuracy. The proposed algorithm is tested on CEC2013 benchmark functions and compared with some popular algorithms and compact algorithms. Experimental results show that the improved algorithm achieves competitive results over compared algorithms. Finally, simulation research is conducted to verify the localization performance of our proposed algorithm.

Keywords Whale optimization algorithm · Compact technique · Parallel technique · DV-Hop · Wireless sensor networks

1 Introduction

In recent decades, Wireless Sensor Networks (WSN) have been widely applied in various fields due to its low cost, reliability, scalability, flexibility, and ease of deployment [1–3]. Some essential elements, such as routing, load balancing, node localization, and data aggregation will affect the performance of WSN [4, 5]. Usually, a good performance of WSN is achieved through the tradeoff between costs and outcome, such as limited energy, memory consumption or computing capability of sensor nodes and a competitive accuracy or efficiency [6–10]. Where node localization is an important issue that directly affects the performance of WSN. In this issue, a robust algorithm can provide a highly accurate localization solution, and at the same time can avoid coverage holes and connection failures caused by random deployment of nodes [11]. Hence, how to design a locating algorithm to achieve a refined tradeoff between limited resources of nodes and high location accuracy of nodes has become one of the challenges in WSN.

The traditional localization algorithms are categorized as range-based and range-free [6–8]. The range-based algorithms, including Time of Arrival(TOA), Time of Difference of Arrival(TODA), Angle of Arrival(AOA) and Received Signal Strength Indicator(RSSI), utilize absolute point-to-point distance between neighbor nodes to achieve the localization results. However, improvements on accuracy depend on extra hardware and therefore increase costs.

Ruo-Bin Wang and Wei-Feng Wang contributed equally to this work and should be considered co-first authors.

✉ Ruo-Bin Wang
robin945@163.com

✉ Lin Xu
xuyly032@mymail.unisa.edu.au

1 School of Information Science and Technology, North China University of Technology, Beijing 100043, China

2 Beijing Urban Governance Research Center, North China University of Technology, Beijing 100043, China

3 STEM, University of South Australia, Adelaide 5095, Australia

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

5 College of Science and Engineering, Flinders University, 1284 South Road, Tonsley, SA 5042, Australia

Published online: 13 July 2022

Content courtesy of Springer Nature, terms of use apply. Rights reserved.

Different from range-based algorithms, range-free algorithms' performance does not closely depend on hardware. Some typical range-free localization algorithms include Amorphous, Centroid, Approximate Point-In Triangle Test (APIT), and Distance Vector-Hop (DV-Hop). DV-Hop is popular for its low environmental impact, low cost and power consumption, wide locating coverage, and it depends only on a network topology to realize localization [9, 10]. Nevertheless, it is still a challenge for locating unknown nodes accurately with original DV-Hop. The problems are caused by the deviation between the unknown node position estimated by the trilateral method and the actual position [12]. From the perspective of computation, the locating issue can be expressed as an NP-hard optimization problem, which can be solved by optimization techniques [13]. Yet, with the increase of nonlinearity and constraints, the traditional optimization techniques often can not find the global optimal solution. Recently, Meta-Heuristic Optimization Algorithms(MHOAs) have received extensive attention in the scientific community and have made significant progress. They are mainly used to address complex optimization issues and are considered as effective and reliable optimization techniques [14, 15].

Whale Optimization Algorithm (WOA) is a MHOA proposed by Mirjalili in recent years. It is widely employed in the issues of optimization because of its simple operation, few adjustment parameters, fast optimization, and self-adaptive mechanism [16]. WOA mimics the hunting behavior of humpback whales using a distinctive bubble-net attacking strategy [17]. Compared to other well-known algorithms, its unique attacking strategy helps it gain competitive exploration capabilities, and its self-adaptive mechanism can balance the phases of exploration and exploitation. Moreover, the self-adaptive mechanism of WOA can facilitate the searching for promising regions in the search space of the initial steps of iterations. Therefore, it gets closer to the optimum faster after passing through nearly half of the iterations [16, 17]. Whereas in dealing with complex optimization problems, it is prone to fall into local optimums, and sometimes it does not converge. In response to these shortcomings, researchers have improved WOA, such as chaotic WOA [18, 19], improved WOA [20, 21], binary WOA [22, 23], hybrid WOA [24, 25], and multi-objective WOA [26, 27]. In recent years, WOA has been employed in range-based node localization algorithms [28–32], but few applications in DV-Hop. For instance, Chai et al. [33] only utilized parallel techniques to improve WOA and applied it to enhance the performance of DV-Hop, but they do not take into account the memory consumption and storage capacity of the node. In this paper, we utilize compact technique to compress the memory of algorithm, then we propose novel parallel techniques to improve the solution accuracy without adding additional equipment.

The main contributions of the paper are listed as follows:

- (i) A Parallel and Compact WOA (PCWOA) is proposed. This paper adopts a compact technique to compress the algorithm, and then proposes novel parallel techniques to enhance the ability to escape from the local optimal and improve the computing capability of the algorithm and solution accuracy.
- (ii) The PCWOA was tested with CEC2013 benchmark functions, and compared with some compact and popular algorithms to verify the competence of our proposed algorithm.
- (iii) The PCWOA is combined with DV-Hop, and multiple simulation experiments are carried out to verify the algorithm's performance on accuracy and stability of location optimization.

The rest of the paper is organized as follows: Sect. 2 briefly introduces a review of related articles in the literature. Discuss the original WOA and the PCWOA in detail in Sect. 3. Section 4 discusses the experiments and results of PCWOA based on CEC2013 benchmark functions. Section 5 introduces the combination of the PCWOA algorithm and DV-Hop algorithm. Section 6 is to use the improved algorithm to implement the localization in WSN. Section 7 concludes this paper.

2 Related works

In this section, we firstly introduce the literatures about improvements merely on DV-Hop algorithm, secondly, we introduce the literatures about MHOAs employing directly on DV-Hop algorithm, thirdly the literatures about improved MHOAs on DV-Hop algorithm are introduced, and finally introduce the literatures about the application of WOA on DV-Hop algorithm and other localization algorithms in WSN.

Chen et al. [34] proposed an improved DV-Hop localization. This algorithm mainly utilizes the least squares method to optimize the average hop-size and proposes a dynamic weight coefficient to correct the average hop-size. Tomic et al. [35] proposed three improved algorithms by adding several geometric improving steps based on retaining the basic steps of the original DV-Hop algorithm. These methods increase the locating accuracy in a certain extent, also increased is the cost for the computational complexity. Moreover, these methods essentially use the Least Square Method(LSM) to calculate the position of an unknown node, which is the reason for errors of DV-hop algorithm.

Content courtesy of Springer Nature, terms of use apply. Rights reserved.

Recently, there has been a series of literature about MHOA immediately combined with DV-Hop in WSN to achieve a better performance. Kanwar et al. [36] proposed a new framework of localization for displaced sensor nodes using Particle Swarm Optimization (PSO). This framework reduces communication between unknown and anchor nodes by calculating the hop size of all anchor nodes on unknown nodes. It effectively minimizes the elapsed time, localization errors, and energy consumption. However, compared with the traditional algorithms, only two of the six schemes proposed in the literature have less localization error and energy consumption results. Therefore, its performance is unstable. Ghagour et al. [37] used the recently developed Squirrel Search Algorithm (SSA) to improve DV-Hop. This scheme achieved higher localization accuracy, stability, and a faster convergence rate. However, it does not consider the impact of node memory and communication radius. Cui et al. [38] presented a high accurate localization algorithm with DV-Hop and Differential Evolution (DE). This paper lessened the hop-size value by the number of common one-hop nodes between adjacent nodes and converted the discrete values of hop-size to continuous values to achieve a better accuracy. However, the DE algorithm has its own defects, such as complex operation and too many operators involved in the calculation, which causes the over consumption of memory and therefore shortens the lifetime of sensor nodes.

The literatures above mainly focused on improvements of DV-Hop algorithm but not on MHOAs themselves. Generally, MHOAs have their own shortcomings, such as slow convergence speed, inability to converge and being prone to trap into local optimum, therefore, employing original MHOAs directly to improve DV-hop will be counterproductive. Moreover, the direct combination of the two algorithms will significantly increase the overall memory consumption and lead to excessive energy consumption of sensor nodes, along with the reduction of transmission rate between sensor nodes. Hence, it is reasonable to boost the performance of the original MHOAs to improve DV-Hop algorithm.

Ouyang et al. [39] proposed an improved Genetic Algorithm (GA) instead of LSM . They randomly exchanged the execution order of mutation operators and crossover operators through a probability mechanism and used a dynamic adjustment mechanism to update parameters dynamically. However, this scheme does not perform well when processing nodes with large radius. And its performance is also affected by the number of unknown node hops: the more hops, the greater the error. Chen et al. [40] weighted the average hop-size of each anchor node, and optimized the location estimated by the 2-dimensions hyperbolic localization algorithm with PSO. In their research, they weight the average hop-size of each anchor, which will lengthen the amount time of calculation and increase the consumption of memory. Cui et al. [41] designed a Cuckoo Search algorithm (CS) that improved the cyber-physical system's DV-Hop performance. This algorithm shows that the one with hybrid distribution combined with the Levy distribution and Cauchy distribution achieves the best performance. Chai et al. [33] proposed a parallel WOA and applied it to optimize the localization of WSN. Parallel techniques can effectively improve the searchability and population diversity of the algorithm. The experimental results show that the proposed parallel WOA can achieve better results. However, this scheme did not consider the influence of node memory, which led to performance degradation in large-scale networks. Li et al. [42] proposed a parallel compact Cat Swarm Optimization (CSO) and applied it to DV-Hop. It effectively improved the localization accuracy and saved the WSN memory. However, the CSO algorithm has defects of poor convergence and over consumption.

To the present author's knowledge, most of the literatures on applications of WOA in WSN localization use range-based techniques [28–32]. The experimental results show that the proposed schemes achieve better localization accuracy, delivery radio, and delay. However, the range-based localization algorithm needs to measure the angle or distance between the anchor node and the unknown node. It also requires additional hardware, which causes high costs in large-scale deployment.

Compared with the existing methods based on MHOAs, the algorithm proposed in this paper has the following advantages:

(a) The WOA selected in this paper has advantages of simple structure, adaptive-mechanism, and robustness, which makes it be competitive.
(b) The compact technique effectively compresses the number of the original population, and therefore reduces the consumption of computing resources, which is essential for node processors.
(c) The novel parallel techniques improve the capability of avoiding trapping into a local optimal and accelerate convergence, which will compensate the loss of accuracy caused by compression.

3 Original WOA and its variants

3.1 Original WOA

WOA is inspired by the hunting behavior of humpback whales. There are three phases: encircling prey, bubble-net attacking method, and searching for prey.

Content courtesy of Springer Nature, terms of use apply. Rights reserved.

In the real world, a whale can make out the position of prey, but the localization cannot be confirmed in the algorithm. Therefore, the algorithm assumes that after initializing the whale population, the current best whale localization is selected as the prey localization, and the rest of the whales approach it. The mathematical model of encircling prey is defined as the following equations:

$$X^{t+1} = X^t_{best} - A \times |R \times X^t_{best} - X^t|, \tag{1}$$

where t is the current iteration, X_{best} and X indicate the position of the best solution obtained so far and the current solution, respectively. A, R are variables and calculated as follows:

$$A = 2ar - a, \tag{2}$$

$$R = 2r, \tag{3}$$

where r is a random value between 0 and 1. The a is linearly decreased from 2 to 0 in the iteration process, and it is calculated as:

$$a = 2 - 2 \times (\frac{t}{T}), \tag{4}$$

where T is the max iteration, the value of a changes along with iteration, which affects the fluctuation range of A. Combined with Eqs. (2) and (4), it is calculated that A is a random value in the interval $[-2, 2]$.

WOA utilizes the adaptive variation of A to ensure a smooth transition between exploration and exploitation. When $|A| \geq 1$, the algorithm does a lot of searching in the exploration phase until it finds the region with a promising solution. When $|A| < 1$, the algorithm enters the phase of exploitation. The solutions can move anywhere between the current and best solutions and are infinitely close to the best solution during the exploitation phase.

In the exploitation phase, the bubble-net attacking method imitates the hunting behavior of whales, and it consists of two approaches: shrinking encircling mechanism and spiral updating position. The former depends on the encircle encircling range determined by A in the Eq. (2). The a decrease along with the encircle range smaller. The latter utilizes the *cos* function to create a spiral equation that mimics the helix-shaped movement of whales. It is defined as follows:

$$X^{t+1} = |X^t_{best} - X^t| \times e^{ch} \times \cos(2\pi h) + X^t_{best}, \tag{5}$$

where c is a constant for defining the shape of a logarithmic spiral, and h is a random number in the interval $[-1, 1]$. During the bubble-net attacking method, the algorithm uses a probability p to choose which way to surround. When $p \geq 0.5$, the algorithm selects the latter, or else, selects the former.

The best solution is chosen initially, and other individuals are close to it. This mechanism can easily make the algorithm fall into local optimum, which need a large number of exploration to search prey. The mathematical model is as below equation:

$$X^{t+1} = X^t_{rand} - A \times |R \times X^t_{rand} - X_t|, \tag{6}$$

where X_{rand} is a random solution chosen from the current solution.

3.2 Compact WOA

Considering the node memory, this paper uses compact technique to reduce the number of populations of the algorithm. The compact technique [44–48] utilizes the distribution characteristic of the original population to construct a probability model. Then, in the iterative process, the probability model is used for algorithm operation to generate new solutions. By comparing the generated solutions, the probability model is updated to replace the original population of the algorithm. The essence of compact technique is population-less. It develops a virtual population instead of the actual population to solve the problem in the process and achieves less memory usage and shorter computing time.

As mentioned above, this virtual population is a probability model of population solutions and is encoded in a data structure. The data structure is named *PerturbationVector* and indicated with PV. Specifically, there are N particles in a population from a macro perspective, and each particle has D dimensions. So, the PV is a $K \times D$ matrix. After adding the compact technique, each dimension can be indicated with a normal distribution, and the PV becomes a $2 \times D$ matrix:

$$PV_c = [\mu_c, \sigma_c], \tag{7}$$

where c represents the current iterations, μ and σ are, respectively, for each design variable, mean and standard deviation value of PV. To keep the algorithm steady, each pair of mean and standard deviation corresponds to a Gaussian Probability Density Function (PDF), which is truncated within the interval $[-1, 1]$ and normalized to an area with an amplitude of 1.

The sampling mechanism of compact technique is associated with the design variable $x[i]$. The candidate solution x from PV requires extensive explanation. For each design variable indexed by i, a truncated Gaussian PDF with the mean $\mu[i]$ and standard deviation $\sigma[i]$ are related. PDF is defined as follows:

Content courtesy of Springer Nature, terms of use apply. Rights reserved.

$$PDF(truncNorm(x)) = \frac{\sqrt{\frac{2}{\pi}} \times e^{-\frac{(x-\mu[i])^2}{2\sigma[i]^2}}}{\sigma[i]\left(erf\left(\frac{\mu[i]+1}{\sqrt{2\sigma[i]}}\right) - erf\left(\frac{\mu[i]-1}{\sqrt{2\sigma[i]}}\right)\right)}, \tag{8}$$

where *erf* is the error function. By consulting a Chebyshev Polynomial, the PDF can correspond to a Cumulative Distribution Function (CDF) with randomly changed from 0 to 1, and the relationship between PDF and CDF is $CDF = \int_0^1 PDF(x)dx$. The CDF can be used to describe the value of the actual solution of the population by a similar probability distribution, while PDF is the macroscopic probability distribution constructed by the population characteristics. The solution $x[i]$ can be randomly generated from the inverse CDF using the *PV* vector. Thus, the CDF can be written as:

$$CDF = \frac{erf\left(\frac{\mu+1}{\sqrt{2}\delta}\right) + erf\left(\frac{x-\mu}{\sqrt{2}\delta}\right)}{erf\left(\frac{\mu+1}{\sqrt{2}\delta}\right) - erf\left(\frac{\mu-1}{\sqrt{2}\delta}\right)}, \tag{9}$$

During the execution of the algorithm, the *PV* vector needs to be continuously updated. The solution generated by PV is compared with the previous solution to determine which is better. The better is marked as a *winner*, and the worse is marked as a *loser*. Use the *winner* and *loser* to update μ and σ in the *PV* vector. The update formula is as follows:

$$\mu_i^{c+1} = \mu_i^c + \frac{1}{N_p}(winner_i - loser_i), \tag{10}$$

$$\sigma_i^{c+1} = \sqrt{(\sigma_i^c)^2 + (\mu_i^c)^2 - (\mu_i^{c+1})^2 + \frac{1}{N_p}(winner_i^2 - loser_i^2)}, \tag{11}$$

In the equation, N_p represents the actual population, μ_i^{c+1}, σ_i^{c+1} represents the mean value and standard deviation after updating, and the initial value of μ and σ are 0 and 10.

Based on the above description of the compact technique, the compact WOA (CWOA) can be implemented, including the following steps. First, generate a random number x with uniform distribution from 0 to 1. Second, use *PV* to generate a solution by the inverse function of CDF. The inverse function of CDF is defined as:

$$y = \sqrt{2}\delta erf^{-1}\left(xerf\left(\frac{\mu+1}{\sqrt{2}\delta}\right) - xerf\left(\frac{\mu-1}{\sqrt{2}\delta}\right) - erf\left(\frac{\mu+1}{\sqrt{2}\delta}\right)\right) + \mu, \tag{12}$$

where $(erf)^{-1}$ is the inverse function of *erf*. Since the range of solutions generated by using the inverse CDF is between -1 and 1. Third, we need to map the value of y to the actual decision space through the following equation.

$$x_{actual} = y \times ((ub - lb))/2 + ((ub + lb))/2. \tag{13}$$

where *ub* and *lb* are the upper and lower bounds of the actual decision space. A new solution is finally generated by executing the algorithm. The fitness value of the new solution is compared with the fitness value of the stored best solution to determine the *winner* and *loser*, and then use *winner* and *loser* update μ, σ, and *PV*.

The pseudo-code of CWOA is shown in Algorithm 1, where d represents the dimension of the problem, and *ub* represents the upper bound of the actual decision space. Use f_{min}, fit_{winner} to store the fitness value of the global optimal solution and *winner*. *Inf* represents an infinite number.

Algorithm 1 Compact WOA

```
Require: Parameters: $d$, $ub$, $lb$ and $N$.
Ensure: Global optimum $best$ and its fitness value $f_{min}$.
1:  for $i = 1 \rightarrow d$ do
2:      $initialize\ \mu = 0, \sigma = \lambda, \lambda = 10$;
3:  end for
4:  $initialize\ best = ub, f_{min} = \inf$;
5:  while $i < Max\_iteration$ do
6:      Get $x_1$ from $PV$ via equations (12),(13);
7:      Update $x_1$ to get $x_2$ via equation (1);
8:      $[winner, loser]$ =compete($x_1, x_2$);
9:      for $i = 1 \rightarrow d$ do
10:         Update $PV$ via equations (10),(11);
11:         if $fit_{winner} < f_{min}$ then
12:             $best = winner$; $f_{min} = fit_{winner}$;
13:         end if
14:     end for
15: end while
```

Content courtesy of Springer Nature, terms of use apply. Rights reserved.

3.3 Parallel techniques

After adding a compact technique, the number of populations involved in the calculation of the algorithm will be reduced. The number of populations changes from k to 2. In this paper, the value of k is 30. The compact technique saves memory and shortens the computation time of an algorithm. Still, it easily makes the algorithm fall into local optimal and loses the accuracy of the solution. Therefore, the parallel technique is employed to avoid the defects. Parallelism [49–54], from a macro perspective, is that multiple groups operate simultaneously. Communication and exchange of information among groups can be realized by parallelism, and its common idea uses better solutions from some groups to replace worse solutions from other groups. In this paper, a novel parallel technique is proposed, which includes three different methods. Two groups are randomly selected and compared from the current, random, and best group. The details of the parallel techniques are as follows:

(a) Technique one: Randomly select a group to compare with the current group. If the fitness value of the current group is worse, the fitness value of the random group will replace it. Specifically, the solutions in the two groups are sorted in descending order, and the average of the solutions in the two groups is calculated. If the average value of the current group reaches a better solution, the new group consists of the second half of the two groups. Then, compare the fitness value of the current group and the new group. The better result replaces the worse. Figure 1 shows the core idea of technique one.

(b) Technique two: The current group is compared with the global optimum to determine whether to restructure the current group. If not, the best fitness value of the current group is assigned to the global optimum. Otherwise, sort the solutions stored in the global in descending order and average the solutions. If the average of the current group is better, a new group is constructed in the same way as technique one. The purpose of forming a new group is to determine whether to update the global optimal fitness value and the corresponding position. The process of technique two is shown in Fig. 2.

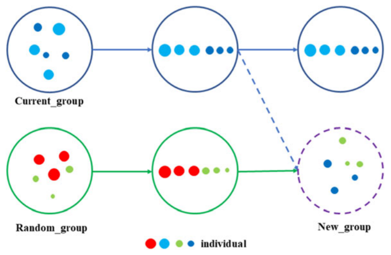


Fig. 1 The parallel technique one(The current group was compared with the random group.)

(c) Technique three: Technique three compares the fitness value of the random group with the global optimum. The update strategy and implementation steps are the same as technique two.

The parallel techniques proposed in this paper are to use the current solution and random solution to affect the global optimal solution, strengthen the communication between groups and improve the accuracy of the solution.

According to the above introduction, the parallel compact WOA (PCWOA) has been constructed, and Algorithm 2 shows its pseudo-code.

Content courtesy of Springer Nature, terms of use apply. Rights reserved.

Algorithm 2 Compact WOA with parallel technique

```
Require: Parameters: $d$, $ub$, $lb$, $N$ and $g$.
Ensure: Global optimum $Globalbest$ and its fitness value $Globalfmin$.
1: Set the number of groups $g$, each group is $G_i (i \le g)$;
2: $Initialize\ G[i].PV, G[i].best\ and\ G[i].f_{min}\ of\ each\ group$;
3: $Initialize\ Globalbest = G[1].best, Globalfmin = G[1].f_{min}$;
4: while $t < Max\_iteration$ do
5:     for $i = 1 \to g$ do
6:         Get $x_1$, $x_2$ via equations (12),(13) and (1);
7:         $[winner, loser]$ =compete($x_1, x_2$);
8:         for $i = 1 \to d$ do
9:             Update $G[i].PV$ via equations (10),(11);
10:        end for
11:        if $fit_{winner} < G[i].f_{min}$ then
12:            $G[i].best = winner$; $G[i].f_{min} = fit_{winner}$;
13:        end if
14:        Update $Globalbest$ and $Globalfmin$;
15:        if $rand < 0.5$ then
16:            if $rand < 0.5$ then
17:                Parallel technique one;
18:            else
19:                Parallel technique two;
20:            end if
21:        else
22:            Parallel technique three;
23:        end if
24:    end for
25:    Change the worst individual to the best individual by the sorting statement;
26:    $t = t + 1$;
27: end while
```

According to Algorithm 2, *rand* is a random number between 0 and 1. The initial method of *PV*, *best*, and f_{min} of each group is the same as Algorithm 1. $G[i].best$ is used to store the best in the group, and $G[i].fmin$ is used to store the fitness value corresponding to $G[i].best$. Here, fit_{winner} represents the fitness value of the *winner*. *Globalbest* and *Globalfmin* represent the optimal solution and fitness value in all groups.

3.4 Time and space complexity of PCWOA

In this section, the time and space complexity of PCWOA are discussed. Mirjalili et al. [16] proved the time complexity of standard WOA. For the convenience to compare, this section redefines some variable names. Here, I stands for the maximum number of iterations, D for the dimension, G for the number of groups, and P for the number of populations. The details of the time and space complexity of WOA, CWOA, and PCWOA are shown in Table 1.

According to Table 1, compared with WOA, the CWOA has a specific improvement in time and space complexity. The time complexity and space complexity of PCWOA depend on the number of groups. Generally, the number of groups in the parallel technique does not exceed 5. In this paper, the number of groups is 3, and the population is 30. Thus, PCWOA outperform WOA both on time and space complexity.

4 Experimental results and analysis

4.1 Benchmark functions and algorithm parameters

In this subsection, firstly, we verify the performance of the PCWOA, which is experimentally tested on CEC2013 [55]. It contains 28 benchmark functions, in which F1-F5 are unimodal functions, F6-F20 are multi-modal functions, and F21-F28 are composition functions.

Secondly, in order to further test the performance of PCWOA, we will compare it with WOA and CWOA. In addition, PCWOA is compared with other compact

Content courtesy of Springer Nature, terms of use apply. Rights reserved.

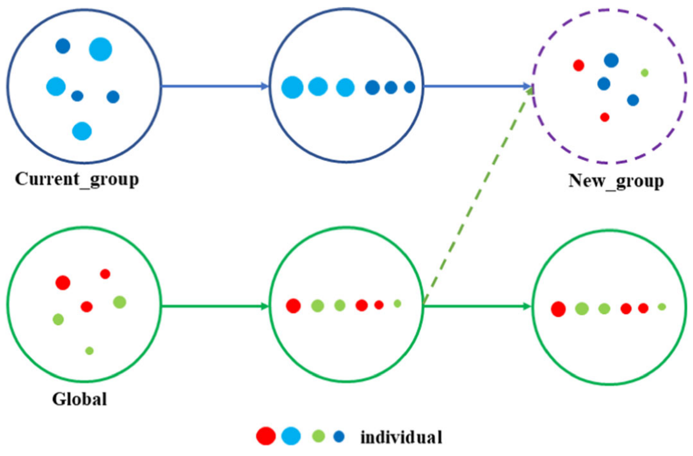


Fig. 2 The parallel technique two(The current group was compared with the global optimum.)

Table 1 Comparing the complexity of the algorithms

Algorithm	WOA	CWOA	PCWOA
Time complexity	O($I * P * D$)	O($I * D$)	O($I * D * G$)
Space complexity	O(P)	O(2)	O(G)

algorithms, such as compact PSO and compact SCA. All algorithm parameters are listed in Table 2.

All algorithms are implemented in MATLAB R2019b, and are conducted on a computer with an Intel Core i5 @3.10GHz CPU and 16GB memory running 64-bit Windows 10.

4.2 Compared with WOA and its variants

According to Algorithm 1 and Algorithm 2, this paper proposed two different versions of improved WOA including CWOA and PCWOA. The parameters of the two algorithms are listed in Table 2. In the 30-dimensions, the two algorithms will possess multiple variables: x, $best$, μ, and σ. In the execution of the algorithm, the consumption of memory is different due to the difference in the technology used by each algorithm. The total number of iterations is the same so that the evaluations of each algorithm are identical, and the $Max\_iteration$ is set to 1000.

Then, use CEC2013 benchmark functions to test algorithm performance. Each algorithm is run 30 times, respectively, and the average value is used to measure the performance of the algorithm. The *Win* represents the number of optimal average values obtained by the algorithm in 28 benchmark functions.

According to the data in Table 3, PCWOA outperforms other algorithms on most of functions. Among the 28 benchmark functions, the PCWOA obtains the 23 best results. Moreover, the improvements in $F1$, $F2$, $F4$, $F5$, and $F17$ are noticeable, ten times more than the original. The WOA with only a compact technique performs commonly on most benchmarks and only achieves good results on $F7$, $F25$, and $F27$. Data in Table 3 show that compact technique is easy to lead the algorithm fall into local optimal, such as the performance of the algorithm on $F1$, $F3$, $F5$, $F10$, and $F19$. In sum, using only compact technique to improve the algorithm may backfire on the performance of the algorithm. We must consider the loss of accuracy when significantly reduce the population participating in the calculation of the algorithm.

In Table 3, the average value obtained by the algorithm in many test functions has little difference, which may not manifest the advantages of PCWOA. Therefore, on this basis, this paper evaluates the solution accuracy through the convergence curve of the algorithm. According to Fig. 3, the optimization and the convergence ability of PCWOA is similar to that of WOA in Fig. 3(a), (c), (f), (m), (o). On other test functions, PCWOA can always find a better optimal solution, especially in Fig. 3(d), (e), (j), (k), (p), (q), (s). It can also be seen from Fig. 3 that WOA quickly falls into convergence. The parallel technique proposed in this paper effectively enhances the ability of the algorithm to jump out of the local optimum. However, the solution accuracy of CWOA is not good, and the convergence speed is slow. The above results show that compact technique can save algorithm memory effectively, but not beneficial for accuracy and convergence.

4.3 Compared with other compact algorithms

In this section, PCWOA is compared with other compact algorithms. This paper selects compact BA, compact SCA, and compact SCA with multiple groups and strategies(MCSCA). The purpose of choosing compact SCA and MCSCA is to highlight further the advantages of adding parallel technique to compact technique and highlight the benefits of the parallel technique proposed in this paper. The parameters of these algorithms are listed in Table 2,

Content courtesy of Springer Nature, terms of use apply. Rights reserved.

Table 2 Parameter setting for related algorithms

Algorithm	Parameters
WOA	Virtual $Np = 30, ub = 100, lb = -100, c = 1;$
CWOA	Virtual $Np = 30, ub = 100, lb = -100, c = 1, \lambda = 10;$
PWOA	Virtual $Np = 30, ub = 100, lb = -100, c = 1, groups = 3;$
PCWOA	Virtual $Np = 30, ub = 100, lb = -100, c = 1, \lambda = 10, groups = 3;$
MCSCA	Virtual $Np = 30, ub = 100, lb = -100, \lambda = 10, groups = 10, F_{1,2,3,4} = 0.5;$
CBA	Virtual $Np = 30, ub = 100, lb = -100, \lambda = 10, A = 0.5, r = 0.5;$
CSCA	Virtual $Np = 30, ub = 100, lb = -100, \lambda = 10;$
SCA	Virtual $Np = 30, ub = 100, lb = -100;$
MVO	Virtual $Np = 30, ub = 100, lb = -100, P = 6, WEP_{Min} = 0.2, WEP_{Max} = 1;$
AOA	Virtual $Np = 30, ub = 100, lb = -100, \alpha = 5, \mu = 0.5;$
SSA	Virtual $Np = 30, ub = 100, lb = -100;$
PSO	Virtual $Np = 30, ub = 100, lb = -100, c1 = c2 = 2, w \in [0.4, 0.9];$

Table 3 The average value is obtained by each algorithm running 30 times

F	WOA	CWOA	PCWOA
F1	−4.24E+02	2.14E+04	**−1.35E+03**
F2	1.17E+08	2.08E+08	**6.45E+07**
F3	6.39E+10	1.18E+11	**3.51E+10**
F4	1.03E+05	7.72E+04	**7.45E+04**
F5	8.99E+00	4.17E+03	**−8.34E+02**
F6	−6.04E+02	9.12E+02	**−7.46E+02**
F7	4.90E+03	**−5.20E+02**	3.30E+02
F8	−6.79E+02	−6.79E+02	**−6.79E+02**
F9	**−5.60E+02**	−5.65E+02	−5.64E+02
F10	1.88E+02	2.21E+03	**−3.09E+02**
F11	1.56E+02	1.65E+02	**1.55E+02**
F12	3.12E+02	2.45E+02	**2.41E+02**
F13	4.02E+02	3.24E+02	**3.14E+02**
F14	6.06E+03	6.97E+03	**5.65E+03**
F15	6.74E+03	7.06E+03	**5.88E+03**
F16	2.02E+02	2.03E+02	**2.02E+02**
F17	1.02E+03	1.03E+03	**9.75E+02**
F18	1.05E+03	1.11E+03	**1.05E+03**
F19	7.16E+02	2.76E+04	**5.53E+02**
F20	7.16E+02	6.15E+02	**6.14E+02**
F21	1.73E+03	2.90E+03	**1.20E+03**
F22	7.82E+03	8.40E+03	**7.42E+03**
F23	8.39E+03	8.39E+03	**7.16E+03**
F24	1.32E+03	1.30E+03	**1.29E+03**
F25	1.43E+03	**1.39E+03**	1.40E+03
F26	1.57E+03	**1.43E+03**	1.49E+03
F27	2.68E+03	2.54E+03	**2.53E+03**
F28	**2.68E+03**	5.88E+03	5.67E+03
Win	2	3	**23**

Bold parts indicate that the average value is the best

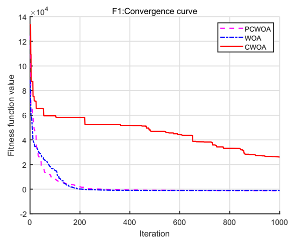
and each algorithm runs 30 times. In Table 3 and Fig. 3, the performance of PCWOA has been verified better than WOA, therefore WOA is not added to the comparison in this section.

The algorithms are compared by Mean, Standard Deviation, and Friedman ranking tests. The results of the algorithms are shown in Table 4. The *Ave* and *Std* represent the mean value and standard deviation value of the algorithm. The Rank indicates the ranking of the algorithm, and the Mean represents the average ranking of algorithms. According to Table 4, it is found that the mean and standard deviation of the algorithm with parallel technique and compact technique have better performance than those with the compact technique only. According to the Friedman ranking test, PCWOA and MCSCA ranked the first and the second, respectively, compact BA ranked the third, and compact SCA ranked the fourth. The ranking results show that PCWOA is much ahead of the average ranking of MCSCA. Unfortunately, PCWOA does not always perform well, such as on $F8$, $F11$, $F12$, $F13$, $F18$, $F24$, and $F28$. But among the 28 test functions, PCWOA took the lead in 20 test functions.

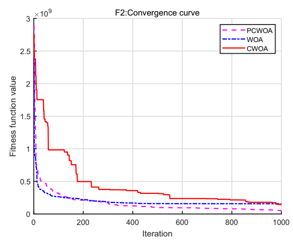
This paper again uses the convergence curve to evaluate the performance of the algorithm. Fig. 4 shows the convergence curve of compact algorithms. From Fig. 4a–p, PCWOA can find the optimal solution. Although the convergence speed is not as fast as MCSCA in Fig. 4(h), (i), (m), (n), its solution accuracy is improved by 23%, 14%, 33%, and 27% compared with other algorithms, respectively.

To sum up, the results in Tables 3 and 4 show that the performance obtained by adding only the compact technology are not good enough. Some results are improved, while others are not as good as the original ones. The results obtained by utilizing both compact technique and

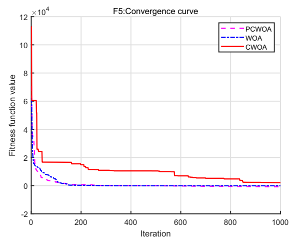
Content courtesy of Springer Nature, terms of use apply. Rights reserved.



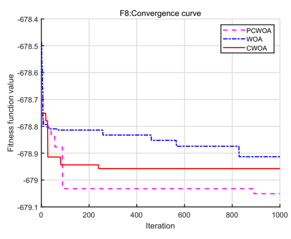
(a) F1



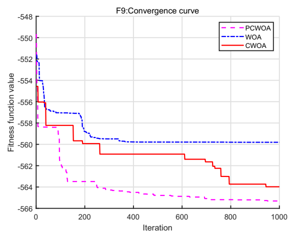
(b) F2



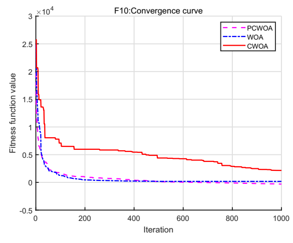
(c) F5



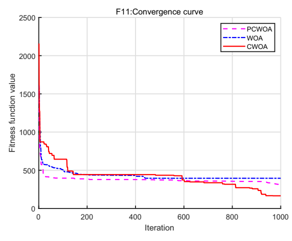
(d) F8



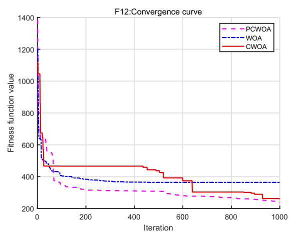
(e) F9



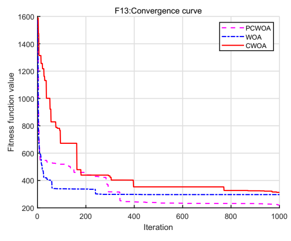
(f) F10



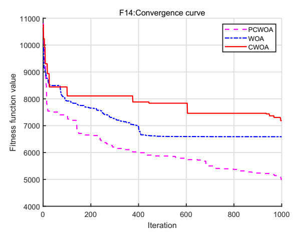
(g) F11



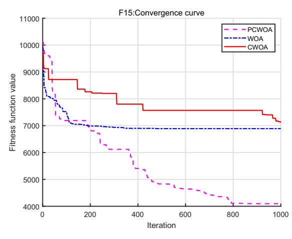
(h) F12



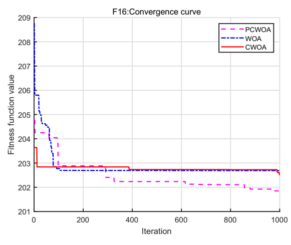
(i) F13



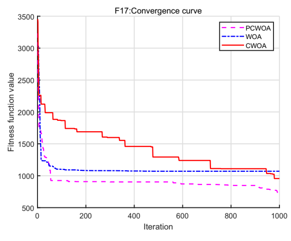
(j) F14



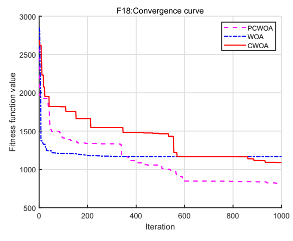
(k) F15



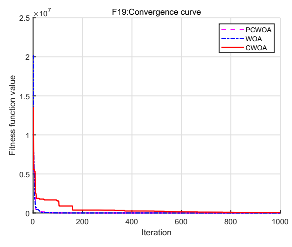
(l) F16



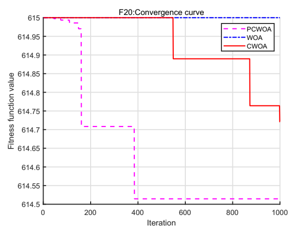
(m) F17



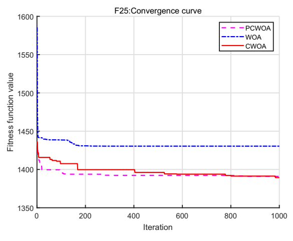
(n) F18



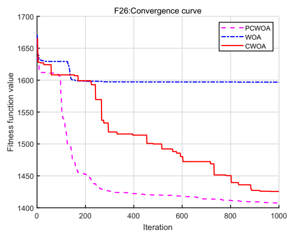
(o) F19



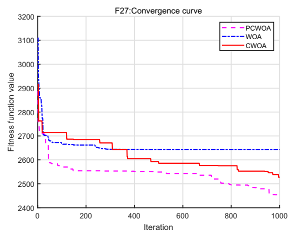
(p) F20



(q) F25



(r) F26



(s) F27

Fig. 3 The convergence curve of fitness value of WOA and its variants

Content courtesy of Springer Nature, terms of use apply. Rights reserved.

Table 4 Results of compact algorithm comparison

F	PCWOA			CBA			CSCA			MCSCA		
	Ave	Std	Rank	Ave	Std	Rank	Ave	Std	Rank	Ave	Std	Rank
F1	−1.35E+03	1.67E+01	**1**	5.87E+04	8.33E+02	3	1.32E+05	1.55E+04	4	1.60E+04	2.89E+03	2
F2	5.99E+07	2.33E+07	**1**	4.68E+09	1.98E+08	4	2.39E+09	4.36E+08	3	2.48E+08	6.83E+07	2
F3	3.44E+10	1.79E+10	**1**	4.23E+21	1.41E+21	4	6.62E+18	1.58E+19	3	4.33E+13	1.35E+14	2
F4	7.26E+04	2.36E+04	3	6.84E+04	2.70E+02	2	2.79E+05	9.95E+04	4	5.25E+04	4.69E+03	**1**
F5	−8.23E+02	3.49E+01	**1**	5.43E+04	1.61E+03	4	4.94E+04	9.71E+03	3	4.21E+03	1.40E+03	2
F6	−7.45E+02	5.58E+01	**1**	1.86E+04	4.42E+02	3	3.16E+04	4.03E+03	4	2.07E+03	1.13E+03	2
F7	2.20E+03	7.75E+05	**1**	2.81E+08	2.54E+07	4	1.36E+06	1.86E+06	3	1.35E+03	3.11E+03	2
F8	−6.79E+02	6.20E−02	2	−6.79E+02	6.00E−02	**1**	−6.79E+02	6.90E−02	3	−6.78E+02	3.10E−01	4
F9	−5.65E+02	2.66E+00	**1**	−5.42E+02	1.25E+00	4	−5.55E+02	1.52E+00	2	−5.45E+02	9.11E+00	3
F10	−2.96E+02	6.82E+01	**1**	1.16E+04	2.46+02	4	1.73E+04	2.35E+03	3	2.27E+03	4.62E+02	2
F11	1.37E+02	1.05E+02	2	6.14E+02	3.15E+01	3	1.17E+03	1.55E+02	4	1.07E+02	6.74E+01	**1**
F12	2.66E+02	1.19E+02	2	6.80E+02	1.76E+01	3	1.57E+03	1.97E+02	4	2.32E+02	8.56E+01	**1**
F13	3.75E+02	1.08E+02	2	9.37E+02	4.10E+01	3	1.68E+03	1.95E+02	4	3.28E+02	5.40E+01	**1**
F14	5.71E+03	7.89E+02	**1**	6.60E+03	2.84E+02	2	9.24E+03	3.93E+02	4	6.96E+03	5.52E+02	3
F15	5.90E+03	7.51E+02	**1**	6.07E+03	4.60E+02	2	9.32E+03	3.52E+02	4	7.51E+03	4.20E+02	3
F16	2.02E+02	4.10E−01	**1**	2.03E+02	6.50E−01	3	2.04E+02	6.10E−01	4	2.03E+02	3.60E−01	2
F17	9.39E+02	1.31E+02	**1**	1.21E+03	1.58E+01	3	4.67E+03	4.06E+02	4	9.40E+02	1.06E+02	2
F18	1.00E+03	1.26E+02	2	1.30E+03	1.57E+01	3	4.79E+03	3.45E+02	4	9.99E+02	7.15E+01	**1**
F19	5.46E+02	1.68E+01	**1**	7.53E+05	4.87E+04	3	9.31E+06	3.62E+06	4	9.66E+03	3.52E+03	2
F20	6.14E+02	3.70E−01	**1**	6.15E+02	0.00E+00	3	6.15E+02	6.51E−13	2	6.15E+02	2.40E−01	4
F21	1.13E+03	5.84E+01	**1**	3.21E+03	2.35E+01	3	1.12E+04	1.14E+03	4	2.53E+03	2.55E+02	2
F22	7.27E+03	7.18E+02	**1**	1.11E+04	2.22E+02	3	1.13E+04	2.46E+02	4	8.66E+03	6.39E+02	2
F23	7.26E+03	7.71E+02	**1**	9.69E+03	4.36E+02	3	1.04E+04	3.71E+02	4	9.11E+03	3.83E+02	2
F24	1.30E+03	5.69E+00	2	1.81E+03	2.38E+01	4	1.40E+03	3.07E+01	3	1.30E+03	5.39E+00	**1**
F25	1.39E+03	5.28E+00	**1**	1.56E+03	5.68E+00	4	1.49E+03	1.66E+01	3	1.40E+03	3.86E+00	2
F26	1.46E+03	8.88E+01	**1**	2.30E+03	8.16E+01	4	1.63E+03	9.56E+00	3	1.55E+03	7.54E+01	2
F27	2.52E+03	5.48E+01	**1**	4.39E+03	5.37E+01	4	2.90E+03	4.65E+01	3	2.65E+03	3.78E+01	2
F28	5.91E+03	9.22E+02	2	1.07E+04	2.21E+02	3	1.51E+04	1.62E+03	4	5.66E+03	5.57E+02	**1**
Mean	**1.14**			**3.18**			**3.5**			**1.82**		
Rank	**1**			**3**			**4**			**2**		

Bold parts indicate that the algorithm ranks the first in the average value obtained after 30 independent runs

Content courtesy of Springer Nature, terms of use apply. Rights reserved.

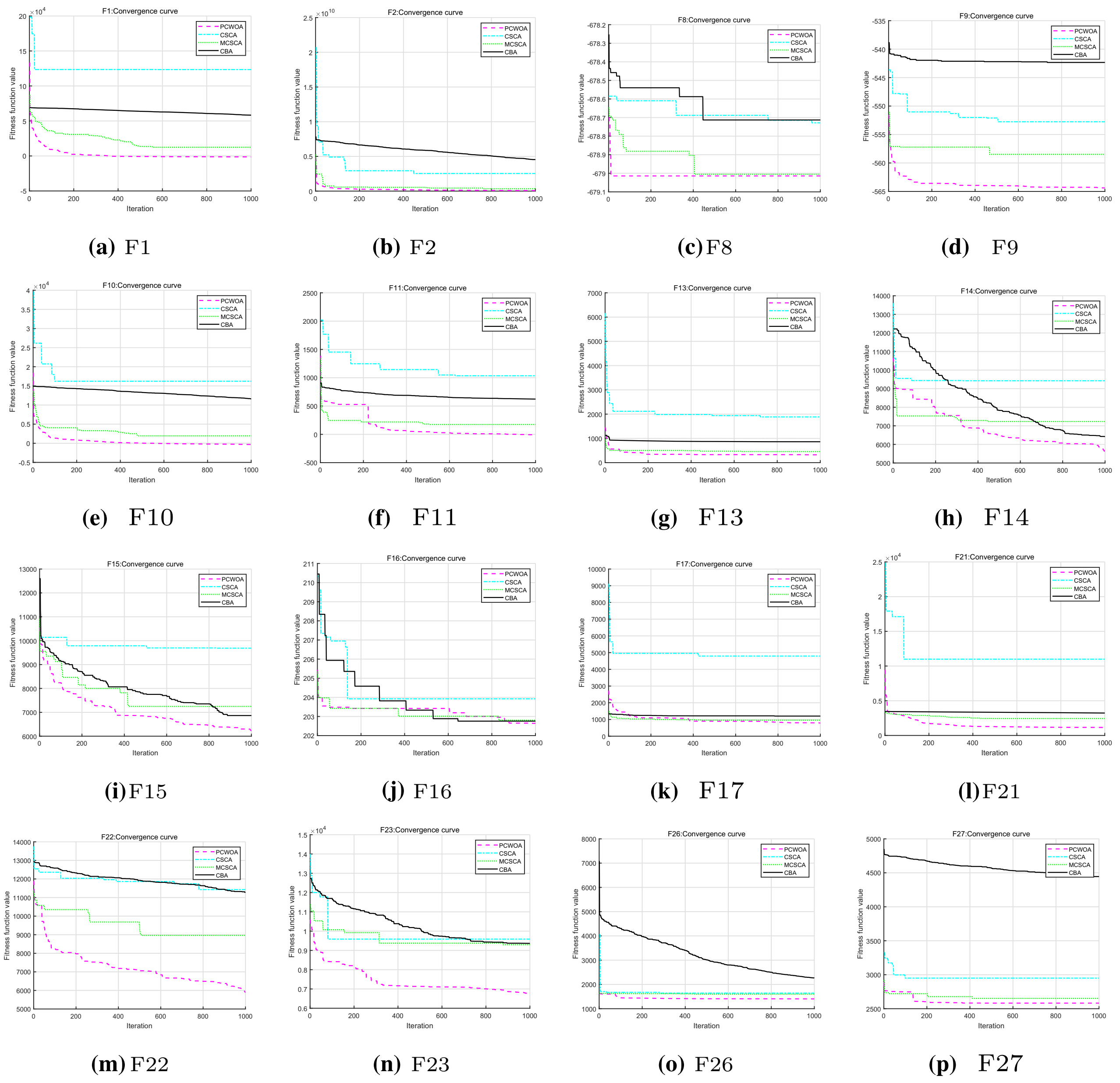


Fig. 4 The convergence curve of fitness value of compact algorithms

parallel techniques are superior to the initial results, and the quality of the parallel techniques directly determine the performance of the algorithm. The parallel techniques in this paper are obviously better than that of MCSCA.

4.4 Compared with popular algorithms

This section compares PCWOA with the other four popular MHOAs proposed in recent years. Mean and best values are used to evaluate their performance. The parameters of the 5 algorithms are listed in Table 2. In Table 5, the *Ave* indicates the mean value, the *Best* represents the optimal value.

Compared with SCA, PCWOA performs poorly on *F*4, *F*7, *F*12, *F*17, and *F*28 in average value. However, the best value of *F*7, *F*17 surpasses the SCA. The PCWOA achieves better results than ALO in mean value and best value except for *F*11, *F*15, and *F*22. Compared with AOA, PCWOA achieves good promising results on most test functions except *F*4 and *F*14. Among the 4 popular algorithms, SSA performs best. It is superior to PCWOA in *F*7, *F*9, *F*12, *F*13, *F*14, *F*17, *F*22, and *F*28, but not competitive in other functions. In *F*8, *F*16, and *F*20, all algorithms find

Content courtesy of Springer Nature, terms of use apply. Rights reserved.

Table 5 Comparison of the PCWOA with the popular algorithms

F	PCWOA		SCA		ALO		AOA		SSA	
	Ave	Best	Ave	Best	Ave	Best	Ave	Best	Ave	Best
F1	−1.35E+03	−1.38E+03	1.37E+04	7.20E+03	2.33E+04	8.75E+03	6.50E+04	4.84E+04	−1.40E+03	−1.40E+03
F2	7.49E+07	3.35E+07	2.13E+08	5.03E+07	2.91E+08	1.68E+08	1.59E+09	7.09E+08	1.29E+08	3.69E+07
F3	3.27E+10	6.14E+09	6.71E+10	2.56E+10	3.32E+13	3.56E+10	1.97E+20	3.82E+14	6.86E+10	4.50E+10
F4	7.09E+04	3.42E+04	**4.56E+04**	3.00E+04	1.58E+05	1.12E+05	**6.80E+04**	6.11E+04	5.52E+04	3.07E+04
F5	−8.45E+02	−8.87E+02	2.07E+03	1.07E+03	5.43E+03	1.25E+03	2.88E+04	8.97E+03	1.25E+03	−2.25E+02
F6	−7.46E+02	−8.60E+02	2.03E+02	−4.68E+02	1.15E+03	1.94E+01	1.60E+04	8.73E+03	−4.96E+02	−6.87E+02
F7	2.63E+00	−6.39E+02	**−5.84E+02**	−6.61E+02	6.74E+02	−6.67E+02	1.37E+07	5.33E+01	**−6.65E+02**	−7.17E+02
F8	−6.79E+02	−6.79E+02	−6.79E+02	−6.79E+02	−6.79E+02	−6.79E+02	−6.79E+02	−6.79E+02	−6.79E+02	−6.79E+02
F9	−5.64E+02	−5.71E+02	−5.58E+02	−5.62E+02	−5.62E+02	−5.66E+02	−5.56E+02	−5.62E+02	**−5.72E+02**	−5.80E+02
F10	−3.14E+02	−4.16E+02	1.62E+03	9.41E+02	2.27E+03	1.61E+03	1.03E+04	6.18E+03	6.20E+02	1.96E+02
F11	1.25E+02	−1.86E+02	8.97E+02	−8.15E+01	**2.78E+01**	−8.47E+01	5.43E+02	1.92E+02	−2.25E+02	−2.97E+02
F12	2.47E+02	6.70E+01	**1.16E+02**	3.93E+01	2.55E+02	1.16E+02	6.57E+02	3.90E+02	**1.10E+02**	1.15E+01
F13	3.01E+02	1.76E+02	3.32E+02	1.87E+02	3.78E+02	2.34E+02	7.48E+02	5.50E+02	**2.63E+02**	1.90E+02
F14	5.82E+03	4.52E+03	7.30E+03	6.91E+03	5.83E+03	5.19E+03	**4.55E+03**	3.47E+03	**5.48E+03**	4.64E+03
F15	5.83E+03	4.27E+03	7.87E+03	6.91E+03	**5.77E+03**	4.22E+03	7.70E+03	6.51E+03	6.15E+03	5.39E+03
F16	2.02E+02	2.01E+02	2.02E+02	2.02E+02	2.02E+02	2.01E+02	2.02E+02	2.01E+02	2.02E+02	2.01E+02
F17	9.35E+02	7.22E+02	**8.43E+02**	7.37E+02	1.19E+03	9.50E+02	1.03E+03	9.25E+02	**8.68E+02**	6.03E+02
F18	9.65E+02	7.16E+02	9.53E+02	8.69E+02	1.16E+03	9.50E+02	1.26E+03	1.23E+03	1.01E+03	7.58E+02
F19	5.53E+02	5.24E+02	8.35E+03	2.38E+03	3.03E+04	5.57E+03	8.43E+05	5.76E+05	4.56E+03	5.44E+02
F20	6.15E+02	6.14+02	6.15E+02	6.14E+02	6.15E+02	6.15E+02	6.15E+02	6.15E+02	6.15E+02	6.15E+02
F21	1.12E+03	1.03E+03	2.76E+03	2.50E+03	2.99E+03	2.68E+02	3.39E+03	3.29E+03	2.60E+03	2.28E+03
F22	7.34E+03	6.15E+03	8.74E+03	7.75E+03	**7.17E+03**	6.40E+03	7.49E+03	6.34E+03	**7.18E+03**	6.08E+03
F23	7.03E+03	5.03E+03	9.21E+03	8.44E+03	7.66E+03	6.76E+03	9.64E+03	7.86E+03	7.73E+03	6.00E+03
F24	1.30E+03	1.28E+03	1.32E+03	1.31E+03	1.33E+03	1.32E+03	1.57E+03	1.38E+03	1.31E+03	1.29E+03
F25	1.39E+03	1.37E+03	1.43E+03	1.43E+03	1.47E+03	1.44E+03	1.54E+03	1.48E+03	1.42E+03	1.40E+03
F26	1.47E+03	1.40E+03	1.53E+03	1.41E+03	1.59E+03	1.59E+03	1.61E+03	1.46E+03	1.57E+03	1.41E+03
F27	2.53E+03	2.38E+03	2.71E+03	2.63E+03	2.66E+03	2.49E+03	3.10E+03	2.75E+03	2.55E+03	2.43E+03
F28	5.96E+03	4.40E+03	**4.30E+03**	3.86E+03	6.55E+03	5.32E+03	7.96E+03	7.13E+03	**4.01E+03**	3.23E+03

Bold parts demonstrate that four algorithms perform better than PCWOA individually

Content courtesy of Springer Nature, terms of use apply. Rights reserved.

the same solution. Compared with SCA, the winning rate of PCWOA is 23/28(82%); compared with ALO, the winning rate of PCWOA is 25/28(89%); compared with AOA, the winning rate of PCWOA is 26/28(93%); compared with SSA, the winning rate of PCWOA is 20/28(71%). In general, the performance of PCWOA is the best.

5 DV-Hop localization algorithm based on PCWOA

5.1 DV-Hop localization algorithm

DV-Hop localization algorithm [43] can be divided into the following three steps:

Step 1: The anchor node broadcasts its location information to the neighbor nodes to obtain the minimum hop-size between the unknown and anchor nodes.

Step 2: According to the location information and hop-sizes between each anchor node recorded in step 1, the average actual distance per hop-size is estimated by Eq. (14):

$$HopSize_i = \frac{\sum_{j=1, j\neq i}^{n} \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}}{\sum_{j\neq i} h_{ij}}, \tag{14}$$

where $HopSize_i$ represents the average per hop-size between anchor node i and others, n indicates the number of anchor nodes and $n \geq 3$, (x_i, y_i) and (x_j, y_j) are the coordinates of anchor node i and j respectively, h_{ij} represents the hop-size between anchor node i and j.

Then, the distance between the unknown node and the anchor node is calculated by Eq. (15).

$$d_{iu} = HopSize_i \times h_{iu}, \tag{15}$$

where u represents the unknown node, h_{iu} indicates the distance between anchor node i and unknown node u.

Step 3: The unknown node uses d_{iu} recorded in step 2 and uses the trilateral measurement method or maximum likelihood estimation method to calculate its coordinates.

5.2 Apply PCWOA in DV-Hop localization algorithm

In the traditional DV-Hop algorithm, the LSM is used to calculate the location of unknown nodes. The reason for the low localization accuracy of traditional DV-Hop is the calculation error of LSM. PCWOA is employed to replace LSM to eliminate error. The Eq. (16) calculates the error between anchor nodes and unknown nodes:

$$error_u = \sum_{i=1}^{n} \sqrt{(x_i - x_u)^2 + (y_i - y_u)^2} - d_{iu}, \tag{16}$$

where $error_u$ indicates the sum of distance errors of unknown node u to all anchor nodes. (x_u, y_u) represents the coordinate of unknown node u. Other parameter settings are shown above. The purpose of using PCWOA is to minimize the $error_u$. Therefore, we utilize the following fitness function to achieve this object:

$$Fun(x, y) = \sum_{i=1}^{n} \sqrt{(x - x_i)^2 + (y - y_i)^2} - d_i, \tag{17}$$

where (x, y) and (x_i, y_i) represent the coordinate of unknown nodes and anchor node i. The d_i indicates the estimated distance between unknown nodes and anchor node i. All individuals of PCWOA represent the position of sensor nodes, and then we use the fitness function to find the optimal position.

6 Simulation research

In this section, we will conduct simulation research. This paper combines PCWOA with DV-Hop to compensate for the localization error of the original DV-Hop for WSN. We also apply PCWOA, parallel WOA (PWOA) [33], WOA [16], and PSO [34] to DV-Hop and compare them with original DV-Hop [43] in WSN. Then, the average localization error (ALE) is defined in Eq. (18) to evaluate their performance.

$$ALE = \frac{\sum_{u=1}^{m} \sqrt{(X - x_u)^2 + (Y - y_u)^2}}{m \times r}. \tag{18}$$

where m represents the number of unknown nodes, (X, Y) and (x_u, y_u) indicate the actual and estimated coordinates of unknown nodes, respectively. r represents the communication radius of the node.

All algorithms are executed 30 times under the same environment to achieve a fair comparison. The max iteration is set as 100, and other parameters are shown in Table 2. According to the research on the existing literatures [33, 37, 40, 42, 43], the nodes in WSN are randomly distributed in a 2-dimensional fixed space which is set charged with 100 m × 100 m in Fig. 5. The total number of nodes is limited within the interval [100,400], and the variation range of anchor nodes is [5,40], the node communication radius is set to change in the interval of [20,40]. It is worth noting that the scope of the simulation environment we set should meet the communication requirements of the least anchor nodes. According to the literature [43], the location of unknown nodes in two-dimensional space needs at least three anchor nodes to be determined.

Content courtesy of Springer Nature, terms of use apply. Rights reserved.

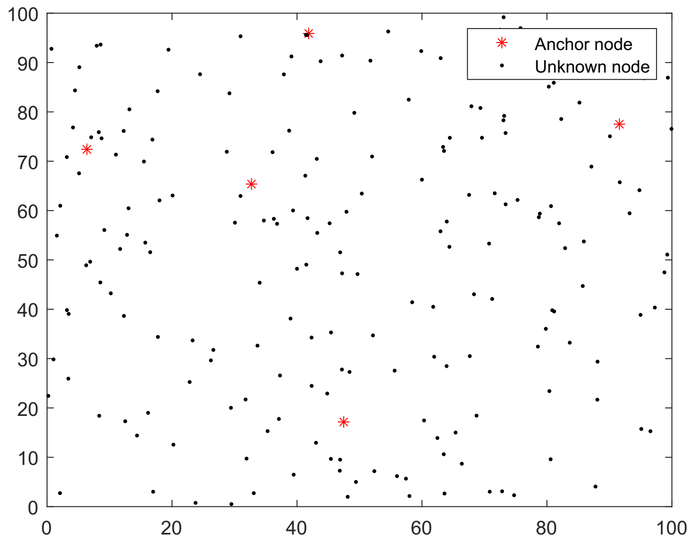


Fig. 5 Initial distribution of nodes

The minimum number of anchor nodes set in this paper is 5, and the minimum communication radius of nodes is 20. Thus, the simulation area set in this paper exactly meets the minimum communication requirements.

6.1 Influences of the number of anchor nodes

To ensure the generalization of the experiment, we set the total number of nodes to 200 and randomly deployed them in an area of 100 m^2, and the node communication radius is set to 20 m. In this experiment, we study the influence of the number of anchor nodes on the node error. Thus, we gradually increase the number of anchor nodes from 5 to 40 with an interval of 5.

From Table 6, the localization error gradually decreases with the increasing number of anchor nodes. The application of the MHOA to DV-Hop significantly improves the performance. WOA is an improved algorithm compared with PSO. It can be seen from Table 6 that the performance of WOA is markedly higher than that of PSO. PCWOA has competitive performance than the original WOA, except for the case of 10 anchor nodes. When the number of anchor nodes is 5, the performance of PWOA overtakes that of PCWOA. Part of the reason is that adding compact technique to the algorithm impairs the algorithm's ability to jump out of local optimum. Another part is that in the proposed parallel techniques, discarding a part of the poor individuals may lose the diversity of the population and perform poorly when dealing with a small number of anchor nodes. In addition, from Table 6, the locating errors decreases with the increase of anchor nodes, but the decreasing trend is mild. Although increasing anchor nodes can improve locating accuracy, employing too many anchor nodes is costly and therefore unnecessary when using DV-Hop or an improved algorithm.

Table 6 Comparison of experimental results of different number of anchor nodes

Anchor node	DV-Hop	PSO	WOA	PWOA	PCWOA
5	0.4988	0.4397	0.4031	**0.3812**	0.3956
10	0.3939	0.2804	0.2644	**0.2584**	0.2830
15	0.3577	0.2540	0.2550	0.2539	**0.2376**
20	0.3402	0.2261	0.2228	0.2215	**0.2124**
25	0.3300	0.2003	0.2005	0.2003	**0.1966**
30	0.3425	0.1979	0.1989	0.1965	**0.1907**
35	0.3137	0.1913	0.1918	0.1910	**0.1862**
40	0.3027	0.1839	0.1851	0.1838	**0.1799**

Bold parts indicate the minimum errors

6.2 Influences of communication radius

To simulate the layout environment of WSN realistically, we study the influences of communication radius. In this experiment, the total number of nodes is set to 200, and the number of anchor nodes is fixed to 20. So, the number of unknown nodes is 180. We change the size of the communication radius and control its range in [15,40].

According to Table 7, it can be seen that the localization error calculated by PCWOA is superior to that of other algorithms. All algorithms achieve the minimum error when the communication radius is 30. When the total number of network nodes and the number of anchor nodes remain unchanged, the larger the communication radius, the higher the connectivity of the network. However, if the communication radius increases while the simulation area remains unchanged, one or several nodes may be omitted in calculating the average hop-size of anchor nodes.

Content courtesy of Springer Nature, terms of use apply. Rights reserved.

Table 7 Comparison of experimental results of different communication radius of nodes

Communication radius	DV-Hop	PSO	WOA	PWOA	PCWOA
20	0.3402	0.2663	0.2235	0.2214	**0.2124**
25	0.3330	0.2066	0.2059	0.2057	**0.2023**
30	0.3095	0.1939	0.1916	0.1906	**0.1886**
35	0.3158	0.2036	0.2044	0.2034	**0.2027**
40	0.3125	0.1940	0.1944	0.1939	**0.1921**

Bold parts indicate the minimum errors

6.3 Influences of the total number of nodes

To research the influences of the total number of nodes and test the comprehensive performance of PCWOA. In this experiment, we fixed the number of anchor nodes and communication radius to 20 and 30, respectively. The performance of PCWOA is evaluated by changing the total number of nodes, and its range is controlled between 100 and 400.

It can be seen from Table 8 that the performance of PCWOA is robust in the case of different numbers of nodes. Compared with the original DV-Hop algorithm and other MHOAs, the performance of PCWOA is significantly improved. From Table 8, all algorithms reach the minimum error when the total number of nodes is 200.

When the communication radius is 30, and the total number of nodes is 200, PCWOA performs best in $100\,\text{m} \times 100\,\text{m}$ fixed space. In Table 6, PCWOA performs best when the number of anchor nodes is 30 and increases by 0.1518 compared with DV-Hop. PCWOA serves best when the communication radius is 25 and increases by 0.1307 compared with DV-Hop in Table 7. In Table 8, PCWOA performs best when the total nodes are 100 and increases by 0.1614 compared with DV-Hop. PCWOA performs best when the total number of nodes is 100, the anchor node is 30, and the communication radius is 25. According to the results listed in Tables 6, 7 and 8, we can sum up cautiously that the PCWOA has competitive performance than the original algorithm.

Table 8 Comparison of experimental results of total number of nodes

Total nodes	DV-Hop	PSO	WOA	PWOA	PCWOA
100	0.3825	0.2285	0.2272	0.2233	**0.2211**
150	0.3502	0.1962	0.1969	0.1953	**0.1936**
200	0.3095	0.1947	0.1912	0.1906	**0.1890**
250	0.3413	0.2209	0.2180	0.2171	**0.2145**
300	0.3193	0.2178	0.2156	0.2145	**0.2144**
350	0.3405	0.2268	0.2258	0.2264	**0.2244**
400	0.3318	0.2267	0.2258	0.2203	**0.2199**

Bold parts indicate the minimum errors

7 Conclusions

This paper proposes a PCWOA and employs it to optimize DV-Hop localization algorithm for WSN. Firstly, WOA with a compact technique can effectively save the memory of the algorithm and reduce the consumption of sensor nodes. Still, it inevitably increases the probability of falling into a local optimum. Secondly, aiming at the defects of the compact algorithm, we propose novel parallel techniques to enhance the capacity of jumping out of a local optimum and improve the solution accuracy. Experiments have shown that the algorithm we proposed performs better, and the accuracy of the solution is strengthened. Finally, the improved algorithm is applied in node localization of WSN, which is verified through experiments that the improvements can significantly increase the localization accuracy.

Through the research, it is revealed that DV-Hop combined with PCWOA performs well on 2-dimensional enviorments. In the future, we will apply this algorithm to 3-dimensional environments and change statically distributed nodes to dynamic. Additionally, we will also conduct further simulation and field experiments by changing different simulation environments and the number of nodes to test the robustness of an improved localization algorithm.

Funding The authors did not receive support from any organization for the submitted work.

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended

Content courtesy of Springer Nature, terms of use apply. Rights reserved.

use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Bai, X., Wang, Z., Sheng, L., & Wang, Z. (2018). Reliable data fusion of hierarchical wireless sensor networks with asynchronous measurement for greenhouse monitoring. *IEEE Transactions on Control Systems Technology, 27*(3), 1036–1046.
2. Majumder, S., Aghayi, E., Noferesti, M., Memarzadeh-Tehran, H., Mondal, T., Pang, Z., & Deen, M. J. (2017). Smart homes for elderly healthcare-recent advances and research challenges. *Sensors, 17*(11), 2496.
3. Kodam, S., Bharathgoud, N., & Ramachandran, B. (2020). A review on smart wearable devices for soldier safety during battlefield using wsn technology. *Materials Today: Proceedings, 33*, 4578–4585.
4. Rajaram, V. & Kumaratharan, N. (2021). Multi-hop optimized routing algorithm and load balanced fuzzy clustering in wireless sensor networks. *Journal of Ambient Intelligence and Humanized Computing, 12*(3), 4281–4289.
5. Rajasekaran, T. & Anandamurugan, S. (2019). Challenges and applications of wireless sensor networks in smart farming-a survey. In *Advances in big data and cloud computing,* (pp. 353–361) Springer
6. Kandris, D., Nakas, C., Vomvas, D., & Koulouras, G. (2020). Applications of wireless sensor networks: an up-to-date survey. *Applied System Innovation, 3*(1), 14.
7. Farjamnia, G., Gasimov, Y., Kazimov, C., & Hashemi, M. (2020). A survey of dv-hop localization methods in wireless sensor networks. *Journal of Communication Engineering 9*(2)
8. Halder, S. & Ghosal, A. (2016). A survey on mobile anchor assisted localization techniques in wireless sensor networks. *Wireless Networks, 22*(7), 2317–2336.
9. Paul, A. K. & Sato, T. (2017). Localization in wireless sensor networks: A survey on algorithms, measurement techniques, applications and challenges. *Journal of Sensor and Actuator Networks, 6*(4), 24.
10. Kumar, S. & Lobiyal, D. (2017). Novel dv-hop localization algorithm for wireless sensor networks. *Telecommunication Systems, 64*(3), 509–524.
11. Nasir, H. J. A., Ku-Mahamud, K. R., & Kamioka, E. (2017). Ant colony optimization approaches in wireless sensor network: performance evaluation. *Journal of Computer Science, 13*(6), 153–164.
12. Shakshuki, E., Elkhail, A. A., Nemer, I., Adam, M., & Sheltami, T. (2019). Comparative study on range free localization algorithms. *Procedia Computer Science, 151*, 501–510.
13. Yang, J., Cai, Y., Tang, D., & Liu, Z. (2019). A novel centralized range-free static node localization algorithm with memetic algorithm and lévy flight. *Sensors, 19*(14), 3242.
14. Singh, P. R., Abd Elaziz, M., & Xiong, S. (2018). Modified spider monkey optimization based on nelder-mead method for global optimization. *Expert Systems with Applications, 110*, 264–289.
15. Hussain, K., Salleh, M. N. M., Cheng, S., & Shi, Y. (2019). Metaheuristic research: a comprehensive survey. *Artificial Intelligence Review, 52*(4), 2191–2233.
16. Mirjalili, S. & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software, 95,* 51–67.
17. Gharehchopogh, F. S. & Gholizadeh, H. (2019). A comprehensive survey: Whale optimization algorithm and its applications. *Swarm and Evolutionary Computation, 48*, 1–24.
18. Kaur, G. & Arora, S. (2018). Chaotic whale optimization algorithm. *Journal of Computational Design and Engineering, 5*(3), 275–284.
19. Oliva, D., Abd El Aziz, M., & Hassanien, A. E. (2017). Parameter estimation of photovoltaic cells using an improved chaotic whale optimization algorithm. *Applied Energy, 200,* 141–154.
20. Tubishat, M., Abushariah, M. A., Idris, N., & Aljarah, I. (2019). Improved whale optimization algorithm for feature selection in arabic sentiment analysis. *Applied Intelligence, 49*(5), 1688–1707.
21. Mostafa Bozorgi, S. & Yazdani, S. (2019). Iwoa: An improved whale optimization algorithm for optimization problems. *Journal of Computational Design and Engineering, 6*(3), 243–259.
22. Hussien, A. G., Hassanien, A. E., Houssein, E. H., Amin, M., & Azar, A. T. (2020). New binary whale optimization algorithm for discrete optimization problems. *Engineering Optimization, 52*(6), 945–959.
23. Reddy, K. S., Panwar, L., Panigrahi, B., & Kumar, R. (2019). Binary whale optimization algorithm: a new metaheuristic approach for profit-based unit commitment problems in competitive electricity markets. *Engineering Optimization, 51*(3), 369–389.
24. Abd El Aziz, M., Ewees, A. A., & Hassanien, A. E. (2017). Whale optimization algorithm and moth-flame optimization for multilevel thresholding image segmentation. *Expert Systems with Applications, 83,* 242–256.
25. Mafarja, M. M. & Mirjalili, S. (2017). Hybrid whale optimization algorithm with simulated annealing for feature selection. *Neurocomputing, 260*, 302–312.
26. Wang, J., Du, P., Niu, T., & Yang, W. (2017). A novel hybrid system based on a new proposed algorithm-multi-objective whale optimization algorithm for wind speed forecasting. *Applied Energy, 208,* 344–360.
27. Abd El Aziz, M., Ewees, A. A., & Hassanien, A. E. (2018). Multi-objective whale optimization algorithm for content-based image retrieval. *Multimedia Tools and Applications, 77*(19), 26135–26172.
28. Lang, F., Su, J., Ye, Z., Shi, X., & Chen, F. (2019). A wireless sensor network location algorithm based on whale algorithm. In *2019 10th IEEE international conference on intelligent data acquisition and advanced computing systems: technology and applications (IDAACS),* (vol. 1, pp. 106–110). IEEE
29. Daely, P. T. & Shin, S. Y. (2016). Range based wireless node localization using dragonfly algorithm. In *2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN),* (pp. 1012–1015). IEEE
30. Miloud, M., Abdellatif, R., & Lorenz, P. (2019). Moth flame optimization algorithm range-based for node localization challenge in decentralized wireless sensor network. *International Journal of Distributed Systems and Technologies (IJDST), 10*(1), 82–109.
31. Shakila, R. & Paramasivan, B. (2021). An improved range based localization using whale optimization algorithm in underwater wireless sensor network. *Journal of Ambient Intelligence and Humanized Computing, 12*(6), 6479–6489.
32. Huang, M. & Yu, B. (2020). Range-based positioning with self-adapting fireworks algorithm for wireless sensor networks. *Mathematical Problems in Engineering 2020*
33. Chai, Q.-W., Chu, S.-C., Pan, J.-S., Hu, P., & Zheng, W.-M. (2020). A parallel woa with two communication strategies applied in dv-hop localization method. *EURASIP Journal on Wireless Communications and Networking, 2020*(1), 1–10.
34. Chen, Y., Li, X., Ding, Y., Xu, J., & Liu, Z. (2018). An improved dv-hop localization algorithm for wireless sensor networks. In *2018 13th IEEE conference on industrial electronics and applications (ICIEA),* (pp. 1831–1836). IEEE

Content courtesy of Springer Nature, terms of use apply. Rights reserved.

35. Tomic, S. & Mezei, I. (2016). Improvements of dv-hop localization algorithm for wireless sensor networks. *Telecommunication Systems, 61*(1), 93–106.
36. Kanwar, V. & Kumar, A. (2021). Dv-hop localization methods for displaced sensor nodes in wireless sensor network using pso. *Wireless Networks, 27*(1), 91–102.
37. Abd El Ghafour, M. G., Kamel, S. H., & Abouelseoud, Y. (2021). Improved dv-hop based on squirrel search algorithm for localization in wireless sensor networks. *Wireless Networks, 27*(4), 2743–2759.
38. Cui, L., Xu, C., Li, G., Ming, Z., Feng, Y., & Lu, N. (2018). A high accurate localization algorithm with dv-hop and differential evolution for wireless sensor network. *Applied Soft Computing, 68*, 39–52.
39. Ouyang, A., Lu, Y., Liu, Y., Wu, M., & Peng, X. (2021). An improved adaptive genetic algorithm based on dv-hop for locating nodes in wireless sensor networks. *Neurocomputing*
40. Chen, X. & Zhang, B. (2012). Improved dv-hop node localization algorithm in wireless sensor networks. *International Journal of Distributed Sensor Networks, 8*(8), 213980.
41. Cui, Z., Sun, B., Wang, G., Xue, Y., & Chen, J. (2017). A novel oriented cuckoo search algorithm to improve dv-hop performance for cyber-physical systems. *Journal of Parallel and Distributed Computing, 103*, 42–52.
42. Li, J., Gao, M., Pan, J.-S., & Chu, S.-C. (2021). A parallel compact cat swarm optimization and its application in dv-hop node localization for wireless sensor network. *Wireless Networks, 27*(3), 2081–2101.
43. Niculescu, D. & Nath, B. (2001) Ad hoc positioning system (aps). In *GLOBECOM'01. IEEE global telecommunications conference (Cat. No. 01CH37270),* (vol. 5, pp. 2926–2931) IEEE
44. Neri, F., Mininno, E., & Iacca, G. (2013). Compact particle swarm optimization. *Information Sciences, 239*, 96–121.
45. Harik, G. R., Lobo, F. G., & Goldberg, D. E. (1999). The compact genetic algorithm. *IEEE Transactions on Evolutionary Computation, 3*(4), 287–297.
46. Mininno, E., Neri, F., Cupertino, F., & Naso, D. (2010). Compact differential evolution. *IEEE Transactions on Evolutionary Computation, 15*(1), 32–54.
47. 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.
48. Zhu, M., Chu, S. -C., Yang, Q., Li, W., & Pan, J. -S. (2021). Compact sine cosine algorithm with multigroup and multistrategy for dispatching system of public transit vehicles. *Journal of Advanced Transportation 2021*
49. Chu, S.-C., Roddick, J. F., & Pan, J.-S. (2005). A parallel particle swarm optimization algorithm with communication strategies. *Journal of Information Science and Engineering, 21*(4), 9.
50. Han, K., Huang, T., & Yin, L. (2021). Quantum parallel multilayer monte carlo optimization algorithm for controller parameters optimization of doubly-fed induction generator-based wind turbines. *Applied Soft Computing, 112*, 107813.
51. Rizk-Allah, R. M., El-Sehiemy, R. A., & Wang, G.-G. (2018). A novel parallel hurricane optimization algorithm for secure emission/economic load dispatch solution. *Applied Soft Computing, 63*, 206–222.
52. Liu, Z., Li, Z., Zhu, P., & Chen, W. (2018). A parallel boundary search particle swarm optimization algorithm for constrained optimization problems. *Structural and Multidisciplinary Optimization, 58*(4), 1505–1522.
53. Jamshidi, V., Nekoukar, V., & Refan, M. H. (2021). Real time uav path planning by parallel grey wolf optimization with align coefficient on can bus. *Cluster Computing,* (pp. 1–15)
54. Wang, R. -B., Wang, W. -F., Xu, L., Pan, J. -S., Chu, S. -C. (2021). An adaptive parallel arithmetic optimization algorithm for robot path planning. *Journal of Advanced Transportation 2021*
55. Liang, J., Qu, B., Suganthan, P., & 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.

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

Content courtesy of Springer Nature, terms of use apply. Rights reserved.

Terms and Conditions

Springer Nature journal content, brought to you courtesy of Springer Nature Customer Service Center GmbH ("Springer Nature").
Springer Nature supports a reasonable amount of sharing of research papers by authors, subscribers and authorised users ("Users"), for small-scale personal, non-commercial use provided that all copyright, trade and service marks and other proprietary notices are maintained. By accessing, sharing, receiving or otherwise using the Springer Nature journal content you agree to these terms of use ("Terms"). For these purposes, Springer Nature considers academic use (by researchers and students) to be non-commercial.
These Terms are supplementary and will apply in addition to any applicable website terms and conditions, a relevant site licence or a personal subscription. These Terms will prevail over any conflict or ambiguity with regards to the relevant terms, a site licence or a personal subscription (to the extent of the conflict or ambiguity only). For Creative Commons-licensed articles, the terms of the Creative Commons license used will apply.
We collect and use personal data to provide access to the Springer Nature journal content. We may also use these personal data internally within ResearchGate and Springer Nature and as agreed share it, in an anonymised way, for purposes of tracking, analysis and reporting. We will not otherwise disclose your personal data outside the ResearchGate or the Springer Nature group of companies unless we have your permission as detailed in the Privacy Policy.
While Users may use the Springer Nature journal content for small scale, personal non-commercial use, it is important to note that Users may not:

1. use such content for the purpose of providing other users with access on a regular or large scale basis or as a means to circumvent access control;
2. use such content where to do so would be considered a criminal or statutory offence in any jurisdiction, or gives rise to civil liability, or is otherwise unlawful;
3. falsely or misleadingly imply or suggest endorsement, approval , sponsorship, or association unless explicitly agreed to by Springer Nature in writing;
4. use bots or other automated methods to access the content or redirect messages
5. override any security feature or exclusionary protocol; or
6. share the content in order to create substitute for Springer Nature products or services or a systematic database of Springer Nature journal content.

In line with the restriction against commercial use, Springer Nature does not permit the creation of a product or service that creates revenue, royalties, rent or income from our content or its inclusion as part of a paid for service or for other commercial gain. Springer Nature journal content cannot be used for inter-library loans and librarians may not upload Springer Nature journal content on a large scale into their, or any other, institutional repository.
These terms of use are reviewed regularly and may be amended at any time. Springer Nature is not obligated to publish any information or content on this website and may remove it or features or functionality at our sole discretion, at any time with or without notice. Springer Nature may revoke this licence to you at any time and remove access to any copies of the Springer Nature journal content which have been saved.
To the fullest extent permitted by law, Springer Nature makes no warranties, representations or guarantees to Users, either express or implied with respect to the Springer nature journal content and all parties disclaim and waive any implied warranties or warranties imposed by law, including merchantability or fitness for any particular purpose.
Please note that these rights do not automatically extend to content, data or other material published by Springer Nature that may be licensed from third parties.
If you would like to use or distribute our Springer Nature journal content to a wider audience or on a regular basis or in any other manner not expressly permitted by these Terms, please contact Springer Nature at

onlineservice@springernature.com