COMPLEX SYSTEM MODELING AND SIMULATION
ISSN 2096-9929 02/06 pp 288–306
Volume 2, Number 4, December 2022
DOI: 10.23919/CSMS.2022.0018

Dual-Stage Hybrid Learning Particle Swarm Optimization Algorithm for Global Optimization Problems

Wei Li, Yangtao Chen, Qian Cai*, Cancan Wang, Ying Huang, and Soroosh Mahmoodi

Abstract: Particle swarm optimization (PSO) is a type of swarm intelligence algorithm that is frequently used to resolve specific global optimization problems due to its rapid convergence and ease of operation. However, PSO still has certain deficiencies, such as a poor trade-off between exploration and exploitation and premature convergence. Hence, this paper proposes a dual-stage hybrid learning particle swarm optimization (DHLPSO). In the algorithm, the iterative process is partitioned into two stages. The learning strategy used at each stage emphasizes exploration and exploitation, respectively. In the first stage, to increase population variety, a Manhattan distance based learning strategy is proposed. In this strategy, each particle chooses the furthest Manhattan distance particle and a better particle for learning. In the second stage, an excellent example learning strategy is adopted to perform local optimization operations on the population, in which each particle learns from the global optimal particle and a better particle. Utilizing the Gaussian mutation strategy, the algorithm's searchability in particular multimodal functions is significantly enhanced. On benchmark functions from CEC 2013, DHLPSO is evaluated alongside other PSO variants already in existence. The comparison results clearly demonstrate that, compared to other cutting-edge PSO variations, DHLPSO implements highly competitive performance in handling global optimization problems.

Key words: particle swarm optimization; Manhattan distance; example learning; gaussian mutation; dual-stage; global optimization problem

1 Introduction

As science and industrial engineering have progressed in tandem, realistic complex optimization problems have increased, and developing efficient algorithms to address these optimization problems has become urgent. Evolution algorithms (EAs), inspired by biological evolution or foraging modes in nature, realize the purpose of obtaining a relatively optimal solution in the feasible domain[1]. Therefore, EAs, for instance, particle swarm optimization (PSO)[2], genetic algorithm (GA)[3, 4], artificial bee colony (ABC)[5], differential evolution (DE)[6, 7], and firefly algorithm (FA)[8], due to their robustness and versatility, are commonly used by researchers to solve certain complicated global optimization issues[9]. Among them, PSO is an ordinary and effective algorithm that learns from the optimal value of the current search through a flock of particles to turn up the optimal solution. Kennedy and Eberhart[10] introduced the initial version of PSO in 1995, which is categorized as swarm intelligence and mimics the foraging behavior of birds[11, 12]. PSO is an iterative, population-based

• Wei Li, Yangtao Chen, Qian Cai, and Cancan Wang are with the School of Information Engineering, Jiangxi University of Science and Technology, Ganzhou 341000, China. E-mail: liwei@jxust.edu.cn; chenyangtao@jxust.edu.cn; caiqian@jxust.edu.cn; wangcancan@mail.jxust.edu.cn.
• Ying Huang is with the School of Mathematical and Computer Science, Gannan Normal University, Ganzhou 341000, China. E-mail: nhwshy@whu.edu.cn.
• Soroosh Mahmoodi is with the Soroosh Khorshid Iranian Co., Abyek Industrial Zone, Qazvin 999067, Iran. E-mail: soroosh.mahmoodi@yahoo.com.
* To whom correspondence should be addressed.
Manuscript received: 2022-07-22; revised: 2022-08-17; accepted: 2022-09-07

learning algorithm with some traits in common with other evolutionary algorithms. However, PSO seeks the global optima of space by learning the flight of each particle in the area and revising its flight trajectory based on the individual historical optima and global historical optima of its neighbors instead of selecting, crossing, and mutating particles for the best solution and more genetic manipulation. For resolving numerous optimization problems, the PSO algorithm is easy to apply and efficient, but researchers have discovered that classical PSO has weak search capabilities and particles might prematurely converge around local optima. In this scenario, a controversial issue in recent years has been how to efficiently tackle these issues and further increase the power of the algorithm. The following categories, in particular, have been suggested by academics as ways to increase PSO performance.

(1) Parameter adaptive control

Numerous studies have shown that an appropriate parameter adaptation strategy will help improve the optimization performance of PSO. There are mainly three parameters ω, c_1, and c_2, and most of the parameter adaptation improvements are based on ω[13]. For a large ω, the convergence rate of the population can be improved, which is biased toward exploration. A small ω can improve the convergence accuracy of the population, which is more suitable for exploitation. Farooq et al.[14] proposed a strategy: opposition-based initialization for inertia weight. To maintain a balance between the proportions of exploration and exploitation capacities, the inertia weight decreases linearly in two stages. To improve PSO's searching capability, Gupta et al.[15] employed four adaptive inertia weight approaches. To improve its optimization features, Tanweer et al.[16] presented a self-adjusting inertial weight to govern particle dynamics. Liu et al.[12] proposed a chaos-based inertia weight, a nonlinear kind of value taking that varies over the iteration process and is highly volatile.

(2) Neighborhood topology

Many scientists, especially when handling multimodal problems, employ neighborhood topology to steer the search trajectory of particles through the information exchange mechanism and improve the diversity of a population. Li et al.[17] developed an adaptive complex network architecture based on fitness distance correlation to efficiently balance the population's global and local search capabilities. Using ring topology, the PSO algorithm proposed by Li et al. can act as a kind of niche algorithm by forming a stable network that retains the best positions obtained thus far by using the local memory of individual particles while these particles more broadly explore the search space[18]. Xia et al.[19] utilized a dynamical topology for a multi-swarm particle swarm, periodically reducing the number of subgroups to enhance exploration and exploitation capabilities. Liang and Suganthan[20] adopted and explained a unique, dynamic multi-swarm PSO where the population is split up into several subswarms that frequently reassemble to exchange information. The strategy is used to obtain better performance on a complex multimodal optimization problem[20].

(3) Learning strategy

Many variants of PSO applied different types of learning strategies to improve their performance[21–23]. To better improve the development ability of the population, heterogeneous comprehensive learning PSO (HCLPSO) is proposed, which divides the population into two subpopulations and learns through different comprehensive learning strategies to focus on either exploration or exploitation[24]. Zhan and Zhang[25] presented orthogonal learning particle swarm optimization (OLPSO), which uses the orthogonal learning technique to build a promising sample to lead the particles on a better path.

(4) Hybrid algorithm strategy

The strengths of different algorithms are combined with other excellent optimization algorithms, which are also often employed to improve the algorithm's solving ability[26]. By constructing genetic learning paradigms using genetic operators, Gong et al.[27] introduced a genetic learning PSO (GLPSO) that combines the advantages of PSO and GA. The samples created by crossover, mutation, and selection for the historical data of particles are both qualified and diversified[27]. Chen et al.[28] developed bee-foraging learning PSO (BFLPSO), which has three different learning phases to strengthen the search capabilities of the population. Because of the excellent global search performance of ABC, many scholars have tried to combine ABC with PSO, thus developing the ABC-PSO hybrid algorithm[29].

Although many PSO variants have greatly improved PSO, they are unable to successfully compromise between the exploration and exploitation of algorithms, and issues like ineffective search efficiency remain for some challenging global optimization problems. A dual-stage hybrid learning particle swarm optimization

strategy was proposed in the research to increase PSO's effectiveness. The following are the contributions of this paper:

- For the population to conduct a sufficient performance for a better solution, a Manhattan distance based learning strategy is proposed.
- An excellent example learning strategy is presented for performing social learning of particles to reach the global optimal solution.
- The algorithm breaks the iterative process down into two stages, and the above two learning strategies are adopted in different stages. In the beginning, the Manhattan distance learning strategy is used, and in the end, the excellent example learning strategy is used.
- A Gaussian mutation strategy is used to quickly release the globally optimal particle from a locally optimal solution to improve precision. The method boosts the algorithm's performance on multimodal functions.

Formatting for the remaining text is as follows: In Section 2, a fundamental description of the PSO concept, social learning PSO (SLPSO), and related research are provided. The detailed implementation of the DHLPSO method is proposed and examined in Section 3. In Section 4, experimental testing is conducted. Section 5 serves as the paper's conclusion.

2 Related Work

2.1 Canonical PSO

For handling issues involving global optimization, PSO has lately gained popularity due to its quick convergence speed and limited operation parameters. Each particle of PSO has its velocity and position, and updates in the search region with each generation of velocity and position to get an optimal solution. Specifically, the updating equation of canonical PSO is expressed as

$$V_i(t+1) = \omega \cdot V_i(t) + c_1 \cdot r_1 \cdot (\text{Pbest}_i(t) - X_i(t)) + c_2 \cdot r_2 \cdot (\text{Gbest}(t) - X_i(t)) \quad (1)$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \quad (2)$$

where the position and speed of the i-th particle are $X_i(t)$ and $V_i(t)$, respectively; t is the current generation number; ω is inertia weight; c_1 and c_2 are acceleration coefficients, r_1 and r_2 are random numbers between 0 and 1; and i ranges from 1 to N, where N represents the overall number of population particles. $\text{Pbest}_i(t)$ denotes the historical optimum for each particle, while $\text{Gbest}(t)$ is the current global optimum. Canonical PSO can be regarded as two types of learning behaviors, namely, self-learning and social learning. Self-learning allows particles to learn their own historical optima, and social learning allows particles to learn the global optima. Thus, particle velocity points to a better direction in the search domain. Inertia weights are typically employed to regulate the amount of previous generation velocity information that is retained for updating. To balance PSO exploitation and exploration, smaller weights are preferable for local search, whereas larger weights are often better for a worldwide search.

2.2 Social learning PSO

Social learning PSO (SLPSO) is one of the PSO variants with strong performance and inherits the advantages of PSO[22, 30]. SLPSO achieves social learning by imitating the sociological concept in a society that each person learns from someone better than himself. In SLPSO, each particle learns a different goal and conducts social average learning. Different from traditional PSO, Pbest and Gbest are utilized for learning. As a result, such an action improves population diversity and is more conducive to an algorithm exploration search, resulting in a better solution and the possibility to avoid particles from falling into a local minimum.

In addition to fitness evaluation, the most critical components in SLPSO are population sorting and behavior learning. Before the learning operation, the population will be sorted to facilitate subsequent operations. The updating equations of SLPSO are presented as follows:

$$V_{i,j}(t+1) = r_1 \cdot V_{i,j}(t) + r_1 \cdot (X_{k,j}(t) - X_{i,j}(t)) + r_2 \cdot \varepsilon \cdot (\bar{X}_j(t) - X_{i,j}(t)) \quad (3)$$

$$\varepsilon = \beta \cdot \frac{n}{m} \quad (4)$$

$$X_{i,j} = \begin{cases} X_{i,j}(t) + V_{i,j}(t+1), & \text{if rand} < P_i; \\ X_{i,j}(t), & \text{otherwise} \end{cases} \quad (5)$$

where t represents the current generation and r_1, r_2, and r_3 are three random vectors that are generated and evenly dispersed in (0, 1). $X_{k,j}$ is the value of the k-th particle in the population in the j-th dimension, which has a better fitness value than particle i. As the population has been sorted in ascending order, note that particle k is different in each dimension, whose range is [1, i]. The control ε is denoted as the social influence factor. The learning probability P_i is used to determine

whether or not the current particle should be updated. The learning possibility of each particle is different, and the better the particle is, the smaller the probability. P_i is calculated as

$$P_i = (i - \frac{i-1}{N})^{\alpha \cdot \log(\lceil \frac{D}{M} \rceil)} \quad (6)$$

From Eq. (6), the population's overall size is N, and the particle's dimension is D. α and M are fixed values of 0.5 and 100, respectively. Compared with PSO, SLPSO is different in three parts. The first part replaces the inertia weight in the PSO with a random number, which improves the randomness and diversity of the population. The second part replaces the self-historical optimality in the individual cognitive part with being superior to the self-individual. The global optimal solution is changed to the global average in each dimension in the third part. In SLPSO, the performance of PSO in enhancing exploration and exploitation is effectively improved through two learnings.

3 Proposed Algorithm

Although SLPSO improves many performances compared with PSO, it has certain shortcomings in handling more complicated global optimization problems. There are three main reasons for these shortcomings: SLPSO replaces inertia weights with random numbers. Random numbers can improve the population's exploration capabilities in the early stages of iteration, and they can also affect the population's local exploitation and have an impact on its convergence in the latter stages. The population then rapidly converges as each particle learns toward the population average and the ideal solution, making it challenging to identify the genuine global optimum. Additionally, the overall computational efficiency is impacted by the fact that there are few updates for the population's best option in the iterative process. This work presents a dual-stage hybrid learning PSO (DHLPSO), which is inspired by the aforementioned research and aims to address some of PSO's weaknesses. The evolution process of the population is divided into two stages. The frame diagram of DHLPSO is described in Fig. 1. The entire iterative procedure is broken into two stages in this algorithm.

In the first stage, a Manhattan distance based learning strategy is carried out mainly for population exploration to obtain potential optimal solutions. In the next stage, an excellent example learning strategy is adopted, that is, the development operation is performed near the current solution to obtain the optimal solution. A good balance of exploration and exploitation is achieved through the strategic hybrid of these two stages. Additionally, a Gaussian mutation strategy for particle stagnation is performed throughout the iterations. With an increase in the iterative process, the variable asynchronous length will continue to decrease. The performance gain between algorithm exploration and exploitation is achieved using these three strategies in two stages. The following subsections provide detailed descriptions of these three strategies.

3.1 Manhattan distance based learning strategy

The distance between two points is estimated by comparing the absolute values of each dimension. The

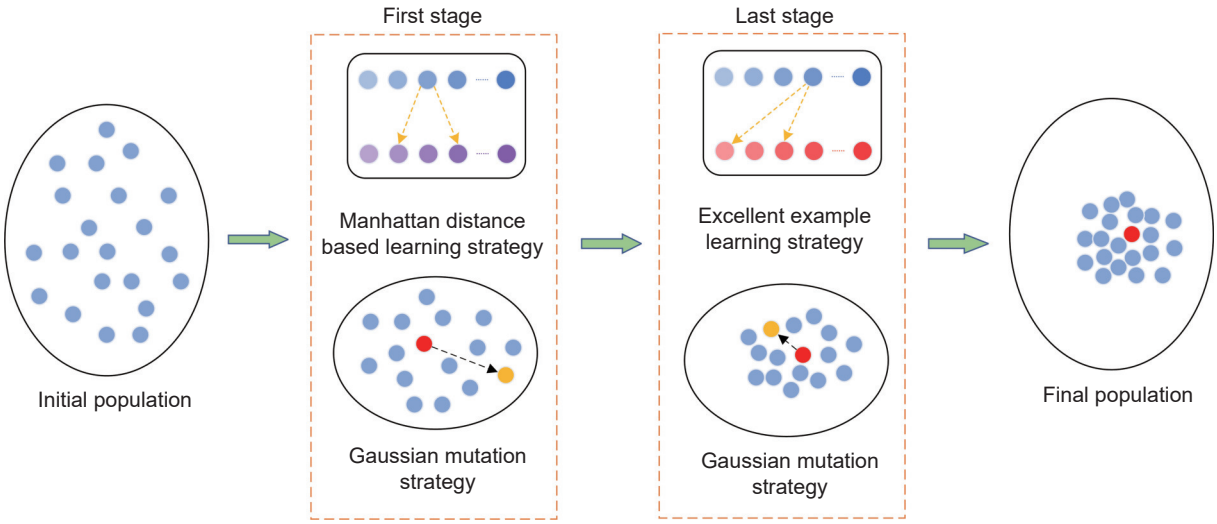


Fig. 1 Frame diagram of DHLPSO.

Manhattan distance is a geometric phrase that describes the distance as the product of the absolute differences between the two points' Cartesian coordinates[31]. For example, in 2D coordinates, the Manhattan distance between two points (x_1, y_1) and (x_2, y_2) is $|x_1 - x_2| + |y_1 - y_2|$.

Consider Fig. 2 as an example to specifically illustrate the Manhattan distance. The green line's length represents the Manhattan distance between the two particles, while the blue circle represents the particle's position. The Euclidean distance between the two particles is shown by the length of the red line, the size of which is $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. The Manhattan distance reduces the amount of computation relative to the Euclidean distance and effectively shows the information of each dimension of the particle.

This paper utilizes the Manhattan distance difference between every two particles for learning as a direct consequence of this discovery. Information about every dimension of the particle can be used to seek a better solution to the space problem using the Manhattan distance. However, if a particle only learns from the farthest Manhattan distance object, which will oscillate erratically and is hard to get a better solution. Therefore, a direction of learning from the better particle is added, so that the particles do not substantially deviate while exploring. The Manhattan distance based learning strategy's velocity updating equation is as follows:

$$V_i(t+1) = \omega_1 \cdot V_i(t) + r_1 \cdot c_1 \cdot (\text{Pbest}_r(t) - X_i(t)) + r_2 \cdot c_2 \cdot (\text{Pbest}_m(t) - X_i(t)) \tag{7}$$

$$\omega_1 = \text{normrnd}(0.4, 0.5) \tag{8}$$

$$c_1 = 2 \cdot \frac{\text{FE}}{\text{MaxFE}} \tag{9}$$

$$c_2 = 2 - c_1 \tag{10}$$

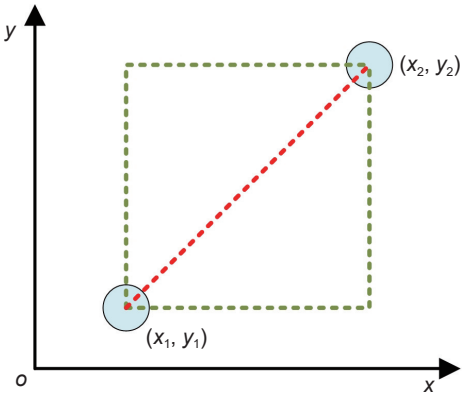


Fig. 2 Illustrations of Manhattan distance.

where ω_1 is the inertia weight, which is randomly selected using a Gaussian distribution with a mean and standard deviation of 0.4 and 0.5, to control the effect of the speed of the previous generation on the next generation. r_1 and r_2 are random vectors that are evenly distributed to each component in [0, 1]. The maximum number of fitness evaluations is MaxFE, and the present fitness value has been evaluated FE times. c_1 and c_2 are two acceleration coefficients, which change according to the iterative process and can be used to control the parameters of the learning step. Two parameters are linearly increased or decreased to govern the population's exploration and exploitation capacity. Pbest_r indicates the particle's ideal position relative to the current particle in the population. Since the population is sorted at each iteration, the value of ranges from 1 to i. The historical optimal solution of the particle with the largest Manhattan distance from the current particle is denoted by Pbest_m. The equation for calculating the Manhattan distance is described below

$$\text{md}_{i,j} = \sum_{d=1}^{D} |X_i^d - X_j^d| \tag{11}$$

$$m_i = \max(\text{md}_i) \tag{12}$$

According to Eq. (11), each particle's Manhattan distance from other particles in the population is calculated, resulting in the symmetric matrix md. Next, the particle with the largest Manhattan distance corresponding to each particle in this symmetric matrix is identified. Note that each particle has the corresponding largest Manhattan distance particle, while the later particle corresponding to the largest Manhattan distance particle is not necessarily the former.

As shown in Fig. 3, x_i learned from x_m and x_k, which are the farthest Manhattan distance particle and a better

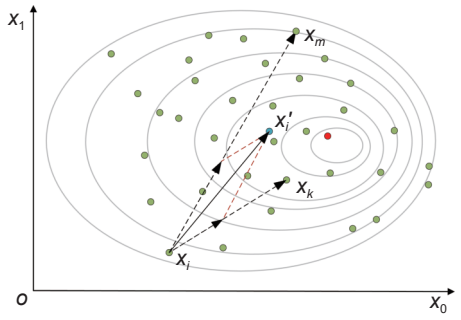


Fig. 3 Illustrations of Manhattan distance based learning strategy.

particle, respectively. Through the learning of the two, x_i', which is closer to the optimal point than the previous particle, is obtained. Through this phenomenon, it can be seen that the Manhattan distance learning strategy can guide the particles to a promising direction, which can be used for the exploration and search of the population.

3.2 Excellent example learning strategy

Inspired by SLPSO, each particle in the population learns from particles with better performance than it, which can enhance the population's ability to be exploited. The particles can be dispersed close to the ideal solution after Manhattan distance difference learning in the preliminary step. An excellent learning strategy would be to do a local search to improve the algorithm's solution accuracy. The updated equation of the excellent example learning strategy is presented as

$$V_i(t+1) = \omega_2 \cdot V_i(t) + r_3 \cdot (\text{Pbest}_r(t) - X_i(t)) + r_4 \cdot (\text{Gbest}(t) - X_i(t)) \tag{13}$$

$$\omega_2 = \text{normrnd}(0.6, 0.4) \tag{14}$$

where ω_2 is the inertia weight of the strategy. r_3 and r_4 are random vectors from 0 to 1. Gbest denotes the current population's best global position. Pbest_r refers to the historical optimal value of a particle with a better fitness value than the current particle. In this strategy, through the learning of Gbest and random optimal particles, in the process of obtaining a close approximation to the existing global best solution, the particle identifies a better solution on its path, thereby further converging to the global optimal solution. The difference between an excellent example learning strategy and social learning strategy of SLPSO is that the mean of the population component is replaced by Gbest.

Figure 4 illustrates the excellent example learning strategy. The current particle x_i is in the process of learning from x_{gbest} and x_k, which means the best particle of global and randomly selected better particles, respectively. Through such an example learning, a new position learned by the present particle can enhance the exploitation ability of the algorithm so that the particles in the population can get better convergence.

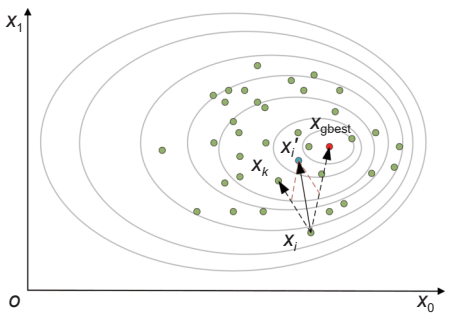


Fig. 4 Illustrations of excellent example learning strategy.

3.3 Gaussian mutation strategy

The mutation operation is a widely accepted approach for maintaining population variety, and it is commonly applied in PSO algorithms. There are various existing mutation strategies, such as Gaussian, Cauchy, and Levy. The Gaussian mutation is typically regarded as better suited for local search, while other methods as mentioned above are employed for global optimization. In this paper, to make it easier for stagnant particles to escape local traps, a Gaussian mutation strategy is implemented to perturb particles in a stagnant state. Its mathematical expression is presented as follows:

$$\text{mGbest}_j(t) = \text{Gbest}_j(t) + \text{sign}(2 \cdot r_5 - 1) \cdot (x_{\max} - x_{\min}) \cdot \text{gaussian}(0, \text{sig}^2) \tag{15}$$

$$\text{sig} = 1 - 0.9 \cdot \frac{\text{FE}}{\text{MaxFE}} \tag{16}$$

where j refers to a random dimension from [1, D] for the current Gbest, sign() determines the perturbation direction of the particle, and $x_{\max} - x_{\min}$ ensures that it will not exceed the constraint range. Through gaussian(), a Gaussian distribution random number within a limited range is generated, which is multiplied by the size of the decision space to achieve a perturbation purpose. The Gaussian disturbance's size is governed by the standard deviation's size, represented by the symbol sig.

The fundamental of realizing the Gaussian mutation strategy is to perform a Gaussian mutation perturbation on the global optimal particle when a particle falls into the local optimum and reaches a certain number of stagnation times. In this way, the purpose of locally perturbing the stagnant particles and jumping out of the current stagnant position is achieved. The best particle's fitness value after the disturbance is calculated if the value is better than that before updating it. Otherwise, the population continues to use the original solution.

3.4 Proposed DHLPSO algorithm

Through the introduction in the previous part, the two learning strategies of this algorithm are explained. This research presents a dual-stage hybrid learning strategy that uses early exploration and late exploitation to

increase PSO performance effectively. To make fully utilize the advantages of staged learning, different normal distribution parameters are set in the two stages to generate inertia weights. Algorithm 1 depicts the DHLPSO procedure in detail.

In Algorithm 1, the termination criterion is when FE is greater than MaxFE, that is, when the number of evaluations reaches the specified maximum number of evaluations, the optimal value given by the output algorithm. ft is set as the node of strategy transition and the value of ft is verified through experiments. In the first stage, the Manhattan distance learning strategy is adopted. Each particle learns from the particle with the farthest Manhattan distance and the particle that is better than itself. The former preserves the population's exploratory nature and allows it to thoroughly hunt for the best value within the viable zone, while the latter ensures that the particles have a certain degree of exploitation and will not escape when the optimal solution is obtained.

Algorithm 1 Dual-stage hybrid learning PSO

Begin

Population size N; dimension D; maximum stagnation generation $\text{stag}_{\max}$; phasing point ft

Initialize population X and velocity V;

Evaluate the fitness values of particles, FE = N;

While termination criterion is not met **do**

  Sort all particles based on the fitness values from the best to worst;

  $k = \text{ceil}(\text{rand} \cdot [i : \text{NP}])$;

  If ft < $(\frac{\text{FE}}{\text{MaxFE}})$**then**

    /*Manhattan distance based learning strategy*/

    Calculate c_1, c_2, and ω_1;

      Get m according to Eqs. (11) and (12);

      Update particles according to Eqs. (7) and (2);

  Else

      /*Excellent example learning strategy*/

      Calculate ω_2;

      Update particles according to Eqs. (13) and (2);

  End If

  Evaluate the fitness values $F(X)$;

  FE = FE + N;

  /*Gaussian mutation strategy*/;

  Calculate the sig according to Eq. (16);

  For i = 1 to N **do**

    If st(i) > $\text{stag}_{\max}$ **then**

    $j = \text{randi}(1, D)$;

    Perturb the mGbest according to Eq. (15);

    End If

    Evaluate the fitness of mGbest;

    If f(mGbest) < f(Gbest) **then**

    Gbest = mGbest;

    FE = FE + 1;

    End If

  End For

End While

End

The majority of the particles are gathered close to the ideal value after the initial learning stage. An excellent example learning strategy is implemented in the next stage, which can make the particles continue to approach the optimal value, to reach the position of the optimal solution. However, when solving multimodal problems, there are often multiple optimal solutions. At this time, the excellent learning strategy may not be able to obtain the optimal value, and it may fall into a local optimum. Therefore, about stagnant particles, the Gaussian mutation strategy is used to disrupt the population's global optimum particle, which can cause particles to escape from local optima and move the population toward a better solution.

4 Experimental Verification

In this study, experiments are conducted to evaluate the applicability of our proposed DHLPSO. First, the paper analyzed the sensitivity of the parameters by setting different values to conduct investigations. Second, to assess the efficiency of DHLPSO on the CEC 2013, comparative experiments are conducted in 30 and 50 dimensions with other state-of-the-art PSO variations. We analyze each DHLPSO strategy's quality and compare it in isolation to the complete algorithm in order to assess its efficacy. Last, a comparison with other algorithms is performed to test the convergence of the algorithm proposed in this paper.

4.1 Experiment configurations

To demonstrate that the provided strategies perform well on global optimization problems, the paper selected the CEC2013[32–34] test suite for verification, which includes unimodal functions ($f_1 - f_5$), multimodal functions ($f_6 - f_{20}$), and composition functions ($f_{21} - f_{28}$). The mean, sorting, and average sorting are used to assess the algorithm's optimization performance in this work. The lower the algorithm's average value is, the better its performance and the higher its ranking.

The experimental results achieved by the DHLPSO algorithm at CEC 2013 are contrasted with those

gained with other excellent PSO variants. These algorithms are relatively classic and excellent algorithms in recent years, so they are comparative and representative. Table 1 displays the necessary parameter settings for various comparison algorithms. To show fairness, all algorithms in the experiments adopt the same parameter settings below in all cases: MaxFE = $D \times 10\,000$ and the number of running for all algorithms is MaxN = 51.

The evaluation is performed by comparing the average of the algorithm running multiple times with the ranking gained by each algorithm. The ranking is decided by computing the difference between the algorithm's real ideal value and the theoretical optimal value for each run on each function. Next, we compare the average of each algorithm on the function, and the algorithm with the lower average will receive a higher ranking. While the ranking can more specifically demonstrate the algorithm's competitive edge, the average number can demonstrate the algorithm's overall performance.

4.2 Sensitivity of parameters

There are two important basic parameters for the proposed DHLPSO. The first parameter is the staging point ft, and the second parameter is the maximum number of stalls $\text{stag}_{\max}$ of each particle. If ft is too little, the population may not be extensive enough to obtain the best solution, and the population may become bound in a local minimum. If the value of ft is too large, the population may not achieve convergence accuracy. Similarly, if $\text{stag}_{\max}$ is too large, the strategy is unable to reach the expected effect. If $\text{stag}_{\max}$ is too small, it would consume too many function evaluations. As a result, in this section, a series of tests are carried out to see how ft and $\text{stag}_{\max}$ affect the performance of DHLPSO. To adequately examine the impact of variables, we set the ft to 0.55, 0.60, 0.65, and 0.70 and set $\text{stag}_{\max}$ to 5, 10, 15, and 20. To evaluate the effectiveness of these 16 DHLPSO versions, we use the CEC 2013 test suite. From Tables A1 and A2 in the Appendix, the average results of the compared algorithms are shown, where the optimal value in each function is marked in bold. At the end of Tables A3 and A4 in the Appendix, it is also stated how many functions each DHLPSO variation uses to achieve the best results. As shown in Tables A1 and A2 in the Appendix, when the stage point ft is fixed, the larger $\text{stag}_{\max}$ is employed, the worse the performance of the DHLPSO variants. However, when $\text{stag}_{\max}$ is set to 10, the performance of DHLPSO gains the best. When $\text{stag}_{\max}$ is too small, it will take up too many functional evaluation times, lowering the efficiency of population optimization. Therefore, $\text{stag}_{\max}$ is set to 10. When $\text{stag}_{\max}$ is fixed, we note that when stage point ft is set to 0.65, the effect is the best, and the best solutions are obtained with a total of 12. The longer the population adopts the Manhattan distance learning strategy in the early stage, the better solution can be fully searched in the search space. However, this time should not be so long that there is not enough time to converge on the excellent example learning strategy and that the optimal solution cannot be obtained. Therefore, the parameters ft and $\text{stag}_{\max}$ are set to 0.65 and 10, respectively.

4.3 Performance comparison of the CEC2013

The research examines the optimization performance of DHLPSO considering seven advanced PSO variants, which have correlations with DHLPSO, as contrasting algorithms. These comparison algorithms are SLPSO, CLPSO, XPSO, GLPSO, HCLPSO, BLPSO, and BFLPSO. According to the relevant references, these several comparison algorithms have advantages over PSO and other variants; thus, these competitive algorithms were selected. The experimental DHLPSO results and contrasting algorithms for the CEC2013 30-dimensional function are shown in Table A3 in the Appendix. In Table A3 in the Appendix, Count is the number of times each algorithm has achieved the first

Table 1 Parameter settings of all the contrasted algorithms.

Algorithm	Reference	Parameter settings
CLPSO	[23]	$\omega = [0.4, 0.9]$, $c_1 = c_2 = 1.494\,45$, $N = 40$, $m = 7$, $p_c = [0.05, 0.5]$
SLPSO	[22]	$N = 100$, $\alpha = 0.5$, $\beta = 0.01$
HCLPSO	[35]	$\omega = [0.2, 0.99]$, $c_1 = [0.5, 2.5]$, $c_2 = [0.5, 2.5]$, $c = [1.5, 3]$
GLPSO	[27]	$N = 50$, $\omega = 0.7298$, $c = 1.496\,618$, pm = 0.01, sg = 7
BLPSO	[36]	$N = 40$, $\omega = [0.2, 0.9]$, $c = 1.494\,45$, $G = 5$
XPSO	[26]	$N = 60$, $\eta = 0.2$, $\text{stag}_{\max} = 5$, $p = 0.2$
BFLPSO	[28]	$N = 40$, $\omega = [0.2, 0.9]$, $c = 1.494\,45$, $I = E = 1$, $G = 5$

rank in 28 functions; Fun is the number of functions.

As shown in Table A3 in the Appendix, for the metric average ranking (Ave), DHLPSO obtains 3.18 and ranks the first, in comparison to the other seven algorithms. DHLPSO gains the first on f_1 and f_5, while DHLPSO can be ranked the second on f_2 and f_4 on these unimodal functions. The complexity of these problems lies in the notion that their fitness landscapes contain smooth local imperfections as well as smooth but narrow ridges, as well as correlations among variables. The comparison results show that DHLPSO has a stronger optimization ability than the selected contrasting algorithm. The outstanding performance of DHLPSO benefits from the robust development capability in the later stage, which makes the algorithm perform better on unimodal functions.

The multimodal functions f_6−f_{20} are among the functions assessed in Table A2 in the Appendix and can be utilized to determine whether the proposed algorithm's strategy has increased exploration ability. In contrast to the other algorithms, DHLPSO still provides the most excellent performance on these multimodal functions. DHLPSO achieves the best results on f_6, f_7, and f_9, and the other functions are also better than different algorithms in general. The primary reason is that DHLPSO's Manhattan distance learning strategy promotes population diversity in the early stages by learning the examples with the maximum Manhattan distance, which enables the algorithm to better search for multimodal functions. On f_{12}, f_{13}, and f_{20}, BFLPSO obtains the best mean value, which is the best algorithm, with the exception of DHLPSO.

Among the eight composition functions, including f_{21}−f_{26}, the proposed DHLPSO algorithm still exhibits the best performance. DHLPSO yields the best performance on f_{22} and f_{26}. In terms of the 30-dimensional ranking of CEC2013, the algorithm ranking is DHLPSO, BFLPSO, BLPSO, HCLPSO, GLPSO, SLPSO, XPSO, and CLPSO. This finding proves that DHLPSO achieves significantly better optimization performance, demonstrating the effectiveness of the two learning strategies of DHLPSO performance. A comparison of these excellent PSO variant algorithms shows that DHLPSO has a better ability to solve global optimization problems.

As shown in Table A2 in the Appendix, DHLPSO can rank the 1st seven times and can rank the 2nd eight times. SLPSO, CLPSO, XPSO, GLPSO, HCLPSO, BLPSO, and BFLPSO can rank the first 5, 3, 2, 4, 1, 3, and 4 times, respectively. The results of DHLPSO on CEC 2013 are better than the results of other comparison algorithms, as shown by the above comparison results and values. In general, DHLPSO achieved the most significant optimization performance. Furthermore, DHLPSO has adequately solved some of its problems by balancing the exploration and exploitation of the population through staged learning. Theoretical research and actual results show that the DHLPSO performs much better than other PSO versions.

To further verify the stability of DHLPSO and its ability to solve higher-dimensional complex problems, this paper still chooses the previous seven algorithms as comparison algorithms. The 50-dimensional functions of the CEC 2013 are being tested. The proposed PSO variation DHLPSO demonstrates greater performance than the other comparison algorithms, as demonstrated in Table A3 in the Appendix. For the metric Ave, SLPSO is ranked the seventh with a score of 5.36. DHLPSO is ranked top with a score of 3.11, outperforming the other seven algorithms. This result proves the effectiveness of the dual-stage learning strategy and Gaussian mutation strategy proposed in this paper. Compared with BFLPSO, DHLPSO has a similar search ability on multimodal functions. However, DHLPSO achieves the best performance in 8 out of 28 functions, and five functions ranks the second. This performance is enough to indicate that DHLPSO has strong optimization ability and scalability.

4.4 Strategy effectiveness analysis

Theoretically, the algorithms of the two learning strategies proposed in this paper are divided into one strategy for exploration, another strategy for exploitation, and a stagnation-handling strategy for jumping out of local optima. Three streamlined versions of DHLPSO—DHLPSO_1, DHLPSO_2, and DHLPSO_3—are developed to test the effects of the three strategies in the paper. DHLPSO_1 refers to the PSO variant that uses only the Manhattan distance learning strategy, while DHLPSO_2 is the PSO variant that only uses the excellent example learning strategy. DHLPSO_3 is a merge of the first two strategies without the PSO variant using the Gaussian mutation strategy. Four different types of functions are employed for verification. For more illustrative purposes, the four functions include Sphere (f_1), Different Powers (f_2), Rastrigin (f_3), and Composition of CEC2013 (f_4). The functions f_1 and f_2 are unimodal, f_3 is multimodal, and f_4 is composite.

Table 2 and Fig. 5 demonstrate that DHLPSO outperforms DHLPSO_1 and DHLPSO_2 on all four functions. This finding illustrates the effectiveness of adopting staged learning and improves the efficiency of algorithm exploration. DHLPSO_1 has good exploratory properties, making it impossible to obtain higher-precision solutions. Due to high exploitability of DHLPSO_2, it is also simple to reach a local optimum. The DHLPSO_3 was acquired by the staged mixing of the two. Functions f_1, f_2, and f_4 show the performance of DHLPSO_3 is better than their individual effects.

The third function demonstrates that the population's performance on function f_3 is greatly improved by the Gaussian mutation strategy, which is a multimodal function. The Gaussian mutation strategy can take a solution locked in a local optimum to easier jump out and into the global space, resulting in a superior solution.

4.5 Nonparametric test

To further show the differences between DHLPSO on the CEC2013 test set and these PSO variants, this paper selects the Wilcoxon signed-rank test[37], a nonparametric statistical analysis method. The purpose of this is to find significant variations between the means of two samples.

The Wilcoxon nonparametric test refers to: A comparison of the two algorithms' performance ratings for the i-th problem among n problems is denoted by d_i. Differences are ordered by absolute value; in the case of a tie, the average rank is used to handle the tie (for example, if there are two differences when possessing ranks 1 and 2, distribute a rank of 1.5 to both differences). Due to its wide range of employing, statistical software programs can be used to calculate the p-value for the test. The p-value provides

Table 2 Parameter settings of all the contrasted algorithms.

Algorithm	Metric	f_1	f_2	f_3	f_4
DHLPSO_1	Mean	4.55×10^{-14}	1.82×10^{-13}	$7.56\times10^{+0}$	$2.32\times10^{+2}$
	Std	1.02×10^{-13}	6.23×10^{-14}	$1.51\times10^{+0}$	$7.01\times10^{+1}$
DHLPSO_2	Mean	2.27×10^{-13}	$4.24\times10^{+1}$	$1.26\times10^{+2}$	$2.63\times10^{+2}$
	Std	$0.00\times10^{+0}$	$9.47\times10^{+1}$	$6.48\times10^{+1}$	$8.58\times10^{+1}$
DHLPSO_3	Mean	$0.00\times10^{+0}$	1.82×10^{-13}	$1.11\times10^{+1}$	$2.51\times10^{+2}$
	Std	$0.00\times10^{+0}$	6.23×10^{-14}	$5.04\times10^{+0}$	$6.98\times10^{+1}$
DHLPSO	Mean	**$0.00\times10^{+0}$**	**1.59×10^{-13}**	**3.77×10^{-2}**	**$2.26\times10^{+2}$**
	Std	**$0.00\times10^{+0}$**	**6.23×10^{-14}**	**7.00×10^{-2}**	**$5.75\times10^{+1}$**

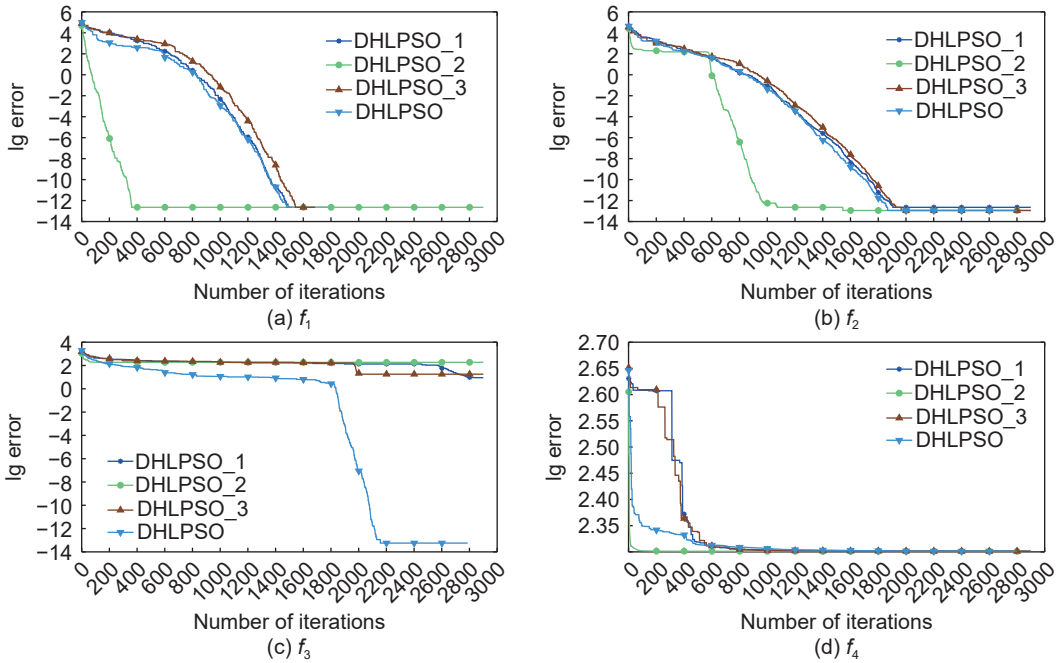


Fig. 5 Convergence curves of the four strategies.

information about whether a statistical hypothesis test is significant, and also indicates the significance of the result: a smaller p-value indicates a smaller correlation between the compared algorithms. IBM SPSS is chosen as the calculation tool.

Let R^+ be the rank sum of the problem for which the previous algorithm outperforms the back algorithm. R^- is the rank sum of the algorithm inverse. Grades with $d_i = 0$ are evenly divided between the sums; if they have an odd number, disregard one. Equations (17) and (18) demonstrate how to calculate the p-values.

$$R^+ = \sum_{d_i>0} \text{rank}(d_i) + \frac{1}{2}\sum_{d_i=0} \text{rank}(d_i) \tag{17}$$

$$R^- = \sum_{d_i<0} \text{rank}(d_i) + \frac{1}{2}\sum_{d_i=0} \text{rank}(d_i) \tag{18}$$

The Wilcoxon nonparametric test results of DHLPSO and these contrast algorithms in the two groups of 30-dimensional and 50-dimensional functions from the CEC2013 are shown in Tables 3 and 4. The formula $n/w/t/l$ denotes that there are n total experimental functions, of which DHLPSO succeeds on w, is equal on t, and fails on l.

The apparent difference between DHLPSO and SLPSO in Table 3 is 5.00×10^{-3}, which is less than 0.05. This value demonstrates that the DHLPSO is predominant over SLPSO. It can be seen that among the 28 optimization functions, 20 of DHLPSO are better than SLPSO, so the effect of DHLPSO is significantly better than that of SLPSO. For HCLPSO, BLPSO, and BFLPSO, their p-values are larger than 0.05, indicating that DHLPSO does not considerably outperform their performance. Nevertheless, it is clear from CEC2013 that there are more DHLPSO functions that are superior to them than worse ones. This result indicates that DHLPSO is still better than that of the remaining PSO variants. Due to the fact that the p-values of them are less than 0.05, DHLPSO is clearly superior to CLPSO, GLPSO, and XPSO.

Simultaneously, Table 4 shows that all the p-value are less than the value of Table 3. This finding also demonstrates that in 50 dimensions, DHLPSO outperforms the comparison algorithm. The numbers indicate that DHLPSO is superior to SLPSO, CLSPO, XPSO, GLPSO, HCLPSO, BLPSO, and BFLPSO for the number 23, 20, 22, 18, 17, 19 and 18, respectively, which demonstrates that DHLPSO is significantly better than the comparison algorithms. Overall, DHLPSO significantly outperforms some PSO variants in global optimization problems.

4.6 Convergence analysis

To more intuitively compare the convergence performance of DHLPSO and these comparison algorithms, the comparison algorithm is selected here to show the convergence graph of the four functions (f_1, f_6, f_{10}, and f_{26}), which is shown in Fig. 6. The four functions are selected in the CEC2013 test set. Among

Table 3 Nonparametric test results from Table A2 in the Appendix.

Comparison	R^+	R^-	p-value	n / w / t / l
DHLPSO vs. SLPSO	306	73	5.00×10^{-3}	28 / 20 / 1 / 7
DHLPSO vs. CLPSO	362	44	2.71×10^{-4}	28 / 23 / 0 / 5
DHLPSO vs. XPSO	337	69	2.19×10^{-3}	28 / 21 / 0 / 7
DHLPSO vs. GLPSO	310	97	1.46×10^{-2}	28 / 19 / 0 / 9
DHLPSO vs. HCLPSO	250	157	2.84×10^{-1}	28 / 17 / 0 / 11
DHLPSO vs. BLPSO	243	163	3.60×10^{-1}	28 / 18 / 0 / 10
DHLPSO vs. BFLPSO	238	169	4.29×10^{-1}	28 / 16 / 0 / 12

Table 4 Nonparametric test results from Table A3 in the Appendix.

Comparison	R^+	R^-	p-value	n / w / t / l
DHLPSO vs. SLPSO	345	62	1.22×10^{-3}	28 / 23 / 0 / 5
DHLPSO vs. CLPSO	334	73	2.86×10^{-3}	28 / 20 / 0 / 8
DHLPSO vs. XPSO	344	62	1.25×10^{-3}	28 / 22 / 0 / 6
DHLPSO vs. GLPSO	306	101	1.89×10^{-2}	28 / 18 / 0 / 10
DHLPSO vs. HCLPSO	271	136	1.21×10^{-1}	28 / 17 / 0 / 11
DHLPSO vs. BLPSO	256	150	2.24×10^{-1}	28 / 19 / 0 / 9
DHLPSO vs. BFLPSO	248	159	3.07×10^{-1}	28 / 18 / 0 / 10

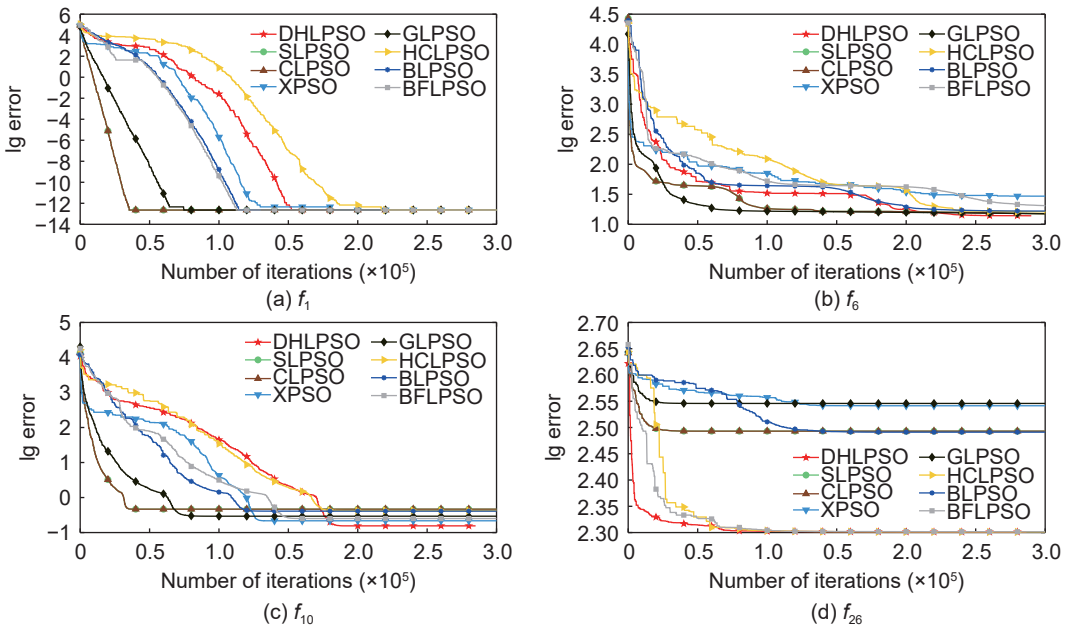


Fig. 6 Convergence curves of the eight comparison algorithms.

these functions, f_1 is an unimodal function, f_6 and f_{10} are multimodal functions, and f_{26} is a composite function.

From Fig. 6, on the f_1 function, although the convergence speed of DHLPSO is relatively slow compared with other comparison algorithms, which is controlled by the global exploration of the Manhattan distance learning strategy in the early stage, DHLPSO quickly obtains the global maximum value after converging to half the number of iterations. DHLPSO convergence is slow in the early stages of iteration, but it is rapid in the later phases of iteration, according to functions f_6 and f_{10} from Fig. 6.

Because the excellent example learning strategy in the later stage has good exploitability, the final solution's convergence accuracy is superior to that of the other remaining algorithms. It continuously advances closer to the ideal value after initially exploring the area around the multimodal function's optimal value, resulting in strong optimization performance. For function f_{26}, DHLPSO's convergence is superior to other algorithms in that it more quickly reaches the ideal value. In conclusion, the convergence performance of DHLPSO is the best.

5 Conclusion

A novel DHLPSO is presented by designing a dual-stage hybrid learning to divide the iterative process into two learning stages with different features in this paper. First, the Manhattan distance learning strategy has been proposed to extensively explore the population in the early stages of iteration, which increases population variety and lowers the risk of being stuck in local optimums. Second, an excellent example learning strategy is employed to do a local search operation in the later stages of the iteration. It learns from the globally optimal particles and particles that are better than itself to achieve the purpose of population development. Two learning strategies are employed during these two phases to strike a balance between exploration and exploitation. Last, employing the proposed Gaussian mutation strategy makes it easier for stagnant populations to break free from local optima.

The DHLPSO algorithm outperforms other seven advanced PSO variants, according to the results of the CEC2013 benchmark test suit. In addition, the paper also conducts some comparative experiments to analyze the effectiveness of these three strategies. To implement the trade-off between exploration and exploitation, DHLPSO employs two learning strategies at dual-stage evolutionary stages. At the early stage, the Manhattan distance learning strategy is adopted to make the population thoroughly search within the feasible range. The presented staged learning in this study is the first time that PSO has been adopted, and it has a relatively good effect on the CEC2013 benchmark functions. Further investigation into the

appropriate improvement measures is warranted.

Although DHLPSO shows better performance, it still has some problems that need to be improved. The first is that it exhibits weak performance in solving composite problems. The second is that the Manhattan distance learning strategy reduces the overall convergence rate, especially on unimodal functions. This is an unavoidable problem with the nature of the algorithm. Finally, although there is a mutation strategy to adjust the stagnation situation, there will still be a situation of falling into a local optimum. Our further work will prioritize improving the performance of the algorithm. The other is to try to improve the algorithm to solve more complex optimization problems.

Appendix

- Table A1 and A2: Parameters ft and $stag_{max}$ investigation of DHLPSO.
- Table A3: Comparison results (D = 30).
- Table A4: Comparison results (D = 50).

Acknowledgment

This work was supported by the National Natural Science Foundation of China (Nos. 62066019 and 61903089), the Natural Science Foundation of Jiangxi Province (Nos. 20202BABL202020 and 20202BAB202014), and the Graduate Innovation Foundation of Jiangxi University of Science and Technology (Nos. XY2021-S092 and YC2022-S641).

Table A1 Parameters ft and $stag_{max}$ ($stag_{max}$=5, 10) investigation of DHLPSO.

Fun	$stag_{max}$ = 5				$stag_{max}$ = 10			
	ft = 0.55	ft = 0.6	ft = 0.65	ft = 0.7	ft = 0.55	ft = 0.6	ft = 0.65	ft = 0.7
	mean	mean	mean	mean	mean	mean	mean	mean
f_1	7.58×10^{-14}	**$0.00\times10^{+0}$**	**$0.00\times10^{+0}$**	**$0.00\times10^{+0}$**	5.31×10^{-14}	**$0.00\times10^{+0}$**	**$0.00\times10^{+0}$**	**$0.00\times10^{+0}$**
f_2	$9.25\times10^{+5}$	$8.98\times10^{+5}$	$8.35\times10^{+5}$	$1.16\times10^{+6}$	$8.23\times10^{+5}$	$8.67\times10^{+5}$	**$7.55\times10^{+5}$**	$1.11\times10^{+6}$
f_3	**$1.89\times10^{+7}$**	$5.03\times10^{+7}$	$2.84\times10^{+7}$	$3.86\times10^{+7}$	$3.42\times10^{+7}$	$2.67\times10^{+7}$	$3.31\times10^{+7}$	$1.93\times10^{+7}$
f_4	$1.36\times10^{+3}$	$1.26\times10^{+3}$	$1.91\times10^{+3}$	$1.87\times10^{+3}$	$1.14\times10^{+3}$	$1.22\times10^{+3}$	$9.77\times10^{+2}$	$1.75\times10^{+3}$
f_5	2.43×10^{-13}	**1.14×10^{-13}**	**1.14×10^{-13}**	1.33×10^{-13}	2.43×10^{-13}	2.20×10^{-13}	**1.14×10^{-13}**	1.59×10^{-13}
f_6	$2.33\times10^{+1}$	$1.70\times10^{+1}$	$1.77\times10^{+1}$	**$1.63\times10^{+1}$**	$1.95\times10^{+1}$	$1.86\times10^{+1}$	$1.95\times10^{+1}$	$1.68\times10^{+1}$
f_7	$8.66\times10^{+0}$	$8.66\times10^{+0}$	$9.43\times10^{+0}$	$9.72\times10^{+0}$	$8.87\times10^{+0}$	$7.54\times10^{+0}$	$6.67\times10^{+0}$	$8.06\times10^{+0}$
f_8	$2.09\times10^{+1}$	$2.09\times10^{+1}$	$2.09\times10^{+1}$	**$2.09\times10^{+1}$**	$2.09\times10^{+1}$	$2.09\times10^{+1}$	$2.09\times10^{+1}$	$2.09\times10^{+1}$
f_9	$3.09\times10^{+1}$	$3.08\times10^{+1}$	$3.20\times10^{+1}$	$2.89\times10^{+1}$	$3.11\times10^{+1}$	$3.17\times10^{+1}$	$3.09\times10^{+1}$	$3.08\times10^{+1}$
f_{10}	1.29×10^{-1}	1.56×10^{-1}	**1.19×10^{-1}**	1.41×10^{-1}	1.25×10^{-1}	1.30×10^{-1}	1.34×10^{-1}	1.30×10^{-1}
f_{11}	1.43×10^{-3}	4.33×10^{-3}	1.00×10^{-3}	4.86×10^{-3}	3.32×10^{-2}	2.24×10^{-13}	**1.02×10^{-13}**	1.88×10^{-1}
f_{12}	$7.69\times10^{+1}$	$7.87\times10^{+1}$	$9.14\times10^{+1}$	$8.04\times10^{+1}$	$7.19\times10^{+1}$	$7.65\times10^{+1}$	$7.74\times10^{+1}$	$7.39\times10^{+1}$
f_{13}	$1.13\times10^{+2}$	$1.13\times10^{+2}$	$1.13\times10^{+2}$	$1.09\times10^{+2}$	$1.10\times10^{+2}$	$1.05\times10^{+2}$	**$8.26\times10^{+1}$**	$1.05\times10^{+2}$
f_{14}	$1.36\times10^{+0}$	$1.64\times10^{+0}$	$1.33\times10^{+0}$	**$1.30\times10^{+0}$**	$6.45\times10^{+0}$	$5.86\times10^{+0}$	$5.19\times10^{+0}$	$5.61\times10^{+0}$
f_{15}	$4.89\times10^{+3}$	$4.35\times10^{+3}$	$4.56\times10^{+3}$	$4.48\times10^{+3}$	$4.27\times10^{+3}$	$4.41\times10^{+3}$	**$3.90\times10^{+3}$**	$4.58\times10^{+3}$
f_{16}	$1.33\times10^{+0}$	$1.43\times10^{+0}$	$1.24\times10^{+0}$	$1.26\times10^{+0}$	$1.22\times10^{+0}$	$1.18\times10^{+0}$	**7.43×10^{-1}**	$1.19\times10^{+0}$
f_{17}	$3.08\times10^{+1}$	$3.08\times10^{+1}$	$3.08\times10^{+1}$	$3.08\times10^{+1}$	$3.04\times10^{+1}$	$3.13\times10^{+1}$	$3.14\times10^{+1}$	$3.15\times10^{+1}$
f_{18}	$8.44\times10^{+1}$	$8.79\times10^{+1}$	$8.46\times10^{+1}$	$8.24\times10^{+1}$	$9.27\times10^{+1}$	**$7.25\times10^{+1}$**	$9.56\times10^{+1}$	$8.53\times10^{+1}$
f_{19}	$1.05\times10^{+0}$	**$1.02\times10^{+0}$**	$1.07\times10^{+0}$	$1.08\times10^{+0}$	$1.32\times10^{+0}$	$1.38\times10^{+0}$	$1.47\times10^{+0}$	$1.41\times10^{+0}$
f_{20}	$1.13\times10^{+1}$	$1.14\times10^{+1}$	$1.13\times10^{+1}$	$1.13\times10^{+1}$	$1.14\times10^{+1}$	$1.12\times10^{+1}$	**$1.04\times10^{+1}$**	$1.12\times10^{+1}$
f_{21}	$2.64\times10^{+2}$	$2.57\times10^{+2}$	**$2.44\times10^{+2}$**	$2.75\times10^{+2}$	$2.77\times10^{+2}$	$2.83\times10^{+2}$	$3.00\times10^{+2}$	$2.83\times10^{+2}$
f_{22}	$7.65\times10^{+1}$	$9.76\times10^{+1}$	$9.95\times10^{+1}$	$8.41\times10^{+1}$	$1.09\times10^{+2}$	$1.00\times10^{+2}$	**$6.46\times10^{+1}$**	$9.97\times10^{+1}$
f_{23}	$4.89\times10^{+3}$	$4.89\times10^{+3}$	$4.92\times10^{+3}$	$4.78\times10^{+3}$	$4.77\times10^{+3}$	$4.72\times10^{+3}$	**$4.20\times10^{+3}$**	$4.76\times10^{+3}$
f_{24}	$2.50\times10^{+2}$	**$2.50\times10^{+2}$**	$2.50\times10^{+2}$	$2.51\times10^{+2}$	$2.50\times10^{+2}$	$2.55\times10^{+2}$	$2.59\times10^{+2}$	$2.52\times10^{+2}$
f_{25}	**$2.59\times10^{+2}$**	$2.63\times10^{+2}$	$2.62\times10^{+2}$	$2.61\times10^{+2}$	$2.63\times10^{+2}$	$2.66\times10^{+2}$	$2.60\times10^{+2}$	$2.60\times10^{+2}$
f_{26}	$2.22\times10^{+2}$	$2.28\times10^{+2}$	$2.22\times10^{+2}$	$2.42\times10^{+2}$	$2.21\times10^{+2}$	$2.28\times10^{+2}$	**$2.00\times10^{+2}$**	$2.23\times10^{+2}$
f_{27}	$7.87\times10^{+2}$	$8.41\times10^{+2}$	$7.99\times10^{+2}$	$7.96\times10^{+2}$	$8.07\times10^{+2}$	$8.29\times10^{+2}$	**$6.75\times10^{+2}$**	$7.03\times10^{+2}$
f_{28}	**$3.00\times10^{+2}$**	**$3.00\times10^{+2}$**	$3.00\times10^{+2}$	$3.00\times10^{+2}$	**$3.00\times10^{+2}$**	**$3.00\times10^{+2}$**	$3.00\times10^{+2}$	**$3.00\times10^{+2}$**
Number of best results	3	6	4	3	1	3	12	2

Table A2 Parameters ft and $stag_{max}$ ($stag_{max}$=15, 20) investigation of DHLPSO.

Fun	$stag_{max}$ = 15				$stag_{max}$ = 20			
	ft = 0.55	ft = 0.6	ft = 0.65	ft = 0.7	ft = 0.55	ft = 0.6	ft = 0.65	ft = 0.7
	mean	mean	mean	mean	mean	mean	mean	mean
f_1	3.79×10^{-14}	$\mathbf{0.00\times10^{+0}}$	$\mathbf{0.00\times10^{+0}}$	$\mathbf{0.00\times10^{+0}}$	3.79×10^{-14}	$\mathbf{0.00\times10^{+0}}$	$\mathbf{0.00\times10^{+0}}$	$\mathbf{0.00\times10^{+0}}$
f_2	$9.46\times10^{+5}$	$8.73\times10^{+5}$	$1.08\times10^{+6}$	$9.71\times10^{+5}$	$8.35\times10^{+5}$	$7.80\times10^{+5}$	$1.02\times10^{+6}$	$9.64\times10^{+5}$
f_3	$2.28\times10^{+7}$	$2.65\times10^{+7}$	$2.13\times10^{+7}$	$2.92\times10^{+7}$	$2.51\times10^{+7}$	$2.05\times10^{+7}$	$2.29\times10^{+7}$	$2.85\times10^{+7}$
f_4	$9.79\times10^{+2}$	$1.23\times10^{+3}$	$1.25\times10^{+3}$	$1.52\times10^{+3}$	$\mathbf{8.00\times10^{+2}}$	$1.42\times10^{+3}$	$1.12\times10^{+3}$	$1.41\times10^{+3}$
f_5	2.31×10^{-13}	2.16×10^{-13}	1.82×10^{-13}	1.55×10^{-13}	2.16×10^{-13}	2.08×10^{-13}	1.82×10^{-13}	1.52×10^{-13}
f_6	$1.97\times10^{+1}$	$1.72\times10^{+1}$	$1.64\times10^{+1}$	$1.67\times10^{+1}$	$1.90\times10^{+1}$	$1.70\times10^{+1}$	$1.67\times10^{+1}$	$1.66\times10^{+1}$
f_7	$7.38\times10^{+0}$	$8.15\times10^{+0}$	$8.07\times10^{+0}$	$7.64\times10^{+0}$	$7.88\times10^{+0}$	$9.21\times10^{+0}$	$7.00\times10^{+0}$	$\mathbf{6.29\times10^{+0}}$
f_8	$2.09\times10^{+1}$	$2.09\times10^{+1}$	$2.09\times10^{+1}$	$2.09\times10^{+1}$	$2.09\times10^{+1}$	$2.09\times10^{+1}$	$2.09\times10^{+1}$	$2.09\times10^{+1}$
f_9	$\mathbf{2.69\times10^{+1}}$	$2.90\times10^{+1}$	$2.98\times10^{+1}$	$2.97\times10^{+1}$	$2.82\times10^{+1}$	$2.97\times10^{+1}$	$2.98\times10^{+1}$	$2.83\times10^{+1}$
f_{10}	1.51×10^{-1}	1.22×10^{-1}	1.27×10^{-1}	1.25×10^{-1}	1.33×10^{-1}	1.23×10^{-1}	1.26×10^{-1}	1.45×10^{-1}
f_{11}	1.33×10^{-1}	7.90×10^{-2}	2.93×10^{-1}	4.57×10^{-1}	$1.03\times10^{+0}$	$1.14\times10^{+0}$	8.28×10^{-1}	$1.52\times10^{+0}$
f_{12}	$6.94\times10^{+1}$	$6.29\times10^{+1}$	$5.74\times10^{+1}$	$7.63\times10^{+1}$	$5.97\times10^{+1}$	$5.78\times10^{+1}$	$6.33\times10^{+1}$	$\mathbf{5.62\times10^{+1}}$
f_{13}	$1.02\times10^{+2}$	$1.09\times10^{+2}$	$9.77\times10^{+1}$	$9.43\times10^{+1}$	$1.10\times10^{+2}$	$1.02\times10^{+2}$	$1.02\times10^{+2}$	$9.26\times10^{+1}$
f_{14}	$2.47\times10^{+1}$	$3.14\times10^{+1}$	$1.80\times10^{+1}$	$2.66\times10^{+1}$	$6.30\times10^{+1}$	$4.73\times10^{+1}$	$4.43\times10^{+1}$	$4.38\times10^{+1}$
f_{15}	$4.43\times10^{+3}$	$4.19\times10^{+3}$	$4.32\times10^{+3}$	$4.62\times10^{+3}$	$4.16\times10^{+3}$	$4.59\times10^{+3}$	$4.17\times10^{+3}$	$4.33\times10^{+3}$
f_{16}	$1.30\times10^{+0}$	$1.22\times10^{+0}$	$1.14\times10^{+0}$	$1.30\times10^{+0}$	$1.15\times10^{+0}$	$1.22\times10^{+0}$	$1.22\times10^{+0}$	$1.35\times10^{+0}$
f_{17}	$3.20\times10^{+1}$	$3.12\times10^{+1}$	$3.00\times10^{+1}$	$3.21\times10^{+1}$	$3.08\times10^{+1}$	$\mathbf{3.00\times10^{+1}}$	$3.10\times10^{+1}$	$3.18\times10^{+1}$
f_{18}	$8.72\times10^{+1}$	$7.66\times10^{+1}$	$8.43\times10^{+1}$	$8.24\times10^{+1}$	$7.54\times10^{+1}$	$7.82\times10^{+1}$	$8.87\times10^{+1}$	$9.10\times10^{+1}$
f_{19}	$1.55\times10^{+0}$	$1.67\times10^{+0}$	$1.55\times10^{+0}$	$1.63\times10^{+0}$	$1.70\times10^{+0}$	$1.77\times10^{+0}$	$1.84\times10^{+0}$	$1.84\times10^{+0}$
f_{20}	$1.12\times10^{+1}$	$1.13\times10^{+1}$	$1.13\times10^{+1}$	$1.12\times10^{+1}$	$1.11\times10^{+1}$	$1.13\times10^{+1}$	$1.12\times10^{+1}$	$1.11\times10^{+1}$
f_{21}	$2.81\times10^{+2}$	$2.83\times10^{+2}$	$2.73\times10^{+2}$	$2.81\times10^{+2}$	$2.95\times10^{+2}$	$2.77\times10^{+2}$	$2.83\times10^{+2}$	$2.71\times10^{+2}$
f_{22}	$1.04\times10^{+2}$	$1.08\times10^{+2}$	$1.08\times10^{+2}$	$1.14\times10^{+2}$	$1.72\times10^{+2}$	$1.29\times10^{+2}$	$1.23\times10^{+2}$	$1.24\times10^{+2}$
f_{23}	$4.85\times10^{+3}$	$4.49\times10^{+3}$	$4.42\times10^{+3}$	$4.69\times10^{+3}$	$4.48\times10^{+3}$	$4.67\times10^{+3}$	$4.55\times10^{+3}$	$4.38\times10^{+3}$
f_{24}	$2.51\times10^{+2}$	$2.50\times10^{+2}$	$2.54\times10^{+2}$	$2.54\times10^{+2}$	$2.50\times10^{+2}$	$2.51\times10^{+2}$	$2.51\times10^{+2}$	$2.50\times10^{+2}$
f_{25}	$2.61\times10^{+2}$	$2.65\times10^{+2}$	$2.64\times10^{+2}$	$2.64\times10^{+2}$	$2.60\times10^{+2}$	$2.62\times10^{+2}$	$2.62\times10^{+2}$	$2.61\times10^{+2}$
f_{26}	$2.16\times10^{+2}$	$2.21\times10^{+2}$	$2.14\times10^{+2}$	$2.12\times10^{+2}$	$2.10\times10^{+2}$	$2.20\times10^{+2}$	$2.23\times10^{+2}$	$2.04\times10^{+2}$
f_{27}	$8.68\times10^{+2}$	$8.34\times10^{+2}$	$7.16\times10^{+2}$	$8.15\times10^{+2}$	$7.83\times10^{+2}$	$7.86\times10^{+2}$	$7.86\times10^{+2}$	$7.01\times10^{+2}$
f_{28}	$\mathbf{3.00\times10^{+2}}$	$3.00\times10^{+2}$	$\mathbf{3.00\times10^{+2}}$	$\mathbf{3.00\times10^{+2}}$	$\mathbf{3.00\times10^{+2}}$	$3.00\times10^{+2}$	$\mathbf{3.00\times10^{+2}}$	$\mathbf{3.00\times10^{+2}}$
Number of best results	2	1	2	2	2	2	2	4

Table A3 Comparison results (D = 30).

Fun	Metric	SLPSO	CLPSO	XPSO	GLPSO	HCLPSO	BLPSO	BFLPSO	DHLPSO
f_1	Mean	1.38×10^{-13}	2.10×10^{-13}	2.01×10^{-13}	2.27×10^{-13}	2.45×10^{-13}	4.37×10^{-15}	7.13×10^{-14}	$\mathbf{0.00\times10^{+0}}$
	Rank	4	6	5	7	8	2	3	**1**
f_2	Mean	$1.25\times10^{+6}$	$1.91\times10^{+7}$	$5.26\times10^{+6}$	$\mathbf{5.95\times10^{+5}}$	$1.40\times10^{+6}$	$8.67\times10^{+6}$	$7.41\times10^{+6}$	$7.55\times10^{+5}$
	Rank	3	8	5	**1**	4	7	6	2
f_3	Mean	$2.87\times10^{+7}$	$3.03\times10^{+8}$	$1.98\times10^{+7}$	$7.21\times10^{+7}$	$2.61\times10^{+7}$	$\mathbf{9.14\times10^{+6}}$	$1.13\times10^{+7}$	$3.31\times10^{+7}$
	Rank	5	8	3	7	4	**1**	2	6
f_4	Mean	$1.05\times10^{+4}$	$2.09\times10^{+4}$	$\mathbf{5.61\times10^{+2}}$	$1.09\times10^{+3}$	$1.13\times10^{+3}$	$2.24\times10^{+3}$	$3.05\times10^{+3}$	$9.77\times10^{+2}$
	Rank	7	8	**1**	3	4	5	6	2
f_5	Mean	$\mathbf{1.14\times10^{-13}}$	2.23×10^{-13}	2.43×10^{-13}	5.31×10^{-13}	3.12×10^{-13}	1.36×10^{-13}	1.38×10^{-13}	1.14×10^{-13}
	Rank	**1**	5	6	8	7	3	4	1
f_6	Mean	$2.06\times10^{+1}$	$3.12\times10^{+1}$	$6.35\times10^{+1}$	$2.85\times10^{+1}$	$2.39\times10^{+1}$	$2.08\times10^{+1}$	$2.27\times10^{+1}$	$\mathbf{1.95\times10^{+1}}$
	Rank	2	7	8	6	5	3	4	**1**

(to be continued)

Table A3 Comparison results (D = 30). (continued)

Fun	Metric	SLPSO	CLPSO	XPSO	GLPSO	HCLPSO	BLPSO	BFLPSO	DHLPSO
f_7	Mean	**$4.43\times10^{+0}$**	$7.36\times10^{+1}$	$1.55\times10^{+1}$	$5.67\times10^{+1}$	$2.22\times10^{+1}$	$9.18\times10^{+0}$	$1.13\times10^{+1}$	$6.67\times10^{+0}$
	Rank	**1**	8	5	7	6	3	4	2
f_8	Mean	$2.10\times10^{+1}$	$2.09\times10^{+1}$	$2.09\times10^{+1}$	$2.09\times10^{+1}$	$2.09\times10^{+1}$	$2.09\times10^{+1}$	$2.09\times10^{+1}$	**$2.09\times10^{+1}$**
	Rank	8	6	5	3	2	7	4	**1**
f_9	Mean	**$9.55\times10^{+0}$**	$2.84\times10^{+1}$	$1.58\times10^{+1}$	$1.96\times10^{+1}$	$2.01\times10^{+1}$	$2.56\times10^{+1}$	$2.53\times10^{+1}$	$3.09\times10^{+1}$
	Rank	**1**	7	2	3	4	6	5	8
f_{10}	Mean	2.89×10^{-1}	$2.99\times10^{+0}$	**1.29×10^{-1}**	2.94×10^{-1}	2.29×10^{-1}	2.85×10^{-1}	2.85×10^{-1}	1.34×10^{-1}
	Rank	6	8	**1**	7	3	4	5	2
f_{11}	Mean	$1.65\times10^{+1}$	**5.57×10^{-14}**	$2.68\times10^{+1}$	3.90×10^{-2}	1.33×10^{-1}	5.55×10^{-1}	1.95×10^{-1}	1.02×10^{-13}
	Rank	7	**1**	8	3	4	6	5	2
f_{12}	Mean	$1.65\times10^{+2}$	$1.22\times10^{+2}$	$5.00\times10^{+1}$	$6.28\times10^{+1}$	$5.24\times10^{+1}$	$2.99\times10^{+1}$	**$2.70\times10^{+1}$**	$7.74\times10^{+1}$
	Rank	8	7	3	5	4	2	**1**	6
f_{13}	Mean	$1.64\times10^{+2}$	$1.58\times10^{+2}$	$1.17\times10^{+2}$	$1.32\times10^{+2}$	$1.16\times10^{+2}$	$5.00\times10^{+1}$	**$4.60\times10^{+1}$**	$8.26\times10^{+1}$
	Rank	8	7	5	6	4	2	**1**	3
f_{14}	Mean	$7.10\times10^{+2}$	$5.38\times10^{+0}$	$1.19\times10^{+3}$	**$2.22\times10^{+0}$**	$3.59\times10^{+1}$	$1.14\times10^{+2}$	$8.52\times10^{+1}$	$5.19\times10^{+0}$
	Rank	7	3	8	**1**	4	6	5	2
f_{15}	Mean	$5.92\times10^{+3}$	$4.79\times10^{+3}$	$4.06\times10^{+3}$	$3.81\times10^{+3}$	**$3.39\times10^{+3}$**	$3.63\times10^{+3}$	$3.43\times10^{+3}$	$3.90\times10^{+3}$
	Rank	8	7	6	4	**1**	3	2	5
f_{16}	Mean	$2.63\times10^{+0}$	$2.02\times10^{+0}$	$2.41\times10^{+0}$	7.84×10^{-1}	$1.20\times10^{+0}$	$1.75\times10^{+0}$	$1.58\times10^{+0}$	**7.43×10^{-1}**
	Rank	8	6	7	2	3	5	4	**1**
f_{17}	Mean	$1.78\times10^{+2}$	$3.08\times10^{+1}$	$6.92\times10^{+1}$	$3.19\times10^{+1}$	$3.08\times10^{+1}$	**$3.04\times10^{+1}$**	$3.09\times10^{+1}$	$3.14\times10^{+1}$
	Rank	8	2	7	6	3	**1**	4	5
f_{18}	Mean	$1.99\times10^{+2}$	$1.99\times10^{+2}$	$1.79\times10^{+2}$	**$7.18\times10^{+1}$**	$7.86\times10^{+1}$	$1.01\times10^{+2}$	$9.61\times10^{+1}$	$9.56\times10^{+1}$
	Rank	8	7	6	**1**	2	5	4	3
f_{19}	Mean	$3.58\times10^{+0}$	**5.81×10^{-1}**	$3.34\times10^{+0}$	$1.78\times10^{+0}$	$1.52\times10^{+0}$	$1.67\times10^{+0}$	$1.55\times10^{+0}$	$1.47\times10^{+0}$
	Rank	8	**1**	7	6	3	5	4	2
f_{20}	Mean	$1.39\times10^{+1}$	$1.36\times10^{+1}$	$1.17\times10^{+1}$	$1.10\times10^{+1}$	$1.07\times10^{+1}$	$1.05\times10^{+1}$	**$1.01\times10^{+1}$**	$1.04\times10^{+1}$
	Rank	8	7	6	5	4	3	**1**	2
f_{21}	Mean	$2.91\times10^{+2}$	**$2.53\times10^{+2}$**	$3.19\times10^{+2}$	$3.40\times10^{+2}$	$2.56\times10^{+2}$	$3.04\times10^{+2}$	$3.06\times10^{+2}$	$3.00\times10^{+2}$
	Rank	3	**1**	7	8	2	5	6	4
f_{22}	Mean	$6.41\times10^{+2}$	$1.01\times10^{+2}$	$1.19\times10^{+3}$	$1.14\times10^{+2}$	$1.14\times10^{+2}$	$1.57\times10^{+2}$	$1.60\times10^{+2}$	**$6.46\times10^{+1}$**
	Rank	7	2	8	3	4	5	6	**1**
f_{23}	Mean	$4.93\times10^{+3}$	$5.57\times10^{+3}$	$3.88\times10^{+3}$	$3.92\times10^{+3}$	$4.06\times10^{+3}$	$3.72\times10^{+3}$	**$3.47\times10^{+3}$**	$4.20\times10^{+3}$
	Rank	7	8	3	4	5	2	**1**	6
f_{24}	Mean	$2.21\times10^{+2}$	$2.74\times10^{+2}$	$2.49\times10^{+2}$	$2.50\times10^{+2}$	$2.29\times10^{+2}$	**$2.14\times10^{+2}$**	$2.16\times10^{+2}$	$2.59\times10^{+2}$
	Rank	3	8	5	6	4	**1**	2	7
f_{25}	Mean	**$2.54\times10^{+2}$**	$2.96\times10^{+2}$	$2.81\times10^{+2}$	$2.72\times10^{+2}$	$2.65\times10^{+2}$	$2.68\times10^{+2}$	$2.59\times10^{+2}$	$2.60\times10^{+2}$
	Rank	**1**	8	7	6	4	5	2	3
f_{26}	Mean	$2.55\times10^{+2}$	$2.02\times10^{+2}$	$2.86\times10^{+2}$	$2.45\times10^{+2}$	$2.03\times10^{+2}$	$2.03\times10^{+2}$	$2.11\times10^{+2}$	**$2.00\times10^{+2}$**
	Rank	7	2	8	6	3	4	5	**1**
f_{27}	Mean	**$4.69\times10^{+2}$**	$7.29\times10^{+2}$	$7.07\times10^{+2}$	$7.59\times10^{+2}$	$5.85\times10^{+2}$	$5.42\times10^{+2}$	$5.11\times10^{+2}$	$6.75\times10^{+2}$
	Rank	**1**	7	6	8	4	3	2	5
f_{28}	Mean	$3.20\times10^{+2}$	$3.00\times10^{+2}$	$4.13\times10^{+2}$	**$2.80\times10^{+2}$**	$2.96\times10^{+2}$	$2.94\times10^{+2}$	$3.00\times10^{+2}$	$3.00\times10^{+2}$
	Rank	7	6	8	**1**	3	2	4	5
Count		5	3	2	4	1	3	4	7
Ave		5.43	5.75	5.57	4.75	3.86	3.79	3.64	3.18
Total		6	8	7	5	4	3	2	1

Table A4 Comparison results (D = 50).

Fun	Metric	SLPSO	CLPSO	XPSO	GLPSO	HCLPSO	BLPSO	BFLPSO	DHLPSO
f_1	Mean	2.27×10^{-13}	2.27×10^{-13}	3.03×10^{-13}	9.02×10^{-13}	5.15×10^{-13}	2.42×10^{-13}	2.27×10^{-13}	**2.27×10^{-13}**
	Rank	2	2	6	8	7	5	2	**1**
f_2	Mean	$1.85\times10^{+6}$	$3.89\times10^{+7}$	$1.73\times10^{+7}$	$2.21\times10^{+6}$	**$1.60\times10^{+6}$**	$1.69\times10^{+7}$	$1.32\times10^{+7}$	$2.13\times10^{+6}$
	Rank	2	8	7	4	**1**	6	5	3
f_3	Mean	$1.78\times10^{+7}$	$2.39\times10^{+9}$	$7.91\times10^{+7}$	$3.25\times10^{+8}$	$2.74\times10^{+8}$	$2.70\times10^{+7}$	$2.93\times10^{+7}$	**$1.13\times10^{+7}$**
	Rank	2	8	5	7	6	3	4	**1**
f_4	Mean	$4.19\times10^{+4}$	$2.34\times10^{+4}$	**$6.04\times10^{+2}$**	$6.42\times10^{+3}$	$1.34\times10^{+3}$	$2.83\times10^{+3}$	$2.47\times10^{+3}$	$1.88\times10^{+3}$
	Rank	8	7	**1**	6	2	5	4	3
f_5	Mean	3.12×10^{-13}	3.28×10^{-13}	6.25×10^{-13}	1.42×10^{-12}	7.24×10^{-13}	3.26×10^{-13}	3.41×10^{-13}	**3.11×10^{-13}**
	Rank	2	4	6	8	7	3	5	**1**
f_6	Mean	$4.40\times10^{+1}$	$4.67\times10^{+1}$	$4.92\times10^{+1}$	$6.21\times10^{+1}$	$4.44\times10^{+1}$	$4.43\times10^{+1}$	$4.60\times10^{+1}$	**$4.35\times10^{+1}$**
	Rank	2	6	7	8	4	3	5	**1**
f_7	Mean	$5.18\times10^{+1}$	$1.04\times10^{+2}$	$3.27\times10^{+1}$	$6.64\times10^{+1}$	$4.09\times10^{+1}$	$2.21\times10^{+1}$	$2.30\times10^{+1}$	**$2.16\times10^{+1}$**
	Rank	6	8	4	7	5	2	3	**1**
f_8	Mean	$2.12\times10^{+1}$	$2.11\times10^{+1}$	$2.11\times10^{+1}$	$2.11\times10^{+1}$	$2.11\times10^{+1}$	$2.11\times10^{+1}$	$2.11\times10^{+1}$	**$2.11\times10^{+1}$**
	Rank	8	3	2	4	6	5	7	**1**
f_9	Mean	**$1.72\times10^{+1}$**	$5.47\times10^{+1}$	$3.33\times10^{+1}$	$3.87\times10^{+1}$	$4.01\times10^{+1}$	$5.03\times10^{+1}$	$5.03\times10^{+1}$	$6.30\times10^{+1}$
	Rank	**1**	7	2	3	4	5	6	8
f_{10}	Mean	2.52×10^{-1}	$7.35\times10^{+0}$	**1.09×10^{-1}**	2.28×10^{-1}	2.16×10^{-1}	3.01×10^{-1}	2.80×10^{-1}	2.11×10^{-1}
	Rank	5	8	**1**	4	3	7	6	2
f_{11}	Mean	$3.73\times10^{+1}$	**7.69×10^{-14}**	$6.72\times10^{+1}$	4.98×10^{-1}	$1.07\times10^{+0}$	$2.63\times10^{+0}$	9.56×10^{-1}	2.65×10^{-1}
	Rank	7	**1**	8	3	5	6	4	2
f_{12}	Mean	$3.47\times10^{+2}$	$3.02\times10^{+2}$	$1.24\times10^{+2}$	$1.45\times10^{+2}$	$1.10\times10^{+2}$	$6.71\times10^{+1}$	**$5.61\times10^{+1}$**	$1.61\times10^{+2}$
	Rank	8	7	4	5	3	2	**1**	6
f_{13}	Mean	$3.47\times10^{+2}$	$3.70\times10^{+2}$	$2.18\times10^{+2}$	$2.93\times10^{+2}$	$2.23\times10^{+2}$	$1.20\times10^{+2}$	**$1.13\times10^{+2}$**	$2.04\times10^{+2}$
	Rank	7	8	4	6	5	2	**1**	3
f_{14}	Mean	$1.24\times10^{+3}$	**$1.93\times10^{+0}$**	$2.24\times10^{+3}$	$1.30\times10^{+1}$	$8.63\times10^{+1}$	$2.76\times10^{+2}$	$2.67\times10^{+2}$	$2.54\times10^{+1}$
	Rank	7	**1**	8	2	4	6	5	3
f_{15}	Mean	$1.31\times10^{+4}$	$9.22\times10^{+3}$	$9.90\times10^{+3}$	$7.32\times10^{+3}$	$7.43\times10^{+3}$	$7.34\times10^{+3}$	**$7.21\times10^{+3}$**	$8.54\times10^{+3}$
	Rank	8	6	7	2	4	3	**1**	5
f_{16}	Mean	$3.49\times10^{+0}$	$2.71\times10^{+0}$	$3.41\times10^{+0}$	**$1.13\times10^{+0}$**	$2.28\times10^{+0}$	$2.12\times10^{+0}$	$2.07\times10^{+0}$	$1.79\times10^{+0}$
	Rank	8	6	7	**1**	5	4	3	2
f_{17}	Mean	$3.82\times10^{+2}$	$5.16\times10^{+1}$	$1.44\times10^{+2}$	$5.75\times10^{+1}$	$5.29\times10^{+1}$	**$5.03\times10^{+1}$**	$5.20\times10^{+1}$	$5.24\times10^{+1}$
	Rank	8	2	7	6	5	**1**	3	4
f_{18}	Mean	$3.98\times10^{+2}$	$4.08\times10^{+2}$	$3.48\times10^{+2}$	**$1.45\times10^{+2}$**	$1.56\times10^{+2}$	$1.80\times10^{+2}$	$1.74\times10^{+2}$	$1.78\times10^{+2}$
	Rank	7	8	6	**1**	2	5	3	4
f_{19}	Mean	$9.46\times10^{+0}$	**$1.52\times10^{+0}$**	$7.93\times10^{+0}$	$3.79\times10^{+0}$	$2.35\times10^{+0}$	$2.92\times10^{+0}$	$2.87\times10^{+0}$	$2.20\times10^{+0}$
	Rank	8	**1**	7	6	3	5	4	2
f_{20}	Mean	$2.23\times10^{+1}$	$2.30\times10^{+1}$	$2.07\times10^{+1}$	$2.00\times10^{+1}$	$2.00\times10^{+1}$	**$1.87\times10^{+1}$**	$1.87\times10^{+1}$	$2.03\times10^{+1}$
	Rank	7	8	6	4	3	**1**	2	5
f_{21}	Mean	$7.94\times10^{+2}$	**$3.85\times10^{+2}$**	$8.63\times10^{+2}$	$9.08\times10^{+2}$	$5.05\times10^{+2}$	$9.30\times10^{+2}$	$9.51\times10^{+2}$	$5.10\times10^{+2}$
	Rank	4	**1**	5	6	2	7	8	3
f_{22}	Mean	$1.28\times10^{+3}$	**$3.75\times10^{+1}$**	$2.76\times10^{+3}$	$5.87\times10^{+1}$	$1.49\times10^{+2}$	$2.81\times10^{+2}$	$2.12\times10^{+2}$	$6.70\times10^{+1}$
	Rank	7	**1**	8	2	4	6	5	3
f_{23}	Mean	$1.29\times10^{+4}$	$1.13\times10^{+4}$	$7.88\times10^{+3}$	$8.00\times10^{+3}$	$8.20\times10^{+3}$	$7.57\times10^{+3}$	**$7.29\times10^{+3}$**	$9.11\times10^{+3}$
	Rank	8	7	3	4	5	2	**1**	6
f_{24}	Mean	$2.50\times10^{+2}$	$3.49\times10^{+2}$	$3.11\times10^{+2}$	$3.02\times10^{+2}$	$2.72\times10^{+2}$	**$2.42\times10^{+2}$**	$2.51\times10^{+2}$	$2.86\times10^{+2}$
	Rank	2	8	7	6	4	**1**	3	5

(to be continued)

Table A4 Comparison results (D = 50). (continued)

Fun	Metric	SLPSO	CLPSO	XPSO	GLPSO	HCLPSO	BLPSO	BFLPSO	DHLPSO
f_{25}	Mean	**$2.95\times10^{+2}$**	$3.88\times10^{+2}$	$3.60\times10^{+2}$	$3.37\times10^{+2}$	$3.55\times10^{+2}$	$3.32\times10^{+2}$	$3.22\times10^{+2}$	$3.19\times10^{+2}$
	Rank	**1**	8	7	5	6	4	3	2
f_{26}	Mean	$3.39\times10^{+2}$	$2.04\times10^{+2}$	$3.85\times10^{+2}$	$3.12\times10^{+2}$	$2.04\times10^{+2}$	$2.46\times10^{+2}$	$2.14\times10^{+2}$	**$2.02\times10^{+2}$**
	Rank	7	3	8	6	2	5	4	**1**
f_{27}	Mean	$8.13\times10^{+2}$	$1.61\times10^{+3}$	$1.28\times10^{+3}$	$1.34\times10^{+3}$	$1.20\times10^{+3}$	$1.25\times10^{+3}$	$9.91\times10^{+2}$	**$4.15\times10^{+2}$**
	Rank	2	8	6	7	4	5	3	**1**
f_{28}	Mean	$7.52\times10^{+2}$	$4.00\times10^{+2}$	$8.18\times10^{+2}$	$6.13\times10^{+2}$	$4.00\times10^{+2}$	**$3.87\times10^{+2}$**	$4.00\times10^{+2}$	$9.06\times10^{+2}$
	Rank	6	4	7	5	3	**1**	2	8
Count		2	5	2	2	1	4	4	8
Ave		5.36	5.32	5.57	4.86	4.07	3.93	3.68	3.11
Total		7	6	8	5	4	3	2	1

References

[1] A. Slowik and H. Kwasnicka, Evolutionary algorithms and their applications to engineering problems, *Neural Computing and Applications*, vol. 32, no. 16, pp. 12363–12379, 2020.

[2] E. H. Houssein, A. G. Gad, K. Hussain, and P. N. Suganthan, Major advances in particle swarm optimization: Theory, analysis, and application, *Swarm and Evolutionary Computation*, vol. 63, p. 100868, 2021.

[3] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. Cambridge, MA, USA: MIT Press, 1992.

[4] G. D'Angelo and F. Palmieri, GGA: A modified genetic algorithm with gradient-based local search for solving constrained optimization problems, *Information Sciences*, vol. 547, pp. 136–162, 2021.

[5] D. Karaboga, An idea based on honey bee swarm for numerical optimization, Tech. Rep. TR06, Erciyes University, Kayseri, Turkey, 2005.

[6] Bilal, M. Pant, H. Zaheer, L. Garcia-Hernandez, and A. Abraham, Differential evolution: A review of more than two decades of research, *Engineering Applications of Artificial Intelligence*, vol. 90, p. 103479, 2020.

[7] W. Li, X. Meng, and Y. Huang, Fitness distance correlation and mixed search strategy for differential evolution, *Neurocomputing*, vol. 458, pp. 514–525, 2021.

[8] W. Li, W. Li, and Y. Huang, Enhancing firefly algorithm with dual-population topology coevolution, *Mathematics*, vol. 10, no. 9, p. 1564, 2022.

[9] M. Jamil and X. -S. Yang, A literature survey of benchmark functions for global optimization problems, arXiv preprint arXiv: 1308.4008, 2013.

[10] J. Kennedy and R. Eberhart, Particle swarm optimization, in *Proc. ICNN'95 - International Conference on Neural Networks*, Perth, Australia, 1995, pp. 1942–1948.

[11] J. Kennedy, Swarm intelligence, in *Handbook of Nature-Inspired and Innovative Computing*, A. Y. Zomaya, ed. New York, NY, USA: Springer, 2006, pp. 187–219.

[12] H. Liu, X. -W. Zhang, and L. -P. Tu, A modified particle swarm optimization using adaptive strategy, *Expert Systems with Applications*, vol. 152, p. 113353, 2020.

[13] M. Taherkhani and R. Safabakhsh, A novel stability-based adaptive inertia weight for particle swarm optimization, *Applied Soft Computing*, vol. 38, pp. 281–295, 2016.

[14] M. U. Farooq, A. Ahmad, and A. Hameed, Opposition-based initialization and a modified pattern for inertia weight (IW) in PSO, in *Proc. 2017 IEEE International Conference on Innovations in Intelligent SysTems and Applications* (*INISTA*), Gdynia, Poland, 2017, pp. 96–101.

[15] I. K. Gupta, A. Choubey, and S. Choubey, Particle swarm optimization with selective multiple inertia weights, in *Proc. 2017 8th International Conference on Computing, Communication and Networking Technologies* (*ICCCNT*), Delhi, India, 2017, pp. 1–6.

[16] M. R. Tanweer, S. Suresh, and N. Sundararajan, Dynamic mentoring and self-regulation based particle swarm optimization algorithm for solving complex real-world optimization problems, *Information Sciences*, vol. 326, pp. 1–24, 2016.

[17] W. Li, B. Sun, Y. Huang, and S. Mahmoodi, Adaptive complex network topology with fitness distance correlation framework for particle swarm optimization, *International Journal of Intelligent Systems*, vol. 37, no. 8, pp. 5217–5247, 2022.

[18] X. Li, Niching without niching parameters: Particle swarm optimization using a ring topology, *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 1, pp. 150–169, 2009.

[19] X. Xia, L. Gui, and Z. -H. Zhan, A multi-swarm particle swarm optimization algorithm based on dynamical topology and purposeful detecting, *Applied Soft Computing*, vol. 67, pp. 126–140, 2018.

[20] J. J. Liang and P. N. Suganthan, Dynamic multi-swarm particle swarm optimizer with local search, in *Proc. 2005 IEEE Congress on Evolutionary Computation*, Edinburgh, UK, 2005, pp. 522–528.

[21] X. Tao, W. Guo, X. Li, Q. He, R. Liu, and J. Zou, Fitness peak clustering based dynamic multi-swarm particle swarm optimization with enhanced learning strategy, *Expert Systems with Applications*, vol. 191, p. 116301, 2022.

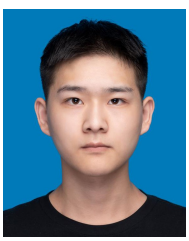
[22] R. Cheng and Y. Jin, A social learning particle swarm optimization algorithm for scalable optimization, *Information Sciences*, vol. 291, pp. 43–60, 2015.

[23] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3,

pp. 281–295, 2006.
[24] B. Liang, Y. Zhao, and Y. Li, A hybrid particle swarm optimization with crisscross learning strategy, *Engineering Applications of Artificial Intelligence*, vol. 105, p. 104418, 2021.
[25] Z. H. Zhan and J. Zhang, Orthogonal learning particle swarm optimization for power electronic circuit optimization with free search range, in *Proc. 2011 IEEE Congress of Evolutionary Computation* (*CEC*), New Orleans, LA, USA, 2011, pp. 2563–2570.
[26] X. Xia, L. Gui, G. He, B. Wei, Y. Zhang, F. Yu, H. Wu, and Z. -H. Zhan, An expanded particle swarm optimization based on multi-exemplar and forgetting ability, *Information Sciences*, vol. 508, pp. 105–120, 2020.
[27] Y. J. Gong, J. J. Li, Y. Zhou, Y. Li, H. S. H. Chung, Y. H. Shi, and J. Zhang, Genetic learning particle swarm optimization, *IEEE Transactions on Cybernetics*, vol. 46, no. 10, pp. 2277–2290, 2015.
[28] X. Chen, H. Tianfield, and W. Du, Bee-foraging learning particle swarm optimization, *Applied Soft Computing*, vol. 102, p. 107134, 2021.
[29] Z. Li, W. Wang, Y. Yan, and Z. Li, PS–ABC: A hybrid algorithm based on particle swarm and artificial bee colony for high-dimensional optimization problems, *Expert Systems with Applications*, vol. 42, no. 22, pp. 8881–8895, 2015.
[30] J. B. Fisher and R. A. Hinde, Opening of milk bottles by birds, *Brit. Birds*, vol. 42, pp. 347–357, 1949.
[31] W. -Y. Chiu, G. G. Yen, and T. -K. Juan, Minimum Manhattan distance approach to multiple criteria decision making in multiobjective optimization problems, *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 6, pp. 972–985, 2016.
[32] J. J. Liang, B. Qu, P. N. Suganthan, and A. G. Hernández-Díaz, Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization, Tech. Rep. 201212, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, 2013.
[33] T. Li, J. Shi, W. Deng, and Z. Hu, Pyramid particle swarm optimization with novel strategies of competition and cooperation, *Applied Soft Computing*, vol. 121, p. 108731, 2022.
[34] S. M. A. Pahnehkolaei, A. Alfi, and J. T. Machado, Analytical stability analysis of the fractional-order particle swarm optimization algorithm, *Chaos, Solitons & Fractals*, vol. 155, p. 111658, 2022.
[35] N. Lynn and P. N. Suganthan, Heterogeneous comprehensive learning particle swarm optimization with enhanced exploration and exploitation, *Swarm and Evolutionary Computation*, vol. 24, pp. 11–24, 2015.
[36] X. Chen, H. Tianfield, C. Mei, W. Du, and G. Liu, Biogeography-based learning particle swarm optimization, *Soft Computing*, vol. 21, no. 24, pp. 7519–7541, 2017.
[37] J. Derrac, S. García, D. Molina, and F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 3–18, 2011.

Yangtao Chen received the BEng degree from Jiangxi University of Science and Technology in 2020. He is currently pursuing the master degree at the School of Information Engineering, Jiangxi University of Science and Technology. His main research interest is particle swarm optimization algorithm and multi-objective evolutionary algorithm.

Qian Cai is an associate professor at the School of Information Engineering, Jiangxi University of Science and Technology, Ganzhou, China. He received the master degree in communication and information system from Huazhong University of Science and Technology in 2007, and the BEng degree in computer science and technology from Nanchang University, Nanchang, China in 2003. His current research interests include artificial intelligence, software engineer, and evolutionary optimization.



Cancan Wang received the BEng degree from Jiangxi University of Science and Technology in 2020. He is currently pursuing the master degree at the School of Information Engineering, Jiangxi University of Science and Technology. His main research interest is ant colony optimization algorithm.

Wei Li received the PhD degree from South China Agricultural University in 2018, the MEng degree in computer application technology from Jiangxi University of Science and Technology in 2008, and the BEng degree in computer science and technology from Jiangxi University of Science and Technology in 2003. He is currently an associate professor at the School of Information Engineering, Jiangxi University of Science and Technology. He has over 30 papers published in fully refereed international journals and conferences and has served as the program chair or program committee member in many international conferences. His research focuses mainly on computational intelligence, evolutionary optimization, fitness landscape analysis, and large-scale optimization.

Ying Huang received the PhD degree in computer software and theory from Wuhan University in 2016. She received the MSc degree in computer application technology from Jiangxi University of Science and Technology in 2008. Currently, she works at Gannan Normal University as an associate professor. She has published over 20 research papers in the international conferences and journals. Her research interests include the areas of service computing, business process, and intelligent optimization algorithms.

Soroosh Mahmoodi received the PhD degree from Xiamen University in 2018, the MSc degree in mechanical engineering from Islamic Azad University of Science and Technology, Takestan Branch, Iran in 2011, and the BSc degree in mechanic engineering from Semnan University of Science and Technology, Iran in 2005. He is a development manager of LED lamps producer, Abyek Industrial Zone, Qazvin, Iran. He has published more 4 SCI high-level academic papers in authoritative journals and four international conferences papers. His research focuses on mechanical engineering, electrical engineering, automation and control engineering, chemical and material engineering, and optics, physics, and computer science engineering.