Hindawi
Mathematical Problems in Engineering
Volume 2019, Article ID 8164083, 22 pages
https://doi.org/10.1155/2019/8164083

Research Article

Improved Adaptive Holonic Particle Swarm Optimization

Hao Li, Hongbin Jin, Hanzhong Wang, and Yanyan Ma

Intelligence Department, Air Force Early-Warning Academy, Wuhan 430019, China

Correspondence should be addressed to Hao Li; afeu_li@163.com

Received 2 July 2019; Revised 19 October 2019; Accepted 20 November 2019; Published 12 December 2019

Academic Editor: Oliver Schütze

For the first time , the Holonic Particle Swarm Optimization (HPSO) algorithm applies multiagent theory about the improvement in the PSO algorithm and achieved good results. In order to further improve the performance of the algorithm, this paper proposes an improved Adaptive Holonic Particle Swarm Optimization (AHPSO) algorithm. Firstly, a brief review of the HPSO algorithm is carried out, and the HPSO algorithm can be further studied in three aspects: grouping strategy, iteration number setting, and state switching discrimination. The HPSO algorithm uses an approximately uniform grouping strategy that is the simplest but does not consider the connections between particles. And if the particles with larger or smaller differences are grouped together in different search stages, the search efficiency will be improved. Therefore, this paper proposes a grouping strategy based on information entropy and system clustering and combines two grouping strategies with corresponding search methods. The performance of the HPSO algorithm depends on the setting of the number of iterations. If it is too small, it is difficult to search for the optimal and it wastes so many computing resources. Therefore, this paper constructs an adaptive termination condition that causes the particles to terminate spontaneously after convergence. The HPSO algorithm only performs a conversion from extensive search to exact search and still has the potential to fall into local optimum. This paper proposes a state switching condition to improve the probability that the algorithm jumps out of the local optimum. Finally, AHPSO and HPSO are compared by using 22 groups of standard test functions. AHPSO is faster in convergence than HPSO, and the number of iterations of AHPSO convergence is employed in HPSO. At this point, there exists a large gap between HPSO and the optimal solution, i.e., AHPSO can have better algorithm efficiency without setting the number of iterations.

1. Introduction

Particle swarm optimization (PSO) has been successfully applied to many optimization problems. Due to its particular superiority in tackling the nonconvex, nondifferentiable, and even not well-defined problems, evolutionary algorithms have attracted increasing interests and achieved good development in both academia and industry in the past several decades [1–7]. PSO algorithm is conducive to machine optimization [8, 9], energy management [10], the accurate prediction and optimization of cogging torque [11], and other engineering aspects. Besides, it also helps to solve minimum set covering problems [12], the obnoxious p median problems [13], and other academic problems. Yet, PSO is premature and prone to local optimization. Scholars have conducted in-depth research on this, and they have achieved fruitful results.

The major research directions are stratified into three types. The first is to dynamically adjust the parameters of PSO search to change the search strategy in line with stages [14]. Since the PSO algorithm's parameters affect the performance of the algorithm, Hu et al. [15] studied the automatic control mechanism to ensure that better parameters are obtained during each iteration. Yet, such method can be calculated only for differentiable functions, which limits its application scope. Bai et al. [16] focused on the study of feasible search space that should be solved urgently to reduce the search space of PSO algorithm so as to accelerate the solving process of multiobjective optimization problem. This method is conducive to solving high dimensional complex problems, but it needs to be further analyzed and modeled. This method can be used only after the constraint conditions are transformed, and the improvement degree of PSO algorithm is limited.

Pham et al. [17] built the low-discrepancy sequence of initialized particle swarm optimization with high-order nonlinear time-varying inertia weight (LHNPSO) algorithm to achieve the segmentation of magnetic resonance imaging. Yan et al. [18] introduced a request and reset strategy to reset low-fit particles, making the reset particles form the best group around the optimal particles, improving search results. Yet, such algorithm will increase the probability of falling into local optimum during the optimization. Jordehi [19] constructed time-varying weight coefficient and adaptive adjusted particle search strategy, which provides feasible plan for estimating circuit model parameters of PV cells/modules. Chen et al. [20] employed fitness function to dynamically and adaptively adjust particle weight and effectively improved the search mode of particles at different times. However, the order of magnitude and variation of fitness function may vary with problems. Such a method needs further improvement.

The second is to combine the PSO algorithm with other heuristic algorithms to supplement each other. Ghorbani et al. [21] built GA-PSO algorithm by incorporating PSO and Genetic Algorithm (GA) to optimize the sizing of an off-grid house with photovoltaic panels, wind turbines, and battery. Accordingly, the favorable results are obtained. This method stratifies the population into two groups to perform PSO and GA operations, respectively. In each iteration, it improves the best position by comparing the optimal results of the PSO algorithm and GA. In fact, each step of the algorithm is running independently, and the results are always those of GA modified PSO. If the performance function of GA is constantly poor, GA has no effect and consumes numerous computing resources. The optimization algorithm is virtually the only PSO processing.

Goudarzi et al. [22] and Ran and Mesut [23] incorporated artificial bee colony (ABC) and PSO, thereby improving the performance of PSO algorithm. Gupta and Saini [24] and Liu et al. [25] used fuzzy theory to improve PSO algorithm, which effectively solved the problems to be solved. This method should set the fuzzy function, whereas the fuzzy function may vary with optimization problem, which impacts the algorithm performance. Jiang et al. [26] employed wavelet mutation to reduce the probability of PSO falling into local optimum, whereas different wavelet bases were selected for different problems, which may result in the difference in performance. This algorithm can be further improved. Chen et al. [27] initialized the group using opposition-based learning (OBL), increased the population diversity, and reduced the possibility of PSO particles to fall into local optimum using the properties of ergodicity, nonrepetition, and irregularity of SCA.

The third is to improve the search performance by changing PSO's particle topology. Issa et al. [28] conducted the ground search in accordance with the concept of hierarchy and the favorable exploitation of prominent regions of sine cosine optimization (SCA) so as to improve the accuracy of search. It conducted the top search using the excellent exploration of the search space of PSO to increase the population diversity. Through these efforts, the Adaptive SCA-PSO algorithm is established, thereby balancing exploration and exploitation and gaining favorable results. Because of the randomness of the initial population, the grouping strategy will inevitably affect the efficiency of the algorithm. This problem is not the focus of literature [28], and there is less discussion on the updating method of parameters in SCA. Thus, such an updating method has more space for further study.

Based on orthogonal experimental design, Zhan et al. [29–31] increased particle diversity and search rate. Yet, this method holds only when no or weak interaction among the factors exists, and the orthogonality may not be likely to achieve in complex problems. In this regard, Al-Bahrani and Patra [32] made great efforts to solve the above problems by building a novel orthogonal PSO algorithm based on orthogonal diagonalization. Besides, 30 standard functions are tested to verify their performance. However, the discussion and its proof process in literature [32] suggest that it is necessary to ensure that all vectors in the d-dimensional search space are linearly independent, thus limiting the applicable scope of the algorithm. P. Liu and J. Liu [33] and Tan et al. [34] employed multileader methods to improve the topological structure of particles, thereby improving the algorithm efficiency. Thus, this literature effectively guides the subsequent research.

Scholars have conducted in-depth research on this problem from different perspectives. Among the method of changing the topology, literature [1] referenced Holonic Particle Swarm Optimization (HPSO). Multiagent systems have now achieved widespread success in many fields but have never been applied to improved PSO algorithms. Because the PSO algorithm particles themselves do not have the ability to learn and evolve autonomously, they are not agents. And if you increase the intelligence level of PSO particles, you will sacrifice the speed of the PSO algorithm. Literature [1] improved the PSO algorithm by holonic approach and improved the topology of particles. For the first time, multiagent theory was applied to the improved PSO algorithm. By designing the hierarchical way, the ability to globally search the particles is improved according to the interaction between the levels and different levels of information. The constraint on the problem is reduced, and there is no explicit limit to the objective function, which broadens the applicability of the algorithm. And the performance of the algorithm has been greatly improved.

The method of the literature [1] has three points that can be improved, namely, the grouping mode, the search state switching condition, and the setting of the number of iterations.

(1) The literature [1] uses a random grouping method. This paper proposes a grouping strategy based on information entropy and system clustering and combines two grouping strategies with corresponding search methods. This method improved search performance.

(2) The search method of the literature [1] is to directly convert to the precise search after performing a fixed number of range searches. Based on the fitness function gradient, this paper constructs adaptive

switching and reverse switching strategies to achieve adaptive mutual conversion between range search and precise search. The method is more reasonable.

(3) The algorithm of [1] is to set the number of iterations. It is very likely that the algorithm has converged and the algorithm is still running. This paper constructs an adaptive termination condition, which causes the particles to terminate spontaneously after convergence, saving computational resources and time.

Structure of this study is organized as follows: in Section 2, the main work of literature [1] is briefly reviewed; in Section 3, the places that can be improved in literature [1] are discussed; in Section 4, a novel grouping strategy is established; Section 7 verifies the effectiveness of the proposed method through simulation and compares the proposed method with mainstream method to highlight its advantages; and Section 8 draws a conclusion and summarizes the whole paper.

2. Review of Holonic Particle Swarm Optimization

This section begins with a brief review of the holonic structure of [1]. The principle of the HPSO algorithm is then discussed. Finally, the architecture and process of HPSO algorithm are introduced.

2.1. Holonic Structure. Multiagent system was used because the agent has the ability to learn and evolve and can emerge the ability that the monomer does not have through communication, cooperation, competition, etc., that is, realize group intelligence. This system has become a research hotspot in today's academic circles and has been widely used in many scientific fields. At present, there are two main ways to improve the capabilities of multiagent systems. One is to study the ability to improve the acquisition, processing, and feedback of individual devices. The other is to build and improve the topology of the system from the perspective of the system and improve the ability of system information transmission and interaction, thereby improving the overall efficiency of the system. The holonic structure is a special topology. By affecting the role, interaction, and organizational behavior of the agent system, the original simple system can generate complex capabilities and achieve more powerful functions.

Holonic approach is illustrated in Figure 1.

Figure 1 shows a three-layer holonic structure consisting of 11 agents. On the Level 1 layer, 11 agents are divided into three groups, and each group of agents only exchanges information within the group. Through information exchange, the group forms a virtual agent on the Level 2 layer. The virtual agent grasps all the information of the agent in the corresponding group of the Level 1 layer and obtains the optimal information of the group. Similarly, the three virtual agents of the Level 2 layer form a new group and interact with each other, thus forming a global agent at the Level 3 layer and grasping all the agent information.

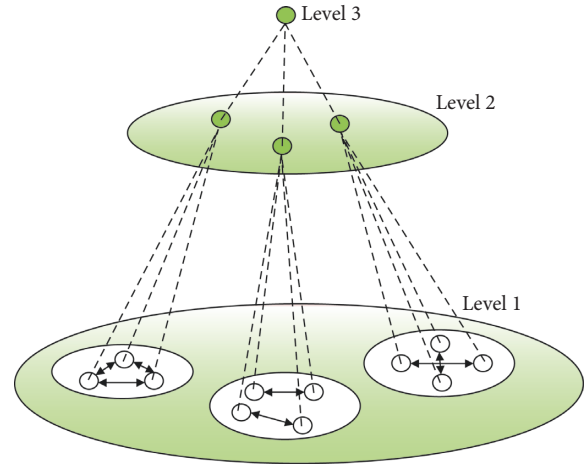

Figure 1: A holonic structure with three levels.

If we do not build a holonic structure, then 11 basic agents' information are needed to perform real-time information interaction, which requires each agent to have strong information transmission, storage, and processing capabilities. In this case, each node stores global information. Each node solves its own optimal state according to the global information. The calculation amount of this process is not only extremely large but also the calculated optimal state at the next moment is calculated based on the global information and its own state. For a single agent, the state at the next moment is single or locally optimal. As far as the whole system is concerned, each basic agent is in an optimal state and may not be able to guarantee the overall optimality. For example, in a radar networking system, a point in an area is less affected by the terrain, and the point is best detected, but if all radars are deployed at that point, the overall performance will be extremely low. At the same time, as the number of nodes increases, the amount of calculations increases exponentially. The holonic structure is controlled by layer management, and the high-level virtual agent controls the area information, and the number of optimizations is small. And from the perspective of the particular region or even the whole region, the state of the agent can be planned to obtain a global optimal solution. As the number of nodes increases, the holonic structure only increases the number of packets, and the amount of calculation will only increase exponentially.

Holonic structure has better flexibility and robustness than hierarchical management. The virtual agent generated by the holonic structure can be dynamically adjusted according to the state of the underlying agent, similar to a mechanism for regional negotiation. In the hierarchical management mode, the relationship between the high-level manager and the corresponding lower-level agent is fixed, and it is difficult to switch nodes, and the flexibility is poor. At the same time, in the hierarchical management mode, once the high-level node fails, the node and the underlying agent are all unable to work and the overall performance is

significantly reduced. The holonic structure's high-level node is virtual, and there is no possibility of failure.

2.2. Principles of HPSO Algorithm. As scholars have performed more and more in-depth research on multiagent systems, more and more results have emerged. And the multiagent system and PSO algorithm have many similarities in architecture, but no scholar has applied the theory in the agent to the improved PSO algorithm. There are two main reasons. One is that in the PSO algorithm, particles are not agents, and they do not have the ability to learn autonomously and complexly. They cannot be considered as multiagent systems. The other is because the PSO algorithm is extremely simple to operate and has strong real-time performance. If the intelligence of the particle is deliberately increased, the complexity of the algorithm will be greatly increased, resulting in a decrease in the calculation speed. It is therefore difficult to directly apply the theory of the agent system to the improved PSO algorithm.

The literature [1] is to improve the topology of PSO particles by using holonic structure in multiagent systems. For the first time, the multi-intelligent system theory was applied to the improved PSO algorithm, and the HPSO algorithm was proposed. To briefly discuss the principle of the HPSO algorithm, it is assumed that 9 particles are used to optimize the optimization problem. The principle of improving the PSO algorithm based on holonic structure is shown in Figure 2.

The first 9 particles in Level 1 are divided into three groups of G1, G2, and G3. The particles in each group calculate the corresponding fitness function according to their own state information. The particles with the best fitness function in the group are promoted to the virtual leader of the group and upgraded to Level 2. After that, the three groups of optimal particles P1, P2, and P3 are constructed into a new group, and the particles with the best fitness function are upgraded to Level 3 as a new leader. In the standard PSO algorithm, the particle state is only affected by three factors: its own speed, its own historical optimality, and global historical optimum. In the HPSO algorithm, the particle state will also be affected by the optimal solution at each level.

This method has obvious advantages. The first is to increase the diversity of the system and reduce the possibility of particles falling into local optimum and premature convergence by increasing the effect of local optimal values on the particle state through a layered architecture. Secondly, after increasing the local optimal layer, the contradiction between the search range and the search accuracy can be balanced. That is, each group is dispersed as much as possible in different areas to ensure a sufficiently large search range. Each group performs a detailed search locally, which improves the accuracy of the search. At the same time, when the search range is large, the particles of the PSO algorithm will oscillate between the individual optimal and the global optimal. HPSO can effectively alleviate this phenomenon by increasing the local optimum. Finally, combined with the simulation results in [1], it can be seen that the HPSO algorithm has obvious advantages over other improved PSO algorithms.

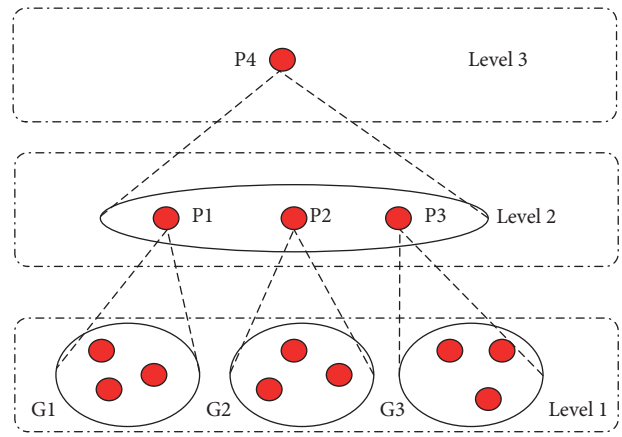

Figure 2: HPSO algorithm schematic.

2.3. HPSO Algorithm Architecture and Process. The algorithm architecture and main flow of HPSO in [1] can be simplified as shown in Figure 3.

The above process can be summarized as follows:

Step 1: initialize the PSO algorithm, the population number n, the number of iterations N, and the regrouping period T_r.

Step 2: build the original holonic structure. Set the number of layers in the structure and the number of groups per layer and randomly group the particles.

Step 3: the first $0.8N$ iterations are mainly to achieve large-scale search, in order to prevent particles from falling into local optimum. Each time a regrouping period T_r is entered, the particles are regrouped. The principle of grouping is to take two particles in the group after grouping. These two particles can come from different groups; otherwise, they are regrouped.

In the first $0.8N$ iterations, the particle parameters are adjusted using equation (1) to obtain the optimal fitness function.

$$\begin{aligned} \mathrm{Vel}^{t+1} &= w \times \mathrm{Vel}^{t+1} + \sum_{i=1}^{\mathrm{levelNumber}} c_i \times r_i \times \frac{\mathrm{levelNumber} - i + 1}{\mathrm{levelNumber}} \\ &\quad \times (\mathrm{pBest}_i - \mathrm{particlePosition}), \\ \mathrm{Position}^{t+1} &= \mathrm{Position}^{t+1} + \mathrm{Vel}^{t+1}. \end{aligned} \tag{1}$$

where Vel is particle's velocity, t is the iteration number, c_i is acceleration coefficient in level i, and r_i is random vector in level i that is uniformly distributed in the range $[0, 1]$. levelNumber shows number of levels in holarchy. In level number one, pBest_i is the personal best position of particle i. In other levels, pBest_i is the best position of each group of holarchy.

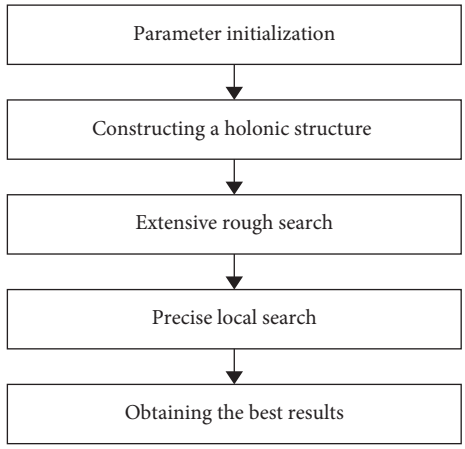

Figure 3: HPSO algorithm flowchart.

particlePosition shows position of each particle. w is called inertia weight. Position^{t+1} denotes the position of $t+1$th generation's particle.

Step 4: after the first $0.8N$ searches, it is basically possible to determine the range in which the optimal solution may appear. Therefore, the latter $0.2N$ searches are mainly for searching for the optimal solution in a small range. Therefore, it is not necessary to regroup, and the fine search is performed using the following formula:

$$\begin{aligned} \mathrm{Vel}^{t+1} &= \omega \times \mathrm{Vel}^{t+1} + c_1 \times r_1 \times (\mathrm{pBest}_1 - \mathrm{particlePosition}) \\ &\quad - \sum_{i=2}^{\mathrm{levelNumber}} c_i \times r_i \times \frac{\mathrm{levelNumber} - i + 1}{\mathrm{levelNumber}} \\ &\quad \times (\mathrm{pBest}_i - \mathrm{particlePosition}), \\ \mathrm{Position}^{t+1} &= \mathrm{Position}^{t+1} + \mathrm{Vel}^{t+1}. \end{aligned} \tag{2}$$

The meaning of the parameters is the same as in Step 3.

Step 5: get the optimization results of the HPSO algorithm.

The above process is the HPSO optimization process. Since the literature [1] is the first to use the holonic structure to improve the PSO algorithm, there is still room for further improvement in some aspects. Therefore, this paper carries out subsequent research.

3. Improvement of Studies

Literature [1] combined the idea of an agent with PSO and considered PSO not only as a single point but also as an intelligent particle interacting between levels, thus improving the efficiency of the algorithm.

3.1. Grouping Strategy of Particles. In literature [1], the author constructed the particle groups by approximate bisection, which may be the most convenient but not necessarily the best. Also, in the conclusion of [1], the authors also discussed that "It is also possible to create groups by special rules to achieve specified purposes." In this regard, this study proposes two completely opposite grouping strategies based on system clustering and information entropy and discusses and compares these two strategies.

The system clustering method calibrates the similarity degree of particles according to the distance between particles. Also, the effect is intuitive and obvious. By means of system clustering, particles with close distance or high similarity can be divided into groups. This grouping method can obtain local regions constructed by similar particles, and it is easy to find the local optimization of a certain region. In this way, the optimization problem is transformed into dynamically adjusting the grouping method and searching for local region optimization, and the entire optimization can then be conducted.

Entropy is a measure of microstate diversity or uniformity in thermodynamics. Information entropy is a concept in information theory, used to describe the probability of different states appearing in a system. This suggests that the greater the entropy is, the greater the disorder will be, and the more possible states will occur. As a result, the probability of the particles falling into global optimization is smaller. This way of grouping is to make the maximum entropy and maximum information in each group after determining the number of groups, i.e., dividing the most distant particles into a group. In such a way, the total entropy of the system will become the maximum, which significantly reduce the possibility of the system falling into local optimum.

As no free lunch suggests, the improved intelligent algorithm is to balance the exploration of the search space (diversification) and the exploitation of prominent regions (intensification). From the above discussion, it can be seen that the grouping method based on system clustering is to improve the local retrieval ability, i.e., the intensification. The method based on information entropy is to make the system more detectable space, i.e., to improve the diversification. Thus, both approaches are strategies for improving grouping, but the goals are completely opposite. Theoretical analysis and experimental verification of the optimized performance of the two methods will be provided later. Because of this contradiction, this study applies the two methods to different stages of search strategy to improve the efficiency of the algorithm.

3.2. Setting of Number of Iterations. In literature [1], the authors set two search strategies for alternation and confirmed that in the iteration process, the first 80% iterations adopt a particle motion state update method through a lot of experimental simulation and analysis. This suggests that the influences of group leaders in lower levels are more than those that are in higher levels, and the ability of detection is improved partially. During the last 20% of iterations, the impact of high-level particles becomes increasingly big, turning into another method to update the motion state of particles.

As literature [1] states, "We have divided number of iterations to 60%–40%, 70%–30%, 80%–20%, and 90%–10%. These divisions were tested on some randomly selected functions. The best results were obtained when iterations were divided to 80% and 20%." The authors have done a lot of work and given the experimental parameters which can get better results.

From the above discussion, it can be seen that there are two limitations in the scheme. The first one is that the algorithm in this paper has a proportion of 80%–20%, which directly leads to the number of iterations set directly related to the algorithm performance. The other is the 80%–20% is only the optimal solution of the 4 groups of experimental parameters in [1]. That is, the ratio of the two search methods, or the switching time of the two search strategies, is highly correlated with the performance of the algorithm after the number of iterations is determined. However, whether the proportion of 80%–20% can make the algorithm to be optimal and whether the proportion is applicable to all methods remains to be further studied.

If a problem is solved by cluster intelligence algorithm, it must be a NP-Hard problem. It is almost unlikely to acquire the number of iterations by empirical equations or rough estimates. The accuracy of the calculations will vary slightly with one more dimension or one less, and the number of iterations us likely to vary completely, let alone solve different problems. For different problems and functions, the parameters of the algorithm itself are likely to be significantly different. Therefore, whether the proportion of 80%–20% obtained by simulation in literature [1] is suitable needs further investigation.

To decide whether to follow the concept of literature [1], the number of iterations should be first determined, and then the ratio of two algorithms should be confirmed. The ultimate goal of this approach is to determine the switching time between the two search strategies. If better results can be obtained, it is not necessary to proceed according to literature [1]. More importantly, the two methods in literature [1] only switched once, whereas this study combines the fitness function value, builds the adaptive adjustment mode of search strategy, and breaks the mode in literature [1]. According to the fitness value of particle swarm in the search process, the switching between the two search strategies can be realized by self-adjusting the search strategy, and the optimization of the problem can be achieved.

3.3. Switch and Termination of Self-Adaption. The above discussion suggests that the algorithm efficiency of reference [1] is associated with the number of iterations. If the number of iterations is set excessively high, the computation of the algorithm will be inevitably more complex. However, if the number of iterations is set too low, the algorithm will end prematurely and probably not converge to the optimal solution. In the meantime, the number of iterations of the algorithm may vary significantly with problems and fitness functions, computational accuracy, and requirements. It is difficult to estimate the number of iterations of the algorithm for a specific problem without any prior knowledge or experimental basis, which severely limits the applicable scope of the algorithm.

There are two reasons to determine the number of iterations of the algorithm. The one is to determine the number of operations of the two search strategies, and the other is to determine the termination conditions of the algorithm. For the first reason, the above discussion has provided a feasible solution. The termination condition of the algorithm seeks to meet the requirements for the results in addition to reaching the number of iterations, i.e., the fitness function no longer decreases or reaches the predetermined calculation accuracy.

Due to different problems, it is hard to determine the decline process and trend of fitness function. To this end, this study builds an adaptive threshold and combines previous grouping strategies. Based on the comparison of different grouping results, the algorithm's termination conditions are constructed to determine the conversion time and termination conditions.

4. Grouping Strategy

4.1. Grouping Mode Based on Hierarchical Clustering Method. When the clustering begins, take n elements (particles) with m dimensions as one class and specify the distance between objects and the distance between classes. The two classes closest to each other are then merged into a new class (the union class), and the distance between the new class and the other classes is calculated. Repeat the merging of the nearest class, each time reducing one class until all the elements are merged into one. Finally, a relation map (cluster tree diagram or pedigree diagram) is formed. It can be clearly seen from the diagram that there should be several categories and the elements contained in each category.

In the analysis of clustering, the class is usually represented by G. Assume that there are m elements in G, denoted as the column vector of x_i ($i = 1, 2, \ldots, m$). d_{ij} denotes the distance between x_i and x_j. D_{KL} is the distance between G_K and G_L. Using different methods to define distance, different system clustering methods are generated. Here, the most mature and easy way to implement the short-distance method is introduced (Single Linkage Method).

Define the distance between classes as the distance between the two closest elements; it is expressed as

$$D_{KL} = \min\left\{d_{ij}\colon x_i \in G_K, x_j \in G_L\right\}. \tag{3}$$

If G_K and G_L are clustered as a new group in a certain step, this group is marked as G_M. Hence, the distance of G_M and any existing group G_J is expressed as

$$D_{MJ} = \min\left\{D_{KJ}, D_{LJ}\right\}, \quad J \neq K, L. \tag{4}$$

The procedures of hierarchical clustering method based on the Single Linkage Method is as follows:

Step 1: each of the initial elements is recognized as a class, and the distance between the elements is

specified. The Euclidean distance is usually used to calculate the distance matrix $D_{(0)}$ of n element(s).

Step 2: find the minimum element of $D_{(0)}$ and set it as D_{KL}. Cluster G_K and G_L as a new group and mark this group as G_M, i.e., $G_M = \{G_K, G_L\}$.

Step 3: calculate the distance between new group G_M and any group G_J using recursion formula, which is expressed as

$$\begin{aligned} D_{MJ} &= \min_{x_i \in G_M, x_j \in G_J} d_{ij} \\ &= \min\left\{ \min_{x_i \in G_K, x_j \in G_J} d_{ij}, \min_{x_i \in G_L, x_j \in G_J} d_{ij} \right\} \\ &= \min\left\{ D_{KJ}, D_{LJ} \right\}. \end{aligned} \tag{5}$$

Modify the distance matrix $D_{(0)}$ and combine the row and column of G_K and G_L as a new row and column corresponding to G_M. The distance between the new row and column is calculated by equation (5), and the rest values of the row and column remain unchanged, thereby obtaining the new matrix $D_{(1)}$.

Step 4: $D_{(1)}$ is processed in the same way to get $D_{(2)}$, which is performed successively until all elements are merged into a preset number of groups.

The initial clustering of particles can be realized through the above process.

4.2. Grouping Mode Based on Information Entropy. Entropy is a measure of the diversity or uniformity of microscopic states in thermodynamics, representing the degree of disorder of a system. Information entropy was first proposed by Shannon in 1948 to describe the degree of disorder of information. The disordered degree of information is negatively correlated with the information's utility, and the information entropy can be used to measure the extent to which the evaluation object contains effective information.

The classical description of information entropy is assume there exist n elements, m dimensions, and the matrix of original data is $\mathbf{X} = [x_{ij}]_{n*m}$, $x_{ij} \geq 0$ ($i = 1, 2, \ldots, n$; $j = 1, 2, \ldots, m$). The original data are unitized; it yields

$$p_{ij} = \frac{x_{ij}}{\sum_{j=1}^{m} x_{ij}}. \tag{6}$$

The unitized matrix is $\mathbf{P} = [p_{ij}]_{n*m}$. Calculate the information entropy of each research object E_j:

$$E_j = -k \sum_{i=1}^{n} \left(p_{ij} \ln p_{ij} \right), \tag{7}$$

where $k = 1/\ln n$, if $p_{ij} = 0$, $p_{ij} \ln p_{ij} = 0$.

In this paper, the original formula of information entropy is not used, whereas the concept of information entropy is borrowed and simplified.

On the premise of setting the number of groups, the greater the entropy within a group is, the higher the possibility of the entropy it will have, and therefore the lower the probability that the group of particles will fall into local optimum. Assume that n elements are divided into Q groups and that there are P grouping methods in total, and the information entropy within each group of each grouping method is $E_{z1}, E_{z2}, \ldots, E_{zQ}$. If the sum of information entropy reaches the maximum, it yields

$$\max_{z \in P} \left(\sum_{s=1}^{Q} E_{zs} \right). \tag{8}$$

Then, the space explored by this grouping method is the largest, and the probability of falling into local optimum is significantly reduced.

Obviously, equation (8) also pertains to an optimization. If all possibilities are traversed and the amount of calculation is increased, an approximate method is adopted in this paper. The greater the entropy of a group, the weaker the relationship between the particles and the greater the distance between them. In contrast to the system clustering method in the previous section, the system clustering method is to cluster particles with close distance into a class, while the information entropy is to cluster particles with multiple distances into a class.

In this regard, when all the elements in the distance matrix $D_{(0)}$ in the system clustering method are inversely taken and other operations are identical, the particles farthest from each other can be clustered into one class. Also, the calculation is significantly simplified.

4.3. Combination of the Grouping Method and the Search Strategy. Through the above discussion, the grouping criteria of the two methods are totally different. Since there are exactly two search strategies in literature [1], two types of search in literature [1] are called extensive search and accurate search for clear expression.

Extensive search is equation (5) in literature [1], namely, equation (1) in this study. The exact search is equation (8) in literature [1], namely, equation (2) in this study.

Since both search methods involve grouping particles, four results are yielded by combining the two methods proposed in this paper with the two search strategies, as shown in Table 1.

Though new grouping methods are established, if the grouping method is mismatched with the search strategy, it will not only lead to a surge of computation, but to a converge on local optimum, or even make the efficiency of the improved algorithm may not be as effective as that of the original algorithm. Therefore, this paper discusses and analyses the above algorithm.

There is a problem in both plan 1 and plan 3. Extensive search suggests that the larger the search space is, the better it will be; however, the systematic clustering method is on the contrary. The particles are locked into several small local areas, as shown in Figure 4.

Table 1: Combination of grouping and search strategy.

Plan	First stage	Second stage
Original plan	Extensive search	Accurate search
Plan 1	System clustering method + extensive search	System clustering method + accurate search
Plan 2	Information entropy method + extensive search	Information entropy method + accurate search
Plan 3	System clustering method + extensive search	Information entropy method + accurate search
Plan 4	Information entropy method + extensive search	System clustering method + accurate search

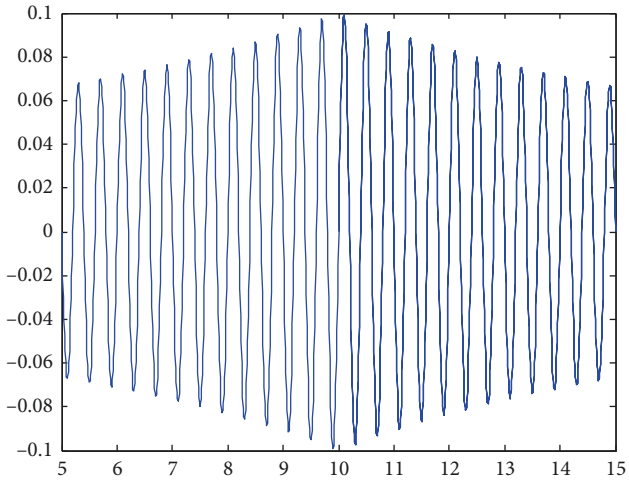

Figure 4: Attenuation of oscillation.

For the minimum of Figure 4, if the initial particles are randomly distributed on both sides of the minimum value, it takes a lot of time to search for multiple local optimal values, and the minimum value may be passed in the subsequent migration process. In such a way, the algorithm may get the nonglobally optimal solution. Combined with the system clustering method and extensive search, the algorithm efficiency will decline. Therefore, plan 1 and plan 3 are not ideal.

For plan 2, in the later accurate search, the search strategy of information entropy may make it difficult for particles to aggregate, and the search efficiency of the precise search area is not high, which decreases the algorithm efficiency.

For plan 4, the information entropy method is combined with the extensive search to make the search more extensive and more possible, and the repeated search for the region is almost impossible. While in the later stage, the system clustering method is combined with accurate search to make the search more accurate. Qualitative analysis suggests that both search strategies are strengthened in plan 4, and the overall efficiency of the algorithm is improved. Therefore, in the first part of the simulation, the algorithm efficiency of the four combinations and the original method (totally five methods) is compared, which corroborates the discussion in this paper.

5. Condition Switching Criteria

The improved particle swarm algorithm in reference [1] involves two switching moments of state. The first is the switching time from extensive search to precise search, and the second is the end of the algorithm. In the original paper, the first switching point is studied, and through a lot of simulation experiments and parameter comparison, the most ideal results in the experiment are given, i.e., extensive search is used in the first 80% iterations, and precision is used in the last 20%, leading to three problems:

(1) Whether the ratio of 80%–20% is the optimal?

(2) By considering only switching from extensive search to precise search once rather than from precise search to extensive search, the likelihood of jumping into local optimization will be reduced.

(3) The number of iterations of the iteration is associated with the performance of the algorithm. How to determine the total number of iterations?

In view of the above problems, two adaptive switching criteria are designed in this section. The first criterion is to switch from extensive search to accurate search according to the absolute slope of fitness function in the context of extensive search. The second criterion is to realize the algorithm's termination or switch to extensive search according to the distance relation of grouping under the precise search condition.

The adaptive strategy allows you to do without presetting the number of iterations. The corresponding ratios and search strategies can be switched. At the same time, the algorithm can be adaptive to terminate, and the termination conditions are strict. Thus, the algorithm is further less likely to fall into local optimum.

5.1. Threshold of Self-Adaption. Based on the above discussion, the particle swarm initially adopts an extensive search strategy based on information entropy method, during which the grouping of particles will not be changed. After a certain number of iterations, the particles gradually aggregate to the region or neighborhood where the global optimal solution is located. In the following process, the search strategy needs to be changed to improve the search accuracy and speed of the algorithm for local areas. At this point, the system clustering-based precision search strategy should be replaced.

In the search process, the value of fitness function decreases as the number of iterations increases. When the local or global optimization is found, the size may be unchanged. The fitness function is not an analytical expression but calculated during the iteration. At the same time, it is impossible to estimate the value of fitness function when the

algorithm converges, and for different optimization problems, the order of magnitude of fitness value is different when the algorithm stops.

Based on the change trend from the absolute value of slope of fitness function, it yields

$$k_i = \left|F_{\text{fit}}(i) - F_{\text{fit}}(i-1)\right|, \tag{9}$$

where I denotes the number of iterations, $F_{\text{fit}}(i)$ is the fitness function value of the ith iteration, and k_i is the absolute value of the change rate of fitness function.

k_i can reflect the searching state of particle. At the initial stage, the fitness function value drops rapidly, and there is no need to switch. When the particle is searched and the local or global optimal is gradually found, the fitness function decreases slowly. Besides, the rate of change overall presents a gradually decreasing trend, which can be switched to accurate search. For this reason, the conditions for judging the switch from extensive search to precise search are constructed, namely,

$$\frac{k_{i-2\times j} - k_{i-j}}{k_{i-2\times j}} < \omega \frac{k_{i-j} - k_i}{k_{i-j}}, \tag{10}$$

where $(k_{i-j} - k_i)/k_{i-j}$ denotes the ratio of change of fitness function $(k_{i-j} - k_i)$ to original change of k_{i-j} after jth iterations. As the particle gets increasingly closer to the optimal solution, the ratio increases, meaning that the falling of particle becomes increasingly slow. Ω is the weight coefficient, normally valued as [0.8, 0.9]. j can be the smaller positive integer. The value of ω and j can be arbitrary since the discriminant condition of reverse switching will be discussed later, even if the initial value is not well taken. It is also possible to switch back through the reverse switching strategy, which is complementary to each other to improve the optimization efficiency, which is also a major advantage of this method.

Given that the case may be trapped in local optimum during the search, i.e., the fitness function may be 0 and the denominator of equation (10) fails. Thus, in the actual optimization process, equation (11) is employed for discrimination:

$$k_{i-j}\left(k_{i-2\times j} - k_{i-j}\right) < \omega k_{i-2\times j}\left(k_{i-j} - k_i\right), \tag{11}$$

namely, the multiplication. It can be switched if it meets equation (11).

However, the theoretical analysis and experimental results suggest that equation (11) is also met when $i-2j$ subiteration approaches the local optimal, the $i-j$ subiteration is the local optimal or approximate local optimal, and the ith subiteration gets rid of the local optimal. Yet, this is obviously not the time for switching. Thus, a criterion is added before formula (11) to determine whether the local optimal is jumped out, which is expressed as

$$k_i < \alpha k_{i-j}, \tag{12}$$

where α can be valued as the positive integer between 2 and 10. If the algorithm gets rid of local optimization, the rate of change of fitness function will increase sharply, which is obviously greater than the last slope. Equation (12) is taken as the first criterion. Then, equation (11) is used for discrimination. When both are satisfied, the system switches to an accurate search mode based on system clustering; otherwise, the extensive search will continue.

5.2. Algorithm Termination and Reverse Switching Conditions. The condition of moving from extensive search to precise search is discussed above. When the particle switches to precise search, local or global optimization is searched. The algorithm is expected to jump out of the local optimum or converge to the global optimum. When the particle searches for the local optimum and finds the optimal value, the fitness function no longer reduces but is not the global optimum. At this point, the fitness function is just an auxiliary discrimination condition, and a new discrimination condition is required.

In addition to the fitness function, the state of all the particles is changing, i.e., the location information. The conventional approach is to end the algorithm when all the particles converge to a point, i.e., the coordinates of all the particles are almost identical. Whether it converges to global optimum is hard to ensure. Therefore, this paper constructs the discriminant condition according to the dynamic position information of the particle and realizes the switching of system state.

The discrimination is still constructed with double discriminant condition, and the fitness function of the particle is no longer decreasing no matter whether the particle converges to global or local optimal. The corresponding slope is 0, suggesting that the first discrimination condition is

$$k_i = 0. \tag{13}$$

If the condition of equation (13) is not satisfied, the system will continue to perform the exact search. Otherwise, the next step will be determined.

When the condition of equation (13) is satisfied, the particles are classified again based on information entropy according to the previous grouping method of system clustering and information entropy, and they are classified but not searched. The sum of distance of each group is defined as SD_p, which is expressed as

$$SD_p = \text{sum}\left\{d_{ij}\right\}, \quad p \in [1, z]. \tag{14}$$

Set the minimum sum of distance as $SD_{pe\min}$ based on the grouping of $\{SD_{pe1}, SD_{pe2}, \ldots, SD_{pez}\}$ using the information entropy method. Set the maximum sum of distance as $SD_{pe\max}$ based on the groups of $\{SD_{ps1}, SD_{ps2}, \ldots, SD_{psz}\}$ following the hierarchical clustering method. These are written as follows:

$$\begin{aligned} SD_{pe\min} &= \min\left\{SD_{pe1}, SD_{pe2}, \ldots, SD_{pez}\right\}, \\ SD_{ps\max} &= \max\left\{SD_{ps1}, SD_{ps2}, \ldots, SD_{psz}\right\}. \end{aligned} \tag{15}$$

The second discrimination condition is

$$SD_{pe\min} \le SD_{ps\max}. \tag{16}$$

When the distance and the closest grouping based on the information entropy method is not greater than the broadest grouping based on the system clustering method, it can be judged that the particle has converged to the global optimal value and the algorithm ends.

The criterion of equation (13) is to show that all particles have been clustered in a certain region. At this point, search again only seeks the optimum in this region, and the search strategy no longer changes. A precise search strategy is used until all particles are clustered at a point, which is then convergent. This method can achieve autonomous convergence and improve the adaptability of the method.

6. Flow of Adaptive Holonic Particle Swarm Optimization

To sum up, the algorithm flow of Adaptive Holonic Particle Swarm Optimization (AHPSO) designed and optimized in this section is illustrated in Figure 5.

The above algorithm flow can be summarized as the following steps:

Step 1: initialize system parameters and set the number of population, inertial coefficient, weight coefficient, group number, system layer number, and other parameters.

Step 2: use the information entropy method to group the particles, implement a comprehensive search strategy, and calculate the fitness function and slope.

Step 3: discriminate using equations (11) and (12). Step 2 is performed when one or all of the two conditions are not met. Step 4 is performed when both conditions are met.

Step 4: group the particles by system clustering.

Step 5: perform the exact search strategy and calculate the fitness function and slope.

Step 6: determine whether the slope of the fitness function is 0. If it is not 0, go back to Step 5. Otherwise, perform Step 7.

Step 7: keep the particles still, group the particles by two methods, calculate $SD_{pe\,\min}$ and $SD_{ps\,\max}$, and determine the relative size of the two. When $SD_{pe\min}$ is larger than $SD_{ps\,\max}$, go back to Step 2. When $SD_{pe\min}$ is smaller than $SD_{ps\,\max}$, the search strategy does not switch but searches precisely until all particles converge to a point.

Step 8: obtain the optimal value coordinate, i.e., the optimal solution, and the algorithm ends.

Step 3 and Step 7 in the above process switch the system state and change the system search strategy adaptively according to the actual state of particles. More importantly, the algorithm can self-converge without presetting the number of iterations, thereby improving its applicability.

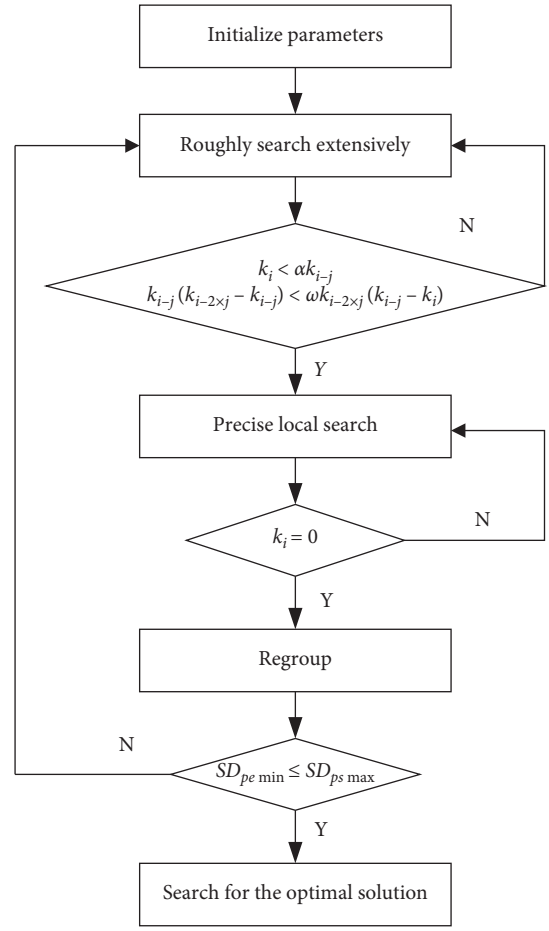

Figure 5: Flow of algorithm.

7. Verification of Simulation

To verify the effectiveness of the proposed algorithm and guarantee the objectivity of algorithm comparison, 22 standard test functions in the appendix of literature [1] are adopted. In addition, the subsequent simulation parameters in this study are consistent with those in [1]. In other words, each algorithm runs 30 times, the population size is 100, the particles are divided into 5 groups, holonic has 3 layers, r is randomly evaluated in [0,1], the value of ω is about one and gradually decreases with the rise of the number of iterations, $c = 1.3667$, the algorithm is iterated 10^6 times, and dimension of the test function is $D = 30$. In the optimized grouping strategy, $j = 1$, $\omega = 0.8$, and $\alpha = 3$. Also, the mean, standard deviation (Std), and median are selected for comparison. Since this study is based on literature [1], it directly refers to the simulation results in literature [1], namely, the optimization results of HPSO on 22 groups of standard test functions, and the optimal value of the algorithm corresponding to each group of test functions.

The simulations were employed in a Lenovo personal computer with Core™ i7-4960 CPU operating under a clock frequency of 2.60 GHz and 16 GB of memory.

In this study, two groups of simulation experiments are conducted. The first group is a comparison of the four new

search mode improved PSOs formed by the combination of the two grouping methods discussed above and the search strategy arrangement and various improved PSO algorithms. The second group is a comparison between the AHPSO algorithm and the HPSO algorithm.

7.1. Comparison between the Combinations of Grouping Method and Searching Strategy. This paper proposes the grouping method based on information entropy and system clustering. At the same time, there are two strategies in the HPSO algorithm: extensive search and accurate search. By permutation and combination, four combination schemes can be formed. Although the performance of the four schemes is analyzed qualitatively, the performance of the four algorithms needs to be verified quantitatively.

For the clear expression, the four search strategies are briefly described. If the first layer is expressed as the information entropy method + extensive search, the second layer is expressed as the system clustering method + precise search mode, briefly expressed as E-S-HPSO. Likewise, the other three are E-E-HPSO, S-E-HPSO, and S-S-HPSO. These four algorithms are optimized with the HPSO on the test function of $D = 30$, since only the improvement effect of the grouping strategy is explained. Therefore, the number of iterations is 10^6, and the proportion of search strategy is 80–20%.

At the same time, referring to the parameters of the literature [1], compared with the existing methods, respectively, Local PSO (LPSO) [35], Hierarchical PSO (PS2O) [36], Multi-Population Cooperative PSO (MCPSO) [37], Competitive and Cooperative PSO with Information Sharing Mechanism (CCPSO-ISM) [38], and Human-Brain Simulated PSO (HSPSO) [39], Fully Informed PSO (FIPSO) [40], and Fitness-Distance-Ratio Based PSO (FDRPSO) [41]. The mean, Std, and median after 30 Monte Carlo simulations were obtained. The results are listed in Tables 2 and 3.

Comparing the results of Tables 2 and 3, it can be seen that the results of E-S-HPSO are slightly stronger than those of HPSO. This is because only the grouping strategy of particles is changed and the search method of particles is not changed. The performance of $F5$ and $F7$ did not surpass that of HPSO. Therefore, only from the results, the E-S-HPSO algorithm did not significantly improve the accuracy of the algorithm.

Overall, the E-S-HPSO algorithm performs better and is significantly better than other improved algorithms.

The results of comparison of functions from $F9$ to $F16$ are listed in Tables 4 and 5.

The comparison results in Tables 4 and 5 suggest that the performance of E-S-HPSO algorithm is stronger than that of HPSO except for $F10$ and $F15$, whereas there is no change in magnitude. This is because the search strategies for both have not changed, and the results have not established a magnitude advantage.

It can be seen from the comparison that the H-S-HPSO algorithm is superior to other improved algorithms.

Finally, $F17$ to $F22$ are compared, and the results are shown in Tables 6 and 7.

Compared with the results in Tables 6 and 7, the performance of the E-S-HPSO is also slightly stronger than that of HPSO, except for $F21$. The test results of the remaining three grouping methods were not as good as E-S-HPSO and HPSO, and the overall performance of E-E-HPSO is stronger than S-E-HPSO. This is because the initial stage of system E-E-HPSO search range is wider, and the possibility of local optimization is lower. S-S-HPSO is stronger than S-E-HPSO, and the performance of S-E-HPSO is one order of magnitude lower than that of E-S-HPSO.

Comparing Table 6 and Table 7, it can be seen that the H-S-HPSO algorithm is superior to other improved algorithms.

It can be seen that the effects of different grouping methods and search strategy combination on algorithm performance are relatively different.

From the test results of 22 standard functions in the above two groups, it can be quantitatively explained that the performance of E-S-HPSO algorithm is better than that of HPSO. Furthermore, by comparing the results parameters of E-S-HPSO with those of S-S-HPSO and E-E-HPSO, respectively, it can be seen that the effect of the information entropy method in the extensive search phase is better than that of the system clustering method. Likewise, the system clustering method is more suitable for the exact search stage.

The advantage of E-S-HPSO improvement is not the optimization result but the optimization rate. This study uses Figure 6 convergence characteristics and performance for different algorithms on different functions; the horizontal axis represents number of function evaluations and vertical axis represents logarithm of $F(x) - F(x^*)$ of [1]. x^* denotes the theoretically optimal solution. Comparing the performance of E-S-HPSO and other algorithms, the simulation results are shown in Figure 6.

Figure 6 suggests that E-S-HPSO has a faster convergence speed. In the initial stage, E-S-HPSO enables extensive search to have a larger range, while in the accurate search stage, the search is more accurate, and the algorithm is more efficient, thus improving the performance of the algorithm.

To further compare the improved performance of the algorithm, this paper uses nonparametric statistical comparison algorithm performance. Considering that the optimization results do not necessarily obey the normal distribution, the variance is not the same. Therefore, the results of the 30 optimization results of the E-S-HPSO algorithm in the above six groups of functions and the results of eight algorithms such as HPSO are sequentially performed by the Wilcoxon test. The significance level detection $\alpha = 0.05$ was set, and the results of obtaining the significance value p are shown in Table 8.

Through the above comparison, it can be observed that $F1$, $F6$, $F13$, and $F16$ can be obtained. For a more basic or simple function, the improved algorithm has no obvious improvement in effect. But for complex, difficult to find functions, such as $F18$ and $F19$, the effect of this method is significantly improved. The values of p are all less than 0.05, and it is considered that there is a significant improvement.

Table 2: Evaluation results on eight functions ($F1$–$F8$, $D = 30$).

Test function	Performance	Best	HPSO	E-S-HPSO	S-E-HPSO	E-E-HPSO	S-S-HPSO
	Mean	0*E*000	0*E*000	0*E*000	0*E*000	0*E*000	0*E*000
*F*1	Std	0*E*000	0*E*000	0*E*000	0*E*000	0*E*000	0*E*000
	Median	0*E*000	0*E*000	0*E*000	0*E*000	0*E*000	0*E*000
	Mean	0*E*000	0*E*000	0*E*000	0*E*000	0*E*000	0*E*000
*F*2	Std	0*E*000	0*E*000	0*E*000	0*E*000	0*E*000	0*E*000
	Median	0*E*000	0*E*000	0*E*000	0*E*000	0*E*000	0*E*000
	Mean	9.63559*E* − 9	9.63559*E* − 9	0*E*000	6.96509*E* + 3	1.1175*E*0	2.97874*E* + 1
*F*3	Std	4.59479*E* − 8	4.59479*E* − 8	0*E*000	5.56447*E* + 3	1.641275*E*0	8.35595*E* + 1
	Median	1.31414*E* − 16	1.31414*E* − 16	0*E*000	6.79982*E* − 3	1.121311*E*0	2.7192*E* + 1
	Mean	4.07853*E* + 6	4.07853*E* + 6	3.808925*E* + 6	1.452359*E* + 7	4.361641*E* + 7	5.031646*E* + 6
*F*4	Std	4.42817*E* + 6	4.42817*E* + 6	3.506181*E* + 6	3.66862*E* + 6	3.66862*E* + 6	5.17233*E* + 6
	Median	5.28766*E* + 6	5.28766*E* + 6	3.865954*E* + 6	1.407532*E* + 6	4.302474*E* + 7	5.102473*E* + 6
	Mean	6.64778*E* + 3	6.64778*E* + 3	7.254042*E* + 3	9.2049605*E* + 3	7.409303*E* + 3	8.120749*E* + 3
*F*5	Std	1.34526*E* + 3	1.34526*E* + 3	6.305487*E* + 3	9.124144*E* + 3	7.417192*E* + 3	8.071724*E* + 3
	Median	6.64302*E* + 3	6.64302*E* + 3	7.611413*E* + 3	9.2794509*E* + 3	7.442609*E* + 3	8.1289*E* + 3
	Mean	3.662076*E*0	3.662076*E*0	3.25191*E*0	7.11516*E* + 1	3.192067*E* + 1	3.00596*E* + 1
*F*6	Std	4.348042*E*0	4.348042*E*0	3.754928*E*0	1.02242*E* + 2	3.228586*E* + 1	1.64801*E* + 1
	Median	0.240781*E*0	2.40781*E* − 1	2.393705*E* − 1	6.99444*E* + 2	2.759627*E* + 1	2.614366*E* + 1
	Mean	6.2172*E* − 15	2.132*E* − 7	7.254042*E* − 7	9.2049605*E* − 3	7.409303*E* − 5	8.120749*E* − 5
*F*7	Std	0*E*000	3.8009*E* − 7	6.305487*E* − 7	9.124144*E* − 3	7.417192*E* − 5	8.071724*E* − 5
	Median	6.2172*E* − 15	6.2172*E* − 15	7.611413*E* − 14	9.2794509*E* − 3	7.442609*E* − 5	8.1289*E* − 5
	Mean	0*E*000	7.4*E* − 4	1.093266*E* − 9	7.73591*E* − 1	6.849328*E* − 2	3.7138*E* − 2
*F*8	Std	0*E*000	2.257*E* − 3	1.109273*E* − 8	1.214117	5.326303*E* − 2	2.2558*E* − 2
	Median	0*E*000	0*E*000	1.050083*E* − 9	8.16841*E* − 1	6.3397*E* − 2	4.27247*E* − 2

Table 3: Evaluation results on eight functions ($F1$–$F8$, $D = 30$).

Test function	Performance	LPSO	MCPSO	PS2O	HSPSO	CCPSO-ISM	FDR-PSO	FIPPSO
	Mean	1.60014*E* − 34	6.26379*E* − 81	1.76947*E*0	1.12945*E* − 8	5.07323*E* − 47	0*E*000	2.93494*E* − 30
*F*1	Std	5.93931*E* − 34	2.49518*E* − 80	6.75503*E*0	4.39952*E* − 8	2.35810*E* − 46	0*E*000	3.33407*E* − 30
	Median	2.86022*E* − 35	9.93019*E* − 82	0*E*000	1.01662*E* − 9	2.45805*E* − 47	0*E*000	2.13543*E* − 30
	Mean	3.27043*E* − 60	9.74950*E* − 89	0*E*000	1.787335*E* − 81	1.70045*E* − 110	0*E*000	1.17408*E* − 68
*F*2	Std	4.41014*E* − 59	6.40710*E* − 88	0*E*000	1.66665*E* − 81	2.08581*E* − 110	0*E*000	4.13610*E* − 68
	Median	4.01844*E* − 62	1.08863*E* − 90	0*E*000	1.84299*E* − 82	1.222593*E* − 112	0*E*000	5.77086*E* − 69
	Mean	1.70201*E*0	1.79866*E* + 4	3.27678*E* + 3	3.40681*E* + 3	3.16500*E*3	1.76578*E*2	3.44002*E*0
*F*3	Std	3.26814*E*0	3.99189*E* + 3	2.38302*E* + 3	2.93343*E* + 3	1.32869*E*3	5.70738*E*1	1.38666*E*0
	Median	1.12323*E*0	1.21027*E* + 4	2.29597*E* + 3	4.33609*E* + 3	6.32184*E*3	1.13972*E*2	3.60090*E*0
	Mean	5.45540*E* + 06	7.23631*E* + 6	1.39998*E* + 7	8.68470*E* + 6	9.18903*E*6	4.38271*E*6	2.18793*E*7
*F*4	Std	5.11632*E* + 07	5.93250*E* + 6	3.16866*E* + 7	2.18534*E* + 6	1.40322*E*6	2.03391*E*6	5.19729*E*6
	Median	3.97467*E* + 06	4.01336*E* + 6	2.74837*E* + 7	5.41334*E* + 6	8.99822*E*6	4.50993*E*6	2.14160*E*7
	Mean	7.51895*E* + 03	9.42133*E* + 3	1.01102*E* + 4	7.89266*E* + 3	6.00111*E*3	7.32004*E*3	1.55529*E*4
*F*5	Std	5.16267*E* + 03	3.00486*E* + 3	1.92308*E* + 4	7.51229*E* + 3	1.29366*E*3	6.48941*E*2	1.09525*E*3
	Median	3.2567*E* + 02	3.73071*E* + 2	1.729841*E* + 4	7.78256*E* + 3	6.45066*E*3	7.60076*E*3	1.73125*E*3
	Mean	1.63600*E* + 1	8.12548*E* + 1	6.58557*E* + 1	1.223063*E*0	1.12802*E* + 1	2.05885*E* + 1	2.68981*E* + 1
*F*6	Std	2.5058*E* − 1	1.7968887*E* + 2	1.37060*E* + 2	1.83256*E*0	1.29335*E* + 1	1.75849*E*0	6.6699*E* − 1
	Median	1.58943*E* + 1	2.10025*E* + 1	7.49879*E* − 16	1.28955*E* + 1	1.13015*E* + 1	2.04516*E* + 1	2.86414*E* + 1
	Mean	6.2172*E*0	1.51248*E* + 1	2.10483*E*0	2.01931*E* − 9	8.65697*E* − 12	2.28853*E*0	4.83970*E* − 4
*F*7	Std	0*E*000	1.12796*E* + 1	4.67083*E*0	4.35639*E* − 7	4.43108*E* − 11	0.57968*E*0	5.80258*E* − 4
	Median	6.2172*E*0	1.86962*E* + 1	5.79893*E* − 15	8.66757*E* − 8	2.70069*E* − 13	2.47471*E*0	2.18214*E* − 4
	Mean	6.12511*E* − 3	7.59568*E* − 3	4.98476*E*0	2.35479*E* − 8	0*E*000	1.46257*E* − 2	2.50855*E* − 13
*F*8	Std	5.48186*E* − 3	6.57201*E* − 3	1.93703*E* + 1	7.27168*E* − 8	0*E*000	1.55388*E* − 2	9.82964*E* − 13
	Median	0*E*000	0*E*000	2.90790*E* − 3	3.69214*E* − 8	0*E*000	7.37120*E* − 3	0*E*000

Table 4: Evaluation results on eight functions ($F9$–$F16$, $D = 30$).

Test function	Performance	Best	HPSO	E-S-HPSO	S-E-HPSO	E-E-HPSO	S-S-HPSO
	Mean	0*E*000	1.39034*E* + 1	1.38478*E* + 01	7.63094*E* + 01	2.01554*E* + 01	3.49364*E* + 01
*F*9	Std	0*E*000	3.83982*E*0	3.44969*E* + 00	7.17534*E* + 01	8.69272*E* + 00	8.82999*E* + 00
	Median	0*E*000	1.42459*E* + 1	1.40510*E* + 01	7.37974*E* + 01	2.36655*E* + 01	3.04197*E* + 01
	Mean	9.3786*E* − 1	4.15448*E* + 1	4.14240*E* + 01	1.78288*E* + 02	4.59246*E* + 01	6.24880*E* + 01
*F*10	Std	9.7714*E* − 2	1.30013*E* + 1	1.27993*E* + 01	5.17995*E* + 01	1.94761*E* + 01	2.29971*E* + 01
	Median	9.87329*E* − 1	2.07210*E* + 1	2.06728*E* + 01	1.74495*E* + 02	4.09533*E* + 01	5.60650*E* + 01
	Mean	2.08050*E* + 1	2.08050*E* + 1	2.08456*E* + 01	1.09322*E* + 02	3.55660*E* + 01	2.46454*E* + 01
*F*11	Std	8.38112*E* − 2	8.38112*E* − 2	5.48504*E* − 01	9.14125*E* + 01	2.59134*E* + 00	3.05622*E* + 00
	Median	2.07210*E* + 1	2.07210*E* + 1	2.11089*E* + 01	1.00339*E* + 02	3.23142*E* + 01	2.58546*E* + 01
	Mean	0*E*000	4.15771*E* + 1	4.12895*E* + 01	2.51448*E* + 02	4.71632*E* + 01	4.25551*E* + 01
*F*12	Std	0*E*000	6.66764*E0*	6.63775*E* + 00	3.28548*E* + 01	9.63942*E* + 00	2.35036*E* + 01
	Median	0*E*000	4.38653*E* + 1	4.37479*E* + 01	2.37401*E* + 02	5.12106*E* + 01	4.03213*E* + 01
	Mean	5.02454*E* + 1	5.02454*E* + 1	5.00688*E* + 01	1.18218*E* + 02	5.53378*E* + 01	7.03207*E* + 01
*F*13	Std	4.77167*E*0	4.77167*E*0	4.36107*E* + 00	3.84270*E* + 01	1.90461*E* + 01	1.04847*E* + 01
	Median	4.9748*E* + 1	4.9748*E* + 1	4.97403*E* + 01	2.18707*E* + 01	6.35530*E* + 01	6.08155*E* + 01
	Mean	2.039144*E* + 1	2.039144*E* + 1	2.03699*E* + 01	3.10676*E* + 01	2.11816*E* + 01	3.42132*E* + 01
*F*14	Std	2.77772*E*0	2.77772*E*0	2.69322*E* + 00	1.81535*E* + 01	1.38456*E* + 01	1.62269*E* + 01
	Median	1.99736*E* + 1	1.99736*E* + 1	1.96490*E* + 01	2.93910*E* + 01	2.40104*E* + 01	2.46657*E* + 01
	Mean	6.0258*E*0	6.0258*E*0	6.21510*E* + 00	8.39310*E* + 01	1.23683*E* + 01	3.16915*E* + 01
*F*15	Std	2.6258*E*0	2.6258*E*0	3.03159*E* + 00	5.41295*E* + 01	1.08439*E* + 01	2.19687*E* + 01
	Median	6.8994*E*0	6.8994*E*0	7.16581*E* + 00	7.72715*E* + 01	2.10405*E* + 01	1.87566*E* − 01
	Mean	1.15262*E* + 1	1.15262*E* + 1	1.12527*E* + 01	1.00618*E* + 02	1.77924*E* + 01	1.72539*E* + 01
*F*16	Std	2.05534*E* − 1	2.05534*E* − 1	5.73736*E* − 02	3.36218*E* + 01	1.49513*E* + 00	3.05312*E* + 00
	Median	1.15263*E* + 1	1.15263*E* + 1	1.11540*E* + 01	8.14009*E* + 01	1.60481*E* + 01	1.59870*E* + 01

Table 5: Evaluation results on eight functions ($F9$–$F16$, $D = 30$).

Test function	Performance	LPSO	MCPSO	PS2O	HSPSO	CCPSO-ISM	FDR-PSO	FIPPSO
	Mean	1.51861*E* + 1	9.14724*E* + 1	8.93109*E* + 1	5.48581*E* − 5	0*E*000	1.14700*E* + 1	4.37977*E* + 1
*F*9	Std	2.11233 *E0*	3.16411*E* + 1	2.78157*E* + 1	9.60707*E* − 5	0*E*000	3.48441*E*0	1.30299*E* + 1
	Median	1.42608*E* + 1	1.02688*E* + 2	7.82676*E* + 1	4.08772*E* − 5	0*E*000	1.4954*E* + 1	4.53087*E* + 1
	Mean	5.77869*E* + 1	1.04384*E* + 2	9.43819*E* + 1	4.63460*E* + 1	8.12661*E* − 1	3.29247*E*0	4.04253*E* + 1
*F*10	Std	2.51335*E* + 1	8.82423*E* + 1	7.60825*E* + 1	1.56922*E* + 1	9.1512*E* − 2	2.6840*E* − 1	1.00620*E* + 1
	Median	5.49712*E* + 1	8.06500*E* + 1	9.38245*E* + 1	4.52979*E* + 1	9.2231*E* − 1	3.14882*E*0	4.30543*E* + 1
	Mean	2.52855*E* + 1	2.64517*E* + 1	2.30206*E* + 1	2.88386*E* + 1	2.47033*E* + 1	2.9646*E* + 1	2.60013*E* + 1
*F*11	Std	1.19867*E* − 2	4.00292*E* − 2	8.27323*E* − 2	6.71221*E* − 3	7.56920*E* − 2	0.00326*E* − 2	5.35043*E* − 2
	Median	2.59809*E* + 1	2.61515*E* + 1	2.66347*E* + 1	2.57792*E* + 1	2.62502*E* + 1	2.37816*E* + 1	2.20501*E* + 1
	Mean	4.54933*E* + 1	6.47593*E* + 1	5.76266*E* + 1	4.63969*E* + 1	0*E*000	4.45944*E* + 1	5.89902*E* + 1
*F*12	Std	1.64122*E* + 1	5.81843*E*0	1.26950*E* + 1	1.55139*E* + 1	0*E*000	1.72914*E* + 1	1.97771*E* + 1
	Median	2.51570*E* + 1	6.39535*E* + 1	6.48912*E* + 1	4.56778*E* + 1	0*E*000	4.08652*E* + 1	5.92656*E* + 1
	Mean	2.77699*E* + 2	1.03733*E*2	1.24333*E* + 2	1.95113*E* + 2	1.32693*E* + 2	5.97127*E* + 1	1.69809*E* + 2
*F*13	Std	2.56696*E* + 1	3.96304*E* + 1	3.59524*E* + 1	2.09328*E* + 1	7.97868*E*0	1.32038*E* + 1	2.86377*E0*
	Median	2.82621*E* + 2	1.17977*E*2	1.09972*E* + 2	1.60887*E* + 2	1.15773*E* + 2	5.86959*E* + 1	1.85220*E* + 2
	Mean	2.84475*E* + 1	2.76923*E* + 1	3.21626*E* + 1	2.09334*E* + 1	2.59571*E* + 1	2.26436*E* + 1	3.56754*E* + 1
*F*14	Std	3.08439*E*0	3.90532*E*0	6.36128*E*0	2.52636*E*0	2.11251*E*0	4.14451*E*0	1.20684*E*0
	Median	21.93042*E* + 1	2.73032*E* + 1	3.17851*E* + 1	2.17317*E* + 1	2.68011*E* + 1	1.13267*E* + 1	3.00738*E* + 1
	Mean	8.79677*E*0	1.25428*E* + 1	2.38048*E* + 1	8.78020*E*0	7.60963*E*0	9.25348*E*0	5.95142*E* + 1
*F*15	Std	6.38876*E*0	9.88228*E*0	7.10735*E*0	3.24385*E*0	3.79567*E*0	6.44398*E*0	1.96684*E* + 1
	Median	8.23932*E*0	8.94184*E*0	2.77004*E*0	7.13170*E*0	6.39411*E*0	9.71773*E*0	5.82241*E* + 1
	Mean	1.69691*E*0	1.10219*E* + 1	1.22394*E* + 1	1.34479*E*0	1.89622*E* + 1	1.50032*E* + 1	1.70448*E* + 1
*F*16	Std	1.451662*E*0	7.17067*E* − 1	2.77381*E* − 1	1.71356*E*0	2.1180*E* − 1	2.69315*E* − 1	0.18570*E* − 1
	Median	1.325485*E*0	1.02227*E* + 1	1.25632*E* + 1	1.85263*E* + 1	1.73979*E* + 1	1.77383*E* + 1	1.09972*E* + 1

Comparing Tables 2 to 7, it can be seen that E-S-HPSO has improved the precision optimization effect compared with HPSO, while there is no obvious improvement. Because the algorithm in this paper is mainly to improve the speed of the algorithm, the next section focuses on the speed of the E-S-HPSO.

Table 6: Evaluation results on eight functions ($F17$–$F22$, $D = 30$).

Test function	Performance	Best	HPSO	E-S-HPSO	S-E-HPSO	E-E-HPSO	S-S-HPSO
	Mean	4.30154$E+2$	5.21984$E+2$	4.52073$E+02$	5.89596$E+02$	5.39387$E+02$	5.30223$E+02$
$F17$	Std	7.89852$E+1$	4.07938$E+1$	4.10052$E+01$	6.97003$E+01$	4.13033$E+01$	4.87111$E+01$
	Median	4.41121$E+2$	5.22974$E+2$	4.52021$E+02$	5.90155$E+02$	5.26600$E+02$	5.26257$E+02$
	Mean	9.58097$E+1$	9.58097$E+1$	9.54203$E+01$	1.65324$E+02$	1.21691$E+02$	1.00038$E+02$
$F18$	Std	1.87319$E+1$	1.87319$E+1$	1.85202$E+01$	2.55312$E+01$	3.32608$E+01$	2.44135$E+01$
	Median	1.03718$E+2$	1.03718$E+2$	1.03673$E+02$	1.29197$E+02$	1.29064$E+02$	1.10376$E+02$
	Mean	8.83331$E+1$	8.83331$E+1$	8.81999$E+01$	1.60737$E+02$	9.46153$E+01$	1.13171$E+02$
$F19$	Std	3.57758$E+1$	3.57758$E+1$	3.56990$E+01$	6.25591$E+01$	5.23445$E+01$	4.87242$E+01$
	Median	7.30900$E+1$	7.30900$E+1$	7.29495$E+01$	1.57529$E+02$	9.19865$E+01$	1.05173$E+02$
	Mean	8.91346$E+2$	9.80514$E+2$	9.20294$E+02$	1.41496$E+03$	9.84232$E+02$	1.01776$E+03$
$F20$	Std	2.34994$E+1$	2.29816$E+1$	2.27180$E+01$	1.01034$E+02$	3.76923$E+01$	3.21281$E+01$
	Median	8.88269$E+2$	9.80231$E+2$	9.30002$E+02$	1.34776$E+03$	1.00582$E+03$	1.00974$E+03$
	Mean	8.95216$E+2$	9.69132$E+2$	9.98694$E+02$	1.02987$E+03$	1.00845$E+03$	9.71668$E+02$
$F21$	Std	1.90048$E+1$	2.52797$E+1$	2.50207$E+01$	4.44542$E+01$	2.48415$E+01$	4.96973$E+01$
	Median	8.93981$E+2$	9.67748$E+2$	9.87276$E+02$	1.04159$E+03$	1.00237$E+03$	1.00512$E+03$
	Mean	8.73037$E+2$	8.73037$E+2$	8.72718$E+02$	1.00006$E+03$	9.71034$E+02$	9.12413$E+02$
$F22$	Std	1.02568$E+2$	1.02568$E+2$	1.02089$E+02$	1.42273$E+02$	1.31093$E+02$	1.36926$E+02$
	Median	8.35297$E+2$	8.35297$E+2$	8.35177$E+02$	1.01747$E+03$	9.95780$E+02$	9.16719$E+02$

Table 7: Evaluation results on eight functions ($F17$–$F22$, $D = 30$).

Test function	Performance	LPSO	MCPSO	PS2O	HSPSO	CCPSO-ISM	FDR-PSO	FIPPSO
	Mean	5.42349$E+2$	5.44900$E+2$	5.35993$E+2$	4.30046$E+2$	5.41884$E+2$	5.79451$E+2$	4.75136$E+2$
$F17$	Std	9.25305$E+1$	8.28155$E+1$	8.35173$E+1$	7.46879$E+1$	4.88435$E+1$	5.84707$E+1$	8.95450$E+1$
	Median	5.92930$E+2$	5.74521$E+2$	5.75955$E+2$	46.83026$E+2$	5.17943$E+2$	5.96367$E+2$	4.20985$E+2$
	Mean	2.02823$E+2$	1.81841$E+2$	4.17035$E+2$	11.46691$E+2$	1.74437$E+2$	4.12716$E+2$	2.23382$E+2$
$F18$	Std	1.65636$E+2$	7.39306$E+1$	9.12702$E+1$	2.52999$E+1$	2.51741$E+1$	1.38824$E+1$	1.62433$E+1$
	Median	3.49331$E+2$	1.25868$E+2$	4.89381$E+2$	1.76458$E+2$	1.18682$E+2$	4.88089$E+2$	1.14105$E+2$
	Mean	2.36447$E+2$	2.36140$E+2$	3.95316$E+2$	1.30640$E+2$	1.34501$E+2$	3.89656$E+2$	2.76803$E+2$
$F19$	Std	4.17344$E+1$	5.93178$E+1$	1.70563$E+2$	7.27530$E+1$	2.55929$E+1$	2.22961$E+2$	7.94407$E+1$
	Median	2.28387$E+2$	1.98377$E+2$	3.97688$E+2$	2.52125$E+2$	1.60227$E+2$	4.06332$E+2$	2.14892$E+2$
	Mean	9.57081$E+2$	9.43144$E+2$	1.34425$E+3$	9.54769$E+2$	9.49275$E+2$	9.06064$E+2$	8.96165$E+2$
$F20$	Std	1.0362$E+1$	5.34765$E+1$	9.39555$E+1$	3.96348$E+1$	6.17441$E+1$	1.40514$E+1$	2.64296$E+1$
	Median	9.44736$E+2$	9.18328$E+2$	9.69140$E+2$	9.96919$E+2$	9.87829$E+2$	9.44939$E+2$	8.82621$E+2$
	Mean	9.48267$E+2$	9.60789$E+2$	1.08716$E+3$	9.04834$E+2$	9.66728$E+2$	9.67694$E+2$	8.29230$E+2$
$F21$	Std	1.12561$E+1$	6.62315$E+1$	4.18593$E+1$	1.04182$E+1$	2.56491$E+1$	1.02051$E+1$	1.03564$E+1$
	Median	9.02383$E+2$	9.51875$E+2$	1.12016$E+3$	9.25798$E+2$	9.89399$E+2$	9.47878$E+2$	8.90258$E+2$
	Mean	9.30376$E+2$	9.48384$E+2$	1.11354$E+3$	9.66485$E+2$	9.82281$E+2$	9.90088$E+2$	8.03380$E+2$
$F22$	Std	1.90752$E+1$	6.28362$E+1$	7.43989$E+1$	1.21847$E+1$	2.82631$E+1$	1.41154$E+1$	1.19903$E+1$
	Median	9.86126$E+2$	9.49257$E+2$	1.49093$E+3$	9.42666$E2$	9.22614$E+2$	9.80798$E+2$	8.73846$E+2$

7.2. Comparison of Performance between AHPSO and HPSO. Through theoretical analysis and simulation verification above, it can be seen that E-S-HPSO has better performance than other four algorithms such as HPSO, which verifies the correctness and advantage of AHPSO's new grouping strategy. To compare the performance of the algorithm, 22 standard test functions of $D = 30$ are optimized. First, AHPSO algorithm is used to optimize each test function 30 times, and the optimization result and iteration number of algorithm termination are recorded. Also, the *ON* parameters of 30 times of the algorithm termination are calculated. The HPSO algorithm is tested in two groups. In the first group, the number of iterations of HPSO algorithm is set to be 10^6, and the iteration ends when it is carried out *ON times*. Besides, the optimization result is recorded as Result 1. In the second group, the average number of iterations of HPSO algorithm is set to be *ON*, and the optimization result is recorded as Result 2. The results of algorithm are shown in Tables 9–11.

It is suggested by comparing the above 8 functions that this method can find the optimal value faster than the setting of 10^6 search times. Under the same number of times, the efficiency of HPSO algorithm is not as good as that of AHPSO algorithm.

The test results of most of the functions in the above experiments are stronger than HPSO, and the performance improvement is related to the improvement of grouping strategy. Although the test results of $F12$ and $F15$ are slightly

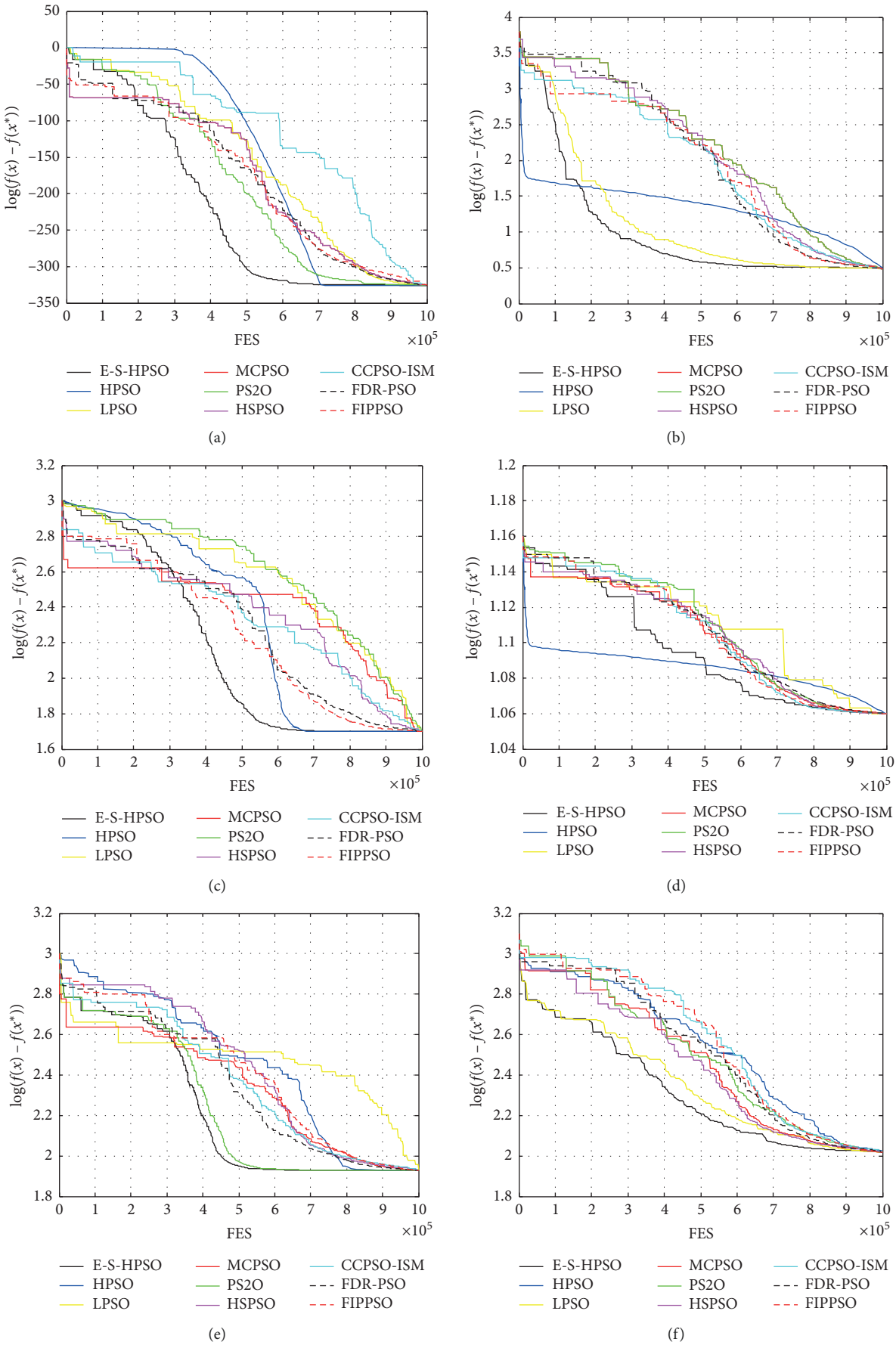

Figure 6: Convergence characteristics and performance for E-S-HPSO and others. (a) $F1(x)$. (b) $F6(x)$. (c) $F13(x)$. (d) $F16(x)$. (e) $F18(x)$. (f) $F19(x)$.

Table 8: Significant test result.

Test function	HPSO	LPSO	MCPSO	PS2O	HSPSO	CCPSO-ISM	FDR-PSO	FIPPSO
$F1$	1	0.3485	0.656	0.04218	0.1419	0.4976	1	0.28003
$F6$	0.8147	0.6874	0.0147	0.0097	0.7058	0.0378	0.7134	0.6597
$F13$	0.1576	0.0976	0.0957	0.0854	0.0890	0.0792	0.0959	0.0682
$F16$	0.6557	0.1659	0.8491	0.6768	0.1357	0.3922	0.6555	0.7722
$F18$	0.7060	0.0318	0.0276	0.0462	0.0371	0.0235	0.0486	0.0317
$F19$	0.6387	0.0381	0.0276	0.0295	0.0186	0.0314	0.0244	0.0216

Table 9: Evaluation results on eight functions ($F1$–$F8$, $D = 30$).

Test function	Performance	Best	HPSO	AHPSO	ON	Result 1	Result 2
	Mean	$0E000$	$0E000$	$0E000$	$5.7151E+05$	$0E000$	$0E000$
$F1$	Std	$0E000$	$0E000$	$0E000$	$4.8890E+04$	$0E000$	$0E000$
	Median	$0E000$	$0E000$	$0E000$	$6.5965E+05$	$0E000$	$0E000$
	Mean	$0E000$	$0E000$	$0E000$	$7.8449E+05$	$0E000$	$0E000$
$F2$	Std	$0E000$	$0E000$	$0E000$	$7.0347E+04$	$0E000$	$0E000$
	Median	$0E000$	$0E000$	$0E000$	$6.9769E+05$	$0E000$	$0E000$
	Mean	$9.6356E-09$	$9.6356E-09$	$9.1614E-09$	$7.3266E+05$	$1.0616E+03$	$1.2739E+03$
$F3$	Std	$4.5948E-08$	$4.5948E-08$	$5.1488E-08$	$7.2689E+03$	$2.3505E+03$	$2.8205E+03$
	Median	$1.3141E-16$	$1.31414E-16$	$4.7461E-16$	$7.9314E+05$	$6.1560E+02$	$7.3872E+02$
	Mean	$4.0785E+06$	$4.0785E+06$	$4.0784E+06$	$8.6078E+05$	$4.0793E+06$	$4.0794E+06$
$F4$	Std	$4.4282E+06$	$4.4282E+06$	$4.4281E+06$	$3.0344E+03$	$4.4284E+06$	$4.4284E+06$
	Median	$5.2877E+06$	$5.2877E+06$	$5.2874E+06$	$8.7140E+05$	$5.2885E+06$	$5.2887E+06$
	Mean	$6.6478E+03$	$6.6478E+03$	$6.5171E+03$	$7.7902E+05$	$7.4126E+03$	$7.5656E+03$
$F5$	Std	$1.3453E+03$	$1.3453E+03$	$8.7452E+02$	$2.9387E+03$	$2.7475E+03$	$3.0280E+03$
	Median	$6.6430E+03$	$6.6430E+03$	$6.5619E+03$	$7.7859E+05$	$8.0654E+03$	$8.3499E+03$
	Mean	$3.6620E+00$	$3.6620E+00$	$3.6524E+00$	$7.4180E+05$	$4.9186E+02$	$5.8949E+02$
$F6$	Std	$4.3480E+00$	$4.3480E+00$	$4.6404E+00$	$7.8728E+03$	$1.8172E+02$	$2.1720E+02$
	Median	$2.4078E+04$	$2.4078E+04$	$1.1016E-01$	$8.0959E+05$	$1.9629E+02$	$2.3550E+02$
	Mean	$6.2172E-15$	$2.1320E-07$	$0E000$	$8.0636E+05$	$1.4193E+03$	$1.7032E+03$
$F7$	Std	$0E000$	$3.8009E-07$	$0E000$	$8.8840E+03$	$2.9158E+02$	$3.4990E+02$
	Median	$6.2172E-15$	$6.2172E-15$	$0E000$	$7.7686E+05$	$1.9781E+02$	$2.3737E+02$
	Mean	$0E000$	$7.4000E-04$	$5.8711E-05$	$7.0170E+05$	$1.5877E+03$	$1.9052E+03$
$F8$	Std	$0E000$	$2.2570E-03$	$6.1175E-03$	$1.1471E+04$	$8.0447E+02$	$9.6536E+02$
	Median	$0E000$	$0E000$	$0E000$	$8.1521E+05$	$6.9662E+02$	$8.3595E+02$

worse than those of HPSO, they are still at the same level. The results of the algorithm are also acceptable.

From the results, the performance of most functions is still better than that of HPSO. Although the performance of $F18$ and $F19$ is a little worse, the difference is not large. However, the average number of iterations of AHPSO is $7.8 * 10^5$, saving more than 20% calculation compared with HPSO. This is because the grouping strategy changed by the algorithm in this study can improve the algorithm speed, and in the meantime, the adaptive state switching can make it easier to get rid of local optimization and decline faster. The adaptive termination can make the algorithm in this study no longer set the number of iterations and expand the applicable scope of the algorithm. Compared with Result 1 and Result 2, the result of the algorithm here is about two orders of magnitude higher than AHPSO. It can be seen that the descent speed of this algorithm improves the applicability of this algorithm.

To clearly compare the performance of algorithms, convergence characteristics and performance for AHPSO and HPSO on different functions are made, as shown in Figure 7.

Figure 7 directly suggests that under the same number of iterations, AHPSO algorithm decreases faster, achieves convergence earlier, and improves the applicable scope of the algorithm. To clearly show the performance of AHPSO and compare with Figure 8 in [1], population distribution at various stages for AHPSO of testing function $F7$ is made (Figure 8).

Compared with HPSO algorithm requiring 10000 iterations, AHPSO algorithm only needs 73,576 iterations to achieve optimization. This is because the adaptive strategy of AHPSO algorithm is not only a condition to recognize state switching but also can make the particle get rid of local maximum quickly. The faster the algorithm's fitness function decreases, the faster the optimization efficiency will be.

The complexity of the two algorithms is analyzed and compared. For the complex optimization function, the number of grouping and regrouping of the two algorithms is far less than the number of optimization iteration, and the

Table 10: Evaluation results on eight functions ($F9$–$F16$, $D = 30$).

Test function	Performance	Best	HPSO	AHPSO	*ON*	Result 1	Result 2
	Mean	0*E*000	1.39034*E* + 1	1.3637*E* + 01	7.6956*E* + 05	1.4707*E* + 02	6.0341*E* + 02
*F*9	Std	0*E*000	3.83982*E*0	3.4930*E* + 00	1.0933*E* + 04	1.7723*E* + 02	2.3003*E* + 02
	Median	0*E*000	1.42459*E* + 1	1.3464*E* + 01	7.7908*E* + 05	4.0518*E* + 02	3.9887*E* + 02
	Mean	9.3786*E* − 1	4.15448*E* + 1	3.9882*E* + 01	7.3786*E* + 05	8.7292*E* + 02	6.2453*E* + 02
*F*10	Std	9.7714*E* − 2	1.30013*E* + 1	1.1395*E* + 01	1.1093*E* + 04	8.1637*E* + 02	2.6481*E* + 02
	Median	9.87329*E* − 1	2.07210*E* + 1	2.0600*E* + 01	7.1851*E* + 05	8.1192*E* + 01	3.1116*E* + 02
	Mean	2.08050*E* + 1	2.08050*E* + 1	2.0006*E* + 01	6.8554*E* + 05	4.2006*E* + 02	6.3790*E* + 02
*F*11	Std	8.38112*E* − 2	8.38112*E* − 2	9.6994*E* − 01	8.6365*E* + 03	5.2696*E* + 02	2.6536*E* + 02
	Median	2.07210*E* + 1	2.07210*E* + 1	1.9887*E* + 01	7.0725*E* + 05	4.3752*E* + 02	8.4510*E* + 02
	Mean	0*E*000	4.15771*E* + 1	5.0263*E* + 01	7.1762*E* + 05	6.9844*E* + 02	1.0242*E* + 03
*F*12	Std	0*E*000	6.66764*E*0	5.4117*E*0	7.7359*E* + 02	6.3464*E* + 02	7.3692*E* + 02
	Median	0*E*000	4.38653*E* + 1	4.3281*E* + 01	8.0754*E* + 05	3.3585*E* + 02	3.8774*E* + 02
	Mean	5.02454*E* + 1	5.02454*E* + 1	4.9382*E* + 01	7.6301*E* + 05	4.8190*E* + 02	6.3431*E* + 02
*F*13	Std	4.77167*E*0	4.77167*E*0	4.7407*E* + 00	1.2141*E* + 04	2.0259*E* + 01	1.1254*E* + 02
	Median	4.9748*E* + 1	4.9748*E* + 1	4.7780*E* + 01	7.4481*E* + 05	1.0338*E* + 03	9.5606*E* + 02
	Mean	2.039144*E* + 1	2.039144*E* + 1	2.0057*E* + 01	7.8154*E* + 05	1.8756*E* + 02	9.0005*E* + 02
*F*14	Std	2.77772*E*0	2.77772*E*0	2.5653*E* + 00	1.1135*E* + 04	1.0899*E* + 02	8.2054*E* + 02
	Median	1.99736*E* + 1	1.99736*E* + 1	1.9229*E* + 01	7.4034*E* + 05	3.9238*E* + 02	2.8070*E* + 02
	Mean	6.0258*E*0	6.0258*E*0	7.6296*E* + 00	7.6277*E* + 05	2.0414*E* + 02	6.0038*E* + 02
*F*15	Std	2.6258*E*0	2.6258*E*0	1.6464*E* + 00	6.8493*E* + 01	4.9231*E* + 02	2.5138*E* + 01
	Median	6.8994*E*0	6.8994*E*0	7.2204*E* + 00	7.1540*E* + 05	3.4639*E* + 02	4.3216*E* + 02
	Mean	1.15262*E* + 1	1.15262*E* + 1	9.6229*E* + 00	7.4880*E* + 05	9.6316*E* + 02	3.2425*E* + 02
*F*16	Std	2.05534*E* − 1	2.05534*E* − 1	1.6351*E* + 00	1.5326*E* + 04	9.2054*E* + 02	1.6169*E* + 02
	Median	1.15263*E* + 1	1.15263*E* + 1	1.1421*E* + 01	7.7511*E* + 05	6.4203*E* + 01	1.9029*E* + 02

Table 11: Evaluation results on eight functions ($F9$–$F16$, $D = 30$).

Test function	Performance	Best	HPSO	AHPSO	*ON*	Result 1	Result 2
	Mean	4.30154*E* + 2	5.21984*E* + 2	5.2112*E* + 02	7.9160*E* + 05	9.6207*E* + 02	8.0948*E* + 02
*F*17	Std	7.89852*E* + 1	4.07938*E* + 1	3.9018*E* + 01	1.4814*E* + 04	5.6794*E* + 02	1.3191*E* + 02
	Median	4.41121*E* + 2	5.22974*E* + 2	5.2219*E* + 02	8.2430*E* + 05	9.8040*E* + 02	1.0992*E* + 03
	Mean	9.58097*E* + 1	9.58097*E* + 1	1.0427*E* + 02	7.9529*E* + 05	9.7118*E* + 02	7.7917*E* + 02
*F*18	Std	1.87319*E* + 1	1.87319*E* + 1	1.7938*E* + 01	1.5549*E* + 03	5.3678*E* + 02	5.6533*E* + 02
	Median	1.03718*E* + 2	1.03718*E* + 2	1.0210*E* + 02	8.1079*E* + 05	1.0473*E* + 03	5.2945*E* + 02
	Mean	8.83331*E* + 1	8.83331*E* + 1	9.6823*E* + 01	8.1977*E* + 05	7.2604*E* + 02	7.3278*E* + 02
*F*19	Std	3.57758*E* + 1	3.57758*E* + 1	3.5021*E* + 01	8.1855*E* + 03	9.9347*E* + 02	6.8339*E* + 02
	Median	7.30900*E* + 1	7.30900*E* + 1	8.2658*E* + 01	7.5919*E* + 05	3.1380*E* + 02	7.5211*E* + 02
	Mean	8.91346*E* + 2	9.80514*E* + 2	9.7893*E* + 02	7.8212*E* + 05	1.6566*E* + 03	1.6163*E* + 03
*F*20	Std	2.34994*E* + 1	2.29816*E* + 1	2.1083*E* + 01	2.9259*E* + 03	3.1205*E* + 02	9.6816*E* + 02
	Median	8.88269*E* + 2	9.80231*E* + 2	9.7958*E* + 02	8.4515*E* + 05	1.6520*E* + 03	1.1892*E* + 03
	Mean	8.95216*E* + 2	9.69132*E* + 2	9.6779*E* + 02	7.8840*E* + 05	1.6643*E* + 03	1.6784*E* + 03
*F*21	Std	1.90048*E* + 1	2.52797*E* + 1	2.4402*E* + 01	5.4079*E* + 03	9.3272*E* + 01	2.6151*E* + 02
	Median	8.93981*E* + 2	9.67748*E* + 2	9.6608*E* + 02	8.1902*E* + 05	1.2225*E* + 03	1.0871*E* + 03
	Mean	8.73037*E* + 2	8.73037*E* + 2	8.7150*E* + 02	8.4696*E* + 05	1.0971*E* + 03	1.4803*E* + 03
*F*22	Std	1.02568*E* + 2	1.02568*E* + 2	1.0223*E* + 02	3.0864*E* + 03	7.7040*E* + 02	5.5271*E* + 02
	Median	8.35297*E* + 2	8.35297*E* + 2	8.3357*E* + 02	7.7373*E* + 05	1.6797*E* + 03	1.2940*E* + 03

calculation quantity is negligible. The particle of both algorithms is updated based on equations (1) and (2), that is, the search strategy is the same, and the algorithm complexity of particle state update is the same. In the HPSO algorithm, the number of iterations should be set. After each iteration, the maximum number of iterations should be determined, that is, one addition and one judgment should be made. In AHPSO algorithm, although it is a double judgment condition, if the first judgment condition is not reached, it will return the search. Besides, the calculation is primarily for the first judgment. In the switching from extensive search to precise search, the first criterion is equation (12), i.e., two subtractions, one multiplication, and one judgment are needed. From accurate search to extensive search or algorithm termination, the first discriminant formula is equation (13), which needs a subtraction and a judgment. The

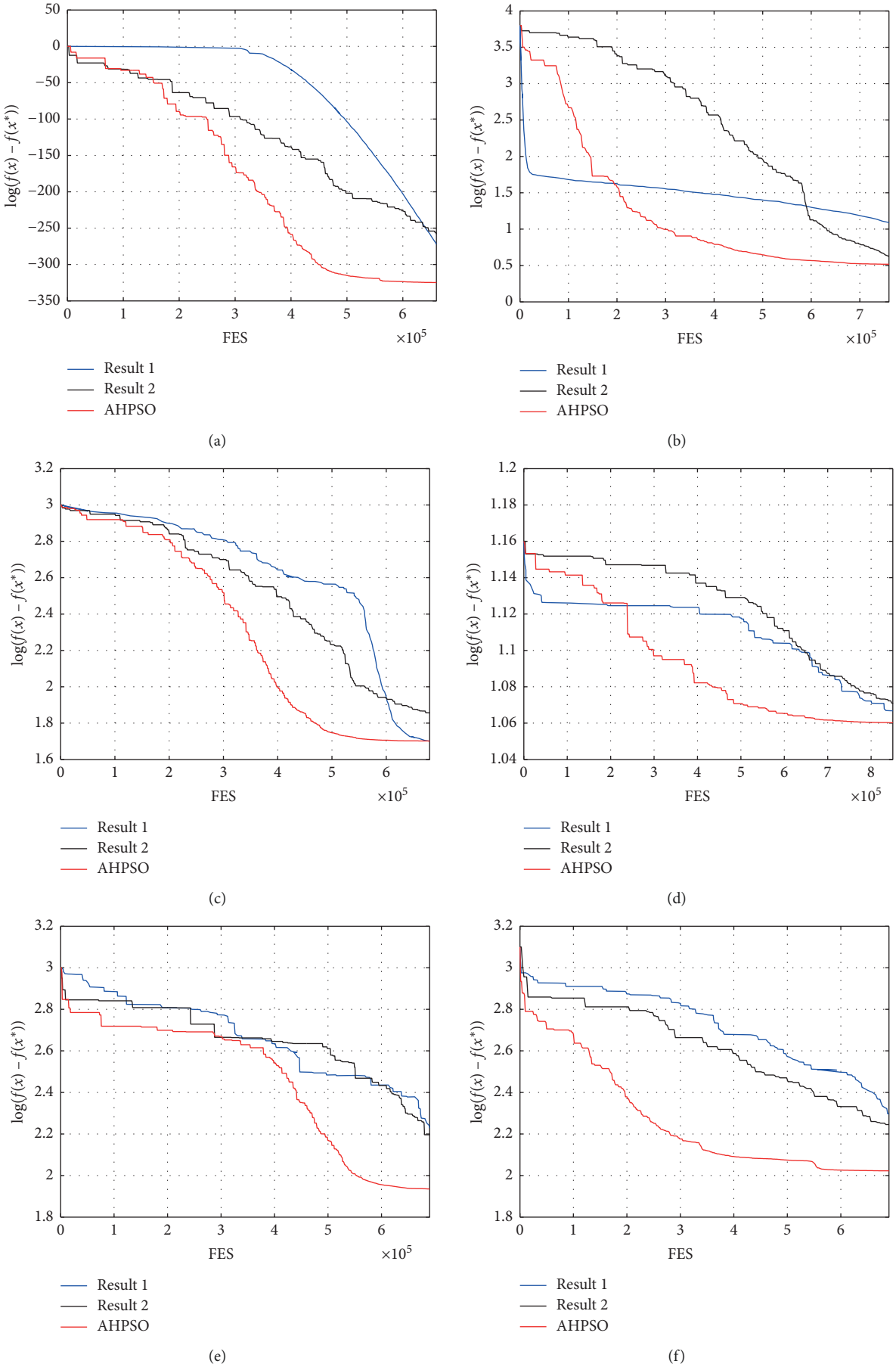

Figure 7: Convergence characteristics and performance for AHPSO and HPSO. (a) $F1(x)$. (b) $F6(x)$. (c) $F13(x)$. (d) $F16(x)$. (e) $F18(x)$. (f) $F19(x)$.

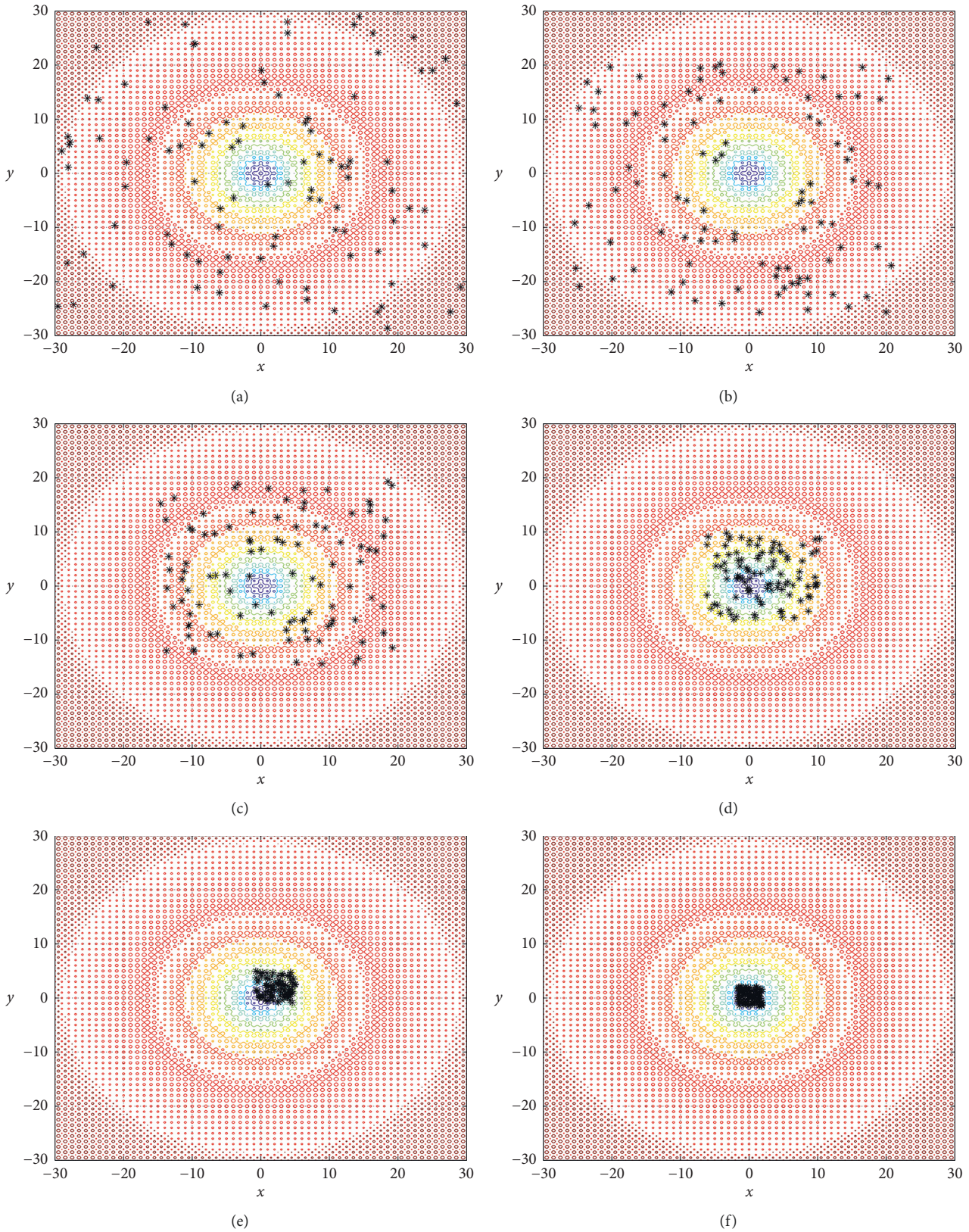

Figure 8: Population distribution at various stages for AHPSO. (a) Iteration = 1. (b) Iteration = 15000. (c) Iteration = 30000. (d) Iteration = 45000. (e) Iteration = 60000. (f) Iteration = 73576.

computational complexity of the sum of all particles in equations (1) and (2) is much higher than that of AHPSO and HPSO. Although the amount of AHPSO's discriminant part is slightly higher than that of HPSO, the magnitude of the amount of extra computation has little impact on single iteration, i.e., the amount of calculation of two algorithms is similar in single iteration. The above figure suggests that AHPSO has a faster convergence speed than HPSO, nearly 78% of the number of HPSO iterations, saving 20% or more of the calculation time, which is of great significance to the optimization of multiple complex functions.

As AHPSO adopts a new and different grouping strategy, the algorithm rate is improved, and the switching state can be adaptive, making it easier to jump out of the local optimum. Therefore, the convergence speed of the algorithm is further improved. From the algorithm complexity analysis and simulation verification, it can be seen that the algorithm efficiency is significantly improved.

8. Conclusions

In this study, the HPSO algorithm is briefly reviewed, and the grouping strategy of HPSO algorithm itself is sorted out. The three points of setting the number of iterations and switching time of search strategy can be further studied, so an improved AHPSO algorithm is proposed. The main work is as follows:

(1) The average grouping strategy is adopted in the HPSO algorithm, which is fast and easy to implement but has little effect on the performance improvement of the algorithm. Therefore, this study employs the grouping strategy of information entropy method and system clustering method for reference to improve the grouping mode of particles and arrange and combine the grouping mode and search strategy. The advantages of grouping strategy are verified through theoretical analysis and simulation.

(2) In the HPSO algorithm, the first 80% of iterations adopt the extensive search strategy, and the last 20% adopt the precise search strategy. This ratio has a larger relationship to the algorithm, and this determined ratio is bound to be impossible to apply to all problems. Accordingly, this study combines the absolute value of slope of particle fitness function to construct the state switching criterion and realize the transition from extensive search to precise search.

(3) HPSO searching strategy can switch only once. If the particle is trapped in the local optimal solution before switching or is not close to the optimal solution, the efficiency of the algorithm after switching must be affected. In this study, by combining the absolute value of the slope of the fitness function and the state information after the regrouping of particles, double discriminant conditions are constructed to realize reverse switching or adaptive termination of the search strategy and form closed loop feedback. As a result, the performance of the algorithm is improved.

(4) The concept of AHPSO almost differs from that of HPSO. HPSO algorithm should set the number of iterations and set the ratio of different search modes according to the number of iterations, while AHPSO constructs the condition of switching between different states. After the AHPSO starts, the algorithm can be switched to the exact search, and then the reverse switch of the search strategy can be realized, or the algorithm adaptively terminates. The efficiency of the algorithm is improved by closed loop, and the ratio between the number of iterations and the search mode is not required to be preset. Thus, the applicability and efficiency of the algorithm are improved.

Data Availability

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

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (Grant nos. 61502522, 61502523, 61871397, and 61501505), Equipment Pre-Research Field Fund (Grant no. JZX7Y20190253036101), Equipment Pre-Research Ministry of Education Joint Fund (Grant no. 6141A02033703), and Hubei Provincial Natural Science Foundation (Grant no. 2019CFC897).

References

[1] M. Roshanzamir, M. A. Balafar, and S. N. Razavi, "Empowering particle swarm optimization algorithm using multi agents' capability: a holonic approach," *Knowledge-Based Systems*, vol. 136, pp. 58–74, 2017.

[2] R. Cheng, M. Li, K. Li, and X. Yao, "Evolutionary multiobjective optimization based multimodal optimization: fitness landscape approximation and peak detection," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 5, pp. 692–706, 2018.

[3] S. Saremi, S. Mirjalili, A. Lewis, A. W. C. Liew, and J. S. Dong, "Enhanced multi-objective particle swarm optimisation for estimating hand postures," *Knowledge-Based Systems*, vol. 158, pp. 175–195, 2018.

[4] A. Ponsich, A. L. Jaimes, and C. A. C. Coello, "A survey on multiobjective evolutionary algorithms for the solution of the portfolio optimization problem and other finance and economics applications," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 3, pp. 321–344, 2013.

[5] J. Ding, C. Yang, Y. Jin, and T. Chai, "Generalized multitasking for evolutionary optimization of expensive problems," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 1, pp. 44–58, 2017.

[6] H. Wu, C. Nie, F.-C. Kuo, H. Leung, and C. J. Colbourn, "A discrete particle swarm optimization for covering array generation," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 4, pp. 575–591, 2015.

[7] W. Hu and G. G. Yen, "Adaptive multiobjective particle swarm optimization based on parallel cell coordinate system," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 1, pp. 1–18, 2015.

[8] G. Bramerdorfer, J. A. Tapia, J. J. Pyrhonen, and A. Cavagnino, "Modern electrical machine design optimization: techniques, trends, and best practices," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 10, pp. 7672–7684, 2018.

[9] J. H. Lee, J.-Y. Song, D.-W. Kim, J.-W. Kim, Y.-J. Kim, and S.-Y. Jung, "Particle swarm optimization algorithm with intelligent particle number control for optimal design of electric machines," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 2, pp. 1791–1798, 2018.

[10] M. Collotta, G. Pau, and V. Maniscalco, "A fuzzy logic approach by using particle swarm optimization for effective energy management in IWSNs," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 12, pp. 9496–9506, 2017.

[11] Z. Xue, H. Li, Y. Zhou, N. Ren, and W. Wen, "Analytical prediction and optimization of cogging torque in surface-mounted permanent magnet machines with modified particle swarm optimization," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 12, pp. 9795–9805, 2017.

[12] G. Lin and J. Guan, "Solving maximum set *k*-covering problem by an adaptive binary particle swarm optimization method," *Knowledge-Based Systems*, vol. 142, pp. 95–107, 2018.

[13] G. Lin and J. Guan, "A hybrid binary particle swarm optimization for the obnoxious p -median problem," *Information Sciences*, vol. 425, pp. 1–17, 2018.

[14] J. Liu, Y. Mei, and X. Li, "An analysis of the inertia weight parameter for binary particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 5, pp. 666–681, 2016.

[15] M. Hu, T. Wu, and J. D. Weir, "An adaptive particle swarm optimization with multiple adaptive methods," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 5, pp. 702–720, 2013.

[16] T. Bai, Y.-b. Kan, J.-x. Chang, Q. Huang, and F.-J. Chang, "Fusing feasible search space into PSO for multi-objective cascade reservoir optimization," *Applied Soft Computing*, vol. 51, pp. 328–340, 2017.

[17] T. X. Pham, P. Siarry, and H. Oulhadj, "Integrating fuzzy entropy clustering with an improved PSO for MRI brain image segmentation," *Applied Soft Computing*, vol. 65, pp. 230–242, 2018.

[18] Y. Yan, R. Zhang, J. Wang, and J. Li, "Modified PSO algorithms with "request and reset" for leak source localization using multiple robots," *Neurocomputing*, vol. 292, pp. 82–90, 2018.

[19] A. R. Jordehi, "Time varying acceleration coefficients particle swarm optimisation (TVACPSO): a new optimisation algorithm for estimating parameters of PV cells and modules," *Energy Conversion and Management*, vol. 129, pp. 262–274, 2016.

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

[21] N. Ghorbani, A. Kasaeian, A. Toopshekan, L. Bahrami, and A. Maghami, "Optimizing a hybrid wind-PV-battery system using GA-PSO and MOPSO for reducing cost and increasing reliability," *Energy*, vol. 154, pp. 581–591, 2018.

[22] S. Goudarzi, W. H. Hassan, M. H. Anisi et al., "ABC-PSO for vertical handover in heterogeneous wireless networks," *Neurocomputing*, vol. 256, pp. 63–81, 2017.

[23] M. S. Ran and Z. Mesut, "A recombination-based hybridization of particle swarm optimization and artificial bee colony algorithm for continuous optimization problems," *Applied Soft Computing*, vol. 13, no. 4, pp. 2188–2203, 2013.

[24] Y. Gupta and A. Saini, "A novel fuzzy-PSO term weighting automatic query expansion approach using combined semantic filtering," *Knowledge-Based Systems*, vol. 136, pp. 97–120, 2017.

[25] Z. Liu, C. Mao, J. Luo, Y. Zhang, and C. L. Philip Chen, "A three-domain fuzzy wavelet network filter using fuzzy PSO for robotic assisted minimally invasive surgery," *Knowledge-Based Systems*, vol. 66, pp. 13–27, 2014.

[26] F. Jiang, H. Xia, Q. Anh Tran, Q. Minh Ha, N. Quang Tran, and J. Hu, "A new binary hybrid particle swarm optimization with wavelet mutation," *Knowledge-Based Systems*, vol. 130, pp. 90–101, 2017.

[27] K. Chen, F. Zhou, L. Yin, S. Wang, Y. Wang, and F. Wan, "A hybrid particle swarm optimizer with sine cosine acceleration coefficients," *Information Sciences*, vol. 422, pp. 218–241, 2018.

[28] M. Issa, A. E. Hassanien, D. Oliva, A. Helmi, I. Ziedan, and A. Alzohairy, "ASCA-PSO: adaptive sine cosine optimization algorithm integrated with particle swarm for pairwise local sequence alignment," *Expert Systems With Applications*, vol. 99, pp. 56–70, 2018.

[29] Z.-H. Zhan, J. Zhang, Y. Li, and Y.-H. Shi, "Orthogonal learning particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 6, pp. 832–847, 2011.

[30] Y.-W Leung and Y. Wang, "An orthogonal genetic algorithm with quantization for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 5, no. 1, pp. 41–53, 2001.

[31] W. F. Gao, S.-Y. Liu, and L. L Huang, "A novel artificial bee colony algorithm based on modified search equation and orthogonal learning," *IEEE Transactions on Cybernetics*, vol. 43, no. 3, pp. 1011–1024, 2013.

[32] L. T. Al-Bahrani and J. C. Patra, "A novel orthogonal PSO algorithm based on orthogonal diagonalization," *Swarm and Evolutionary Computation*, vol. 40, pp. 1–23, 2018.

[33] P. Liu and J. Liu, "Multi-leader PSO (MLPSO): a new PSO variant for solving global optimization problems," *Applied Soft Computing*, vol. 61, pp. 256–263, 2017.

[34] T. Y. Tan, L. Zhang, S. C. Neoh, and C. P. Lim, "Intelligent skin cancer detection using enhanced particle swarm optimization," *Knowledge-Based Systems*, vol. 158, pp. 118–135, 2018.

[35] P. N. Suganthan, "Particle swarm optimiser with neighbourhood operator," in *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, vol. 3, pp. 1958–1962, IEEE, Washington, DC, USA, July 1999.

[36] H. Chen and Y. Zhu, "Optimization based on symbiotic multi-species coevolution," *Applied Mathematics and Computation*, vol. 205, no. 1, pp. 47–60, 2008.

[37] B. Niu, Y. Zhu, and X. He, "Multi-population cooperative particle swarm optimization," in *Proceedings of the European Conference on Artificial Life*, pp. 874–883, Springer, Canterbury, UK, September 2005.

[38] Y. Li, Z.-H. Zhan, S. Lin, J. Zhang, and X. Luo, "Competitive and cooperative particle swarm optimization with

information sharing mechanism for global optimization problems," *Information Sciences*, vol. 293, pp. 370–382, 2015.

[39] R.-L. Tang and Y.-J. Fang, "Modification of particle swarm optimization with human simulated property," *Neurocomputing*, vol. 153, pp. 319–331, 2015.

[40] R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: simpler, maybe better," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 204–210, 2004.

[41] T. Peram, K. Veeramachaneni, and C. K. Mohan, "Fitness-distance-ratio based particle swarm optimization," in *Proceedings of the 2003 IEEE Swarm Intelligence Symposium SIS'03 (Cat. No. 03EX706)*, pp. 174–181, IEEE, Indianapolis, IN, USA, April 2003.