Research Article

Multiobjective Particle Swarm Optimization Based on Ideal Distance

Shihua Wang, Yanmin Liu, Kangge Zou, Nana Li, and Yaowei Wu

1School of Mathematics and Statistics, Guizhou University, Guiyang 550025, China
2Zunyi Normal University, Zunyi 563002, China
3School of Data Science and Information Engineering, Guizhou Minzu University, Guiyang 550025, China
4School of Mathematics and Computational Statistics, Wuyi University, Jiangmen 529000, China

Correspondence should be addressed to Yanmin Liu; yanmin7813@163.com

Received 28 August 2021; Revised 2 December 2021; Accepted 2 March 2022; Published 11 April 2022

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

Recently, multiobjective particle swarm optimization (MOPSO) has been widely used in science and engineering, however how to effectively improve the convergence and distribution of the algorithm has always been a hot research topic on multiobjective optimization problems (MOPs). To solve this problem, we propose a multiobjective particle swarm optimization based on the ideal distance (IDMOPSO). In IDMOPSO, the adaptive grid and ideal distance are used to optimize and improve the selection method of global learning samples and the size control strategy of the external archive, and the fine-tuning parameters are introduced to adjust particle flight in the swarm dynamically. Additionally, to prevent the algorithm from falling into a local optimum, the cosine factor is introduced to mutate the position of the particles during the exploitation and exploration process. Finally, IDMOPSO, several other popular MOPSOs and MOEAs were simulated on the benchmarks functions to test the performance of the proposed algorithm using IGD and HV indicators. The experimental results show that IDMOPSO has the better convergence, diversity, and excellent solution ability compared to the other algorithms.

1. Introduction

Multiobjective optimization problems (MOPs) exist widely in engineering applications and scientific research problems [1]. Traditional methods have significant limitations when solving MOPs, therefore, scholars have proposed many excellent multiobjective optimization algorithms to solve complex MOPs [2–4]. PSO is an optimization algorithm based on swarm intelligence proposed by Kennedy et al. [5]. Due to its rapid convergence and few parameters, PSO has achieved suitable applications in the field of single-objective optimization. Coello et al. proposed multiobjective particle swarm optimization (MOPSO) [6] by integrating some strategies with the need for real problems.

At present, the challenges of using PSO to solve MOPs are as follows: the first one is how to maintain the size of external archives effectively, and the second one is how to select global learning samples that take into account diversity and convergence; the last one is how to balance the convergence and diversity of algorithm. In response to these challenges, scholars have made unremitting efforts and proposed many excellent improved MOPSOs [7–11]. For example, Han et al. [7] proposed an adaptive MOPSO based on a hybrid framework of solution distribution entropy and population interval information to improve the algorithm’s performance (AMOPSO), where a globally optimal solution selection mechanism based on entropy was first proposed. Its purpose is to analyze the current evolutionary trend and balance the convergence and diversity of AMOPSO. Zhang et al. [8] proposed a particle update strategy. A mutation operator is used to balance the searchability, adjust the parameters, and update the global leader. Leng et al. [9] proposed a MOPSO based on the optimal grid distance (GDMOPO), where grid strategies...
are introduced to the maintenance strategy of external archives and the selection of global learning samples. The algorithm can effectively balance the convergence and distribution of the algorithm. However, it lacks a strategy to prevent the algorithm from falling into a local optimum. Zhang et al. [10] proposed a MOPSO based on a competition mechanism (CMOPSO), in which elite particles are proposed to guide the population evolution. The algorithm provides a reference for scholars to improve MOPSO. However, its performance is not good when optimizing large-scale problems. Han et al. [11] proposed MOPSO. However, its performance is not good when contrast to single-objective optimization problems, in


decision variables and constraints.

\min \ y = F(x) = (f_1(x), f_2(x), \ldots, f_m(x))

\begin{align}
\quad & \quad \text{s.t.} \\
& g_i(x) \geq 0 \quad i = 1, 2, \ldots, p \\
& h_j(x) = 0 \quad j = 1, 2, \ldots, q,
\end{align}

where \( x = (x_1, x_2, \ldots, x_n) \in X \subset R^n \) is an n-dimensional vector in the decision space, \( y = (y_1, y_2, \ldots, y_m) \in Y \subset R^m \) is the number of objective functions, \( g_i(x) \geq 0 (i = 1, 2, \ldots, p) \) is the i-th constraint of p inequality constraints, and \( h_j(x) = 0 (j = 1, 2, \ldots, q) \) is the j-th constraint of q equality constraints.

**Definition 1.** Pareto dominance: \( x_\nu, x_\nu' \in \overline{X} \) is a feasible solution to this multiobjective problem, where \( x_\nu \) is dominant compared to \( x_\nu', (x_\nu, x_\nu') \) only if

\[ \forall i = 1, 2, \ldots, m, f_i(x_\nu) \leq f_i(x_\nu') \exists j = 1, 2, \ldots, m, f_j(x_\nu') < f_j(x_\nu). \]  

**Definition 2.** Pareto-optimal: \( x^* \in X \) is the Pareto-optimal solution (nondominated solution or noninferior solution) on \( X \), only if the following condition is satisfied:

\[ \exists x \in \overline{X}: x \prec x^*. \]  

**Definition 3.** Pareto-optimal set: for the multiobjective optimization problem, the optional solution set can be defined as follows:

\[ P^* = \{ x \in \overline{X} \mid \exists x^*, f_i(x^*) \leq f_i(x), \quad (i = 1, 2, \ldots, m) \}. \]  

**Definition 4.** Pareto-optimal front: the curved surface consists of the objective function values corresponding to all pareto-optimal solutions in the pareto-optimal solution set \( P^* \) is called the pareto front

\[ PF = \{ F(x^*) = (f_1(x^*), f_2(x^*), \ldots, f_m(x^*)) \mid x^* \in P^* \}. \]  

2. Research Background

2.1. Basic Definitions of MOPs. A minimum MOP with \( n \) decision variables and \( m \) objectives is described as follows:

\[
\begin{aligned}
\min & \quad y = F(x) = (f_1(x), f_2(x), \ldots, f_m(x)) \\
\text{s.t.} & \quad g_i(x) \geq 0, \quad i = 1, 2, \ldots, p \\
& \quad h_j(x) = 0, \quad j = 1, 2, \ldots, q, 
\end{aligned}
\]

where \( x = (x_1, x_2, \ldots, x_n) \in X \subset R^n \) is an n-dimensional vector in the decision space, \( y = (y_1, y_2, \ldots, y_m) \in Y \subset R^m \) is the number of objective functions, \( g_i(x) \geq 0 (i = 1, 2, \ldots, p) \) is the i-th constraint of p inequality constraints, and \( h_j(x) = 0 (j = 1, 2, \ldots, q) \) is the j-th constraint of q equality constraints.
\[ v_i(t + 1) = w_i v_i(t) + c_1 r_1 (p_{best}(t) - x_i(t)) + c_2 r_2 (g_{best}(t) - x_i(t)), \]
\[ x_i(t + 1) = x_i(t) + v_i(t + 1), \]

where \( t \) is the iteration number, and \( w \) is the inertial weight that ensures that the particle can expand the search space. \( r_1 \) and \( r_2 \) are random numbers generated uniformly in the range \([0,1]\), \( c_1 \) and \( c_2 \) are the learning factors. \( p_{best} \), is the individual optimal for the \( i \)-th particle, and \( g_{best} \) is the global optimum for the swarm.

2.3. Related Work. To improve the performance of MOPSO, many scholars have continuously explored many excellent strategies [12]. However, most of the current MOPSOs have difficulties selecting the optimal solution in the external archive, balancing the local and global search, etc. Deng et al. [13] proposed three strategies: a differential mutation, a feasible solution space transformation strategy, and a new multi-group mutation mechanism. The proposed strategies improve diversity and convergence and accelerate the convergence speed. Zhang et al. [14] proposed a decomposition-based multiobjective archiving method (DAA) to speed up the convergence of the algorithm, where the entire objective space is evenly divided into subspaces according to a set of weights, and a distance-based standardization method combined with Pareto dominance is proposed to balance the algorithm search. Deng et al. [15] designed a hybrid mutation strategy based on local domain advantages and SaNSDE. The mutation strategy is used in the early stage to ensure search efficiency. The SaNSDE is used in the later stage to prevent search stagnation. In addition, a differential evolution algorithm and quantum evolution algorithm are used to enhance the diversity of the population. They use quantum rotation to speed up the convergence rate, which effectively balances the local search ability and global search ability of the algorithm.

For simple operation and fast convergence of MOPSO, many improvements for MOPSOs have emerged [4]. All improvements can be summarized into the following categories: (1) MOPSO based on Pareto dominance, which guides the population evolution by selecting individual optimal and global optimal as learning exemplars, such as CMOPSO [10]. (2) MOPSO based on decomposition, such as MPSO/D [16], dMOPSO [17], and MMOPSO [18]. (3) MOPSO based on indicators such as R2HMOPSO [19]. (4) MOPSO based on sortings, such as GMRMOPSO [2] and GDMOPSO [9]. In 2004, Coello et al. proposed MOPSO [6], which provided a new method for PSO to solve MOPs. Deb et al. combined the crowded distance method in NSGAII [20] and proposed a crowded distance to select global learning exemplars. Nebro et al. proposed SMPSO [21], in which the speed limit was added to optimize the population. Zhang and Li used the aggregation method in MOEAD [3] to decompose MOPs into single-objective optimization problems and proposed MOPSO/D [22]. Martinez and Coello decomposed MOPs into multiple subproblems in dMOPSO [17] and optimized each subproblem separately. In [18], Chen et al. used the decomposition idea to transform MOPs into a single-objective optimization problem and proposed a new speed update mechanism. In addition to the MOPSO mentioned above based on crowding distance and decomposition, many representative MOPSOs have appeared in recent years, such as GMRMOPSO [2], MOPSO-ASFS [23], AMPSO/SEE [24], MOPSONN [25], and others. In particular, in MOPSONN, the author proposed an archive updating mechanism based on the nearest neighbor method using the competition mechanism. The archive is updated according to the nearby distance. In the late iteration, the maximum cost rule and the sum of costs rule are used to update the archives. The current MOPSO improvement has some effect in dealing with MOPs. However, these methods have shown some improvement space when optimizing certain problems. Motivated by the above-mentioned external archive and global selection strategy of the learning sample, we propose a novel MOPSO (IDMOPSO) based on the adaptive grid and ideal distance.

3. MOPSO Based on Ideal Distance

3.1. Grid Construction. The diversity of the population is an essential criterion for evaluating the performance of an algorithm [6, 11]. In our proposed IDMOPSO, we adopted a strategy to map the objective function space to the grid. The number of particles in the grid was regarded as the density of the grid [9]. The smaller the number of particles in the grid, the smaller the grid density, and vice versa. When the grid density in several grids is the same, these grids are considered to have the same diversity. Here, the density is zero if the grid is on a particle. Several definitions related to grid construction are provided as follows.

**Definition 5.** The upper and lower boundary of the grid coordinates: in the objective space, the maximum and minimum values of the \( m \)-th objective are \( f_m(x) \) and \( \min f_m(x) \), and the upper \( (U_m) \) and lower boundary \( (L_m) \) of the \( m \)-th object are defined as follows:

\[ L_m = \min f_m(x) - \min f_m(x), \]
\[ U_m = \max f_m(x) + \frac{\max f_m(x) - \min f_m(x)}{c}, \]

where \( c \) is the number of grid divisions that can be selected according to different optimization problems.

**Definition 6.** Grid coordinates: if the function value of particle \( x \) on the \( m \)-dimensional is \( f_m(x) \), the corresponding grid coordinates are

\[ G_m(x) = \left[ \frac{f_m(x) - L_m}{d_m} \right], \]

where \( d_m = U_m - L_m/c \) represents the grid width. Figure 1 shows a specific example mapping an objective space to grid
3.2. Ideal Distance and $\sigma$ Dominance. To evaluate the quality of the particles effectively, this study uses the position information of particles in the grid to find the virtual positive ideal point, ideal negative point, and ultimate ideal point in the grid. This paper considers $Z^* = (f_1^{\text{min}}, f_2^{\text{min}}, \ldots, f_M^{\text{min}})$ and $Z^{-*} = (f_1^{\text{max}}, f_2^{\text{max}}, \ldots, f_M^{\text{max}})$ as the ideal positive and ideal negative points. The ultimate ideal point is described by $Z^0 = (f_1^0, f_2^0, \ldots, f_M^0)$. $f_j^\text{min}$ representing the minimum value of the $j$-th target in the current evolution process and $f_j^\text{max}$ represents the maximum value of the $j$-th target in the current evolution process. $f_j^0$ represents is 0 of the $j$-th objective value in the current evolution process.

Let the Euclidean distance of the particle $i$ to $Z^*$ be its positive ideal distance,

$$d(i, Z^*) = \sqrt{\sum_{j=1}^{M} (f_j(i) - f_j^{\text{min}})^2}. \quad (10)$$

Let the Euclidean distance of the particle $i$ to $Z^{-*}$ be its negative ideal distance,

$$d(i, Z^{-*}) = \sqrt{\sum_{j=1}^{M} (f_j(i) - f_j^{\text{max}})^2}. \quad (11)$$

Let the Euclidean distance of particle $i$ to $Z^0$ be its ultimate ideal distance.

$$d(i, Z^0) = \sqrt{\sum_{j=1}^{M} (f_j(i) - f_j^0)^2}. \quad (12)$$

In this study, the ideal point is a virtual reference point that does not exist in actual problems. Thus, the ideal positive point has relatively better properties, whereas the ideal negative point has the opposite. Therefore, if a solution is close to the positive ideal and ultimate ideal point, this solution is optimal. Equation (13) gives $\sigma$ dominance related to positive, negative, and ultimate ideal distances.

$$\sigma(i) = d_1(i) + d_2(i) - d_3(i). \quad (13)$$

Where $d_1(i) = d(i, Z^*)$, $d_2(i) = d(i, Z^{-*})$, $d_3(i) = d(i, Z^0)$. For any particle $P_1$ and $P_2$ in the external archive, if $P_1$ and $\sigma$ dominate $P_2$, it must satisfy $\sigma(P_1) < \sigma(P_2)$ in the rule of $\sigma$ dominance. For example, Figure 2 shows the particle $P_1 = (2, 5)$, $P_2 = (2, 8, 4)$, $P_3 = (3, 3, 3, 9)$, $P_4 = (4, 5, 2, 5)$ in the external archive, and the ideal and negative ideal points are $Z^* = (2, 2, 5)$ and $Z^{-*} = (4, 5, 5)$, respectively. By calculation, $d_1(P_1) = 2.50$, $d_2(P_1) = 2.50$, $d_3(P_1) = 5.38$, $d_1(P_2) = 1.70$, $d_2(P_2) = 1.97$, $d_3(P_2) = 4.88$, $d_1(P_3) = 1.91$, $d_2(P_3) = 1.62$, $d_3(P_3) = 5.10$, $d_1(P_4) = 2.50$, $d_2(P_4) = 2.50$, $d_3(P_4) = 5.15$ (all values are kept in two decimal places), and then $\sigma(P_1) = 5.38$, $\sigma(P_2) = 4.61$, $\sigma(P_3) = 5.39$, $\sigma(P_4) = 5.15$. So $P_2$, $\sigma$ dominate $P_1$, $P_3$, and $P_4$. When selecting noninferior solutions, because of the large number of noninferior solutions, we cannot compare them to each other, so ideal points are usually introduced as a reference. Here, the ultimate and positive ideal points are closest to the lower left of the coordinate axis than all noninferior solutions. Therefore, the solution closest to the positive and ultimate ideal point is the optimal solution, guiding other particles to fly to the optimal position.

3.3. New Speed Update Mechanism and Mutation Operation. Equations (6) and (7) are generally used to update the speed and position of the particles for most MOPSOs. However, because of specific differences among the particles, if the same method is used to update all particles, it will go against the advantages of some individuals for swarm evolution. Therefore, we introduce the fine-tuning parameters to adjust the speed of the particles, where the parameter $\mu$ determines the inheritance of the previous speed of the particles to coordinate the global and local search of the algorithm. In addition, the cosine factor is introduced to mutate the position of the particles to improve the ability to jump out of the local optimal solutions. The specific speed and position update formulas are as follows:
3.4. Maintenance of External Archives based on the $\sigma$ Dominance Principle. MOPSO usually adopts external archives to store noninferior solutions generated by each iteration in the algorithm running process. Then, it uses a specific mechanism to control their size. To provide the dominant candidate solution for global learning exemplars, the maintenance strategy combining the adaptive grid and $\sigma$ dominance principle is adopted to control the size of external archives. When the size of the external archive exceeds the preset maximum value, some particles are deleted. First, the adaptive grid is used to evaluate the distribution of noninferior solutions in the external archive and then find the areas where the particles are densely distributed in the external archive. Next, the dominance of the noninferior solutions is evaluated using the $\sigma$ dominance principle. The particles with poor dominance are deleted. Figure 3 gives one example about the $\sigma$ dominance principle. Because particles 3, 4, 5, and 6 have the highest grid density, and particle 6 is dominated by particles 3, 4, and 5 under the $\sigma$ dominance principle, particle 6 is deleted.

To further test the effectiveness of the $\sigma$ dominance principle, Figure 4 shows the different results when applying the $\sigma$ dominance principle and the random selection strategy to maintain the size of the external archive, where the algorithm performs 200 iterations under 30 runs running independently on ZDT1 and ZDT2. As shown in Figure 4, the $\sigma$ dominance principle has a smaller GD [29] value than the random selection strategy, indicating that the adaptive grid and $\sigma$ dominance principle can effectively maintain external archives compared with the random selection strategy.

3.5. Selection Strategy of Global Learning Exemplar Based on the $\sigma$ Dominance Principle. The selection of the global learning exemplar is the key factor because the global learning exemplar will lead the whole swarm fly to move to the true Pareto frontier. Therefore, if some particles with poor convergence are selected, the convergence speed of the algorithm may be slower and affect the convergence accuracy. If some particles with poor distribution are selected, the algorithm can easily fall into the local optimum. Here, the combination strategy of the adaptive grid and the $\sigma$ dominance principle is adopted to select global learning examples. The selection steps were as follows:

Step 1. Select the grid with the smallest grid density. Suppose there are many grids with the same density. In that case, the particles in the same grid are randomly selected as candidate solutions for global learning exemplars.

Step 2. Among the candidate solutions selected in Step 1, the candidate solution with the smallest $\sigma$ value is selected as the global learning exemplar.

In Figure 3, comparing the grid density, we can see that particles 1 and 2 have the smallest, and the distribution of particles is more uniform. Therefore, the particles in the grid were used as candidate solutions for the global learning sample. Because particle 1 is better than particle 2 under the $\sigma$ dominance principle, particle 1 is used as the global learning exemplar to guide the swarm to fly. To test the effectiveness of the $\sigma$ dominance principle in the selection strategy.
Table 1: The influence of $\alpha$ value on the IGD of the algorithm.

| Problems | IGD | $\alpha = 1$ | $\alpha = 2$ | $\alpha = 3$ | $\alpha = 4$ | $\alpha = 5$ |
|----------|-----|-------------|-------------|-------------|-------------|-------------|
| ZDT1     | Mean| 9.9152e−03  | 6.0037e−03  | 5.2661e−03  | 5.4532e−03  | 5.6963e−03  |
| ZDT2     | Mean| 1.0883e−02  | 5.8845e−03  | 5.5884e−03  | 5.9663e−03  | 5.4151e−03  |
| ZDT3     | Mean| 2.0896e−01  | 2.0751e−01  | 2.0127e−01  | 2.0483e−01  | 2.0666e−01  |
| ZDT4     | Mean| 1.0555e−02  | 5.2735e−03  | 5.5132e−03  | 5.2203e−03  | 5.3524e−03  |
| ZDT6     | Mean| 7.6585e−02  | 2.4912e−03  | 2.4514e−03  | 2.2851e−03  | 2.3637e−03  |

Table 2: The influence of $\alpha$ value on the HV of the algorithm.

| Problems | HV | $\alpha = 1$ | $\alpha = 2$ | $\alpha = 3$ | $\alpha = 4$ | $\alpha = 5$ |
|----------|----|-------------|-------------|-------------|-------------|-------------|
| ZDT1     | Mean| 7.1242e−01  | 7.1861e−01  | 7.1867e−01  | 7.1852e−01  | 7.1854e−01  |
| ZDT2     | Mean| 4.3619e−01  | 4.4138e−01  | 4.4188e−01  | 4.3980e−01  | 4.4213e−01  |
| ZDT3     | Mean| 6.5189e−01  | 6.5402e−01  | 6.5568e−01  | 6.5285e−01  | 6.5375e−01  |
| ZDT4     | Mean| 7.1145e−01  | 7.1810e−01  | 7.1812e−01  | 7.1806e−01  | 7.1787e−01  |
| ZDT6     | Mean| 3.5907e−01  | 3.8262e−01  | 3.8329e−01  | 3.8302e−01  | 3.8206e−01  |

Figure 3: Maintenance of external archive and selection of global learning sample.

Figure 4: Comparison of $\sigma$ dominance principle and random strategy in maintaining an external archive.
strategy of the global learning exemplar, Figure 5 shows the different results under the \( \sigma \) dominance principle and random selection strategy to select global learning exemplars where the algorithm performs 200 iterations under 30 runs running independently on ZDT1 and ZDT2. It can be seen from Figure 5 that the \( \sigma \) dominance principle makes the learning exemplars obtain a smaller IGD value. The proposed strategy first uses an adaptive grid to analyze the density of particles, and areas with sparse particles are confirmed where the dominant particles are selected as learning samples. This strategy maintains the diversity of the swarm. It effectively improves the probability of the swarm moving to the true Pareto frontier.

3.6. IDMOPSO Algorithm Flow. A flowchart of the IDMOPSO is shown in Figure 6. The specific process is as follows:

- **Step 1.** Set the relevant parameters of IDMOPSO and initialize the population and velocity
- **Step 2.** Calculate the fitness value of the particles and perform Pareto nondominated sorting
- **Step 3.** For the swarm, determine the individual learning exemplars (\( g_{best} \)) for each particle.
- **Step 4.** Establish an external archive. Then, use the adaptive grid and \( \sigma \) dominance principle to control its size and update the particles in the archive.
- **Step 5.** Use the adaptive grid and \( \sigma \) dominance principle to select a particle in the external archive as the global learning exemplars (\( g_{best} \)) to guide the swarm evolution.
- **Step 6.** Use equations (14) and (15) to update the position and velocity of each particle, respectively.
- **Step 7.** Determine whether the termination condition is met; if satisfied, the iteration ends; otherwise, return to Step 2.

4. Experimental Results and Analysis

4.1. Experimental Conditions and Parameter Settings. To effectively evaluate the performance of IDMOPSO, we compared IDMOPSO with specific popular and novel algorithms, including four MOPSOs (MPSO/D [16], dMOPSO [17], SMPSO [21], and NMPSO [30]) and four MOEAs (MOEA/D [3], MOEADD [31], MOMBII [32], IMMOEA) [33]. The swarm size of all algorithms was set to 200, and Table 3 lists the main parameter settings for all algorithms. In Table 3, \( \text{div} \) is the division network number of cells; \( c_1 \) and \( c_2 \) are the parameters of the velocity equation used in the MOPSO; \( p_c \) and \( p_m \) are the crossover probability and mutation probability, respectively; \( \eta_c \) and \( \eta_m \) are the distribution indices of SBX and PM, respectively; \( F \) and \( CR \) are parameters set by the authors in differential evolution; \( T \) is the number of divisions in the GA; \( K \) is the number of reference vectors; \( A \) is the threshold of variances; \( E \) is the tolerance threshold; \( R \) is the record size of nadir vectors. The source code of the comparison algorithm is obtained from platEMO [34]. 22 benchmark functions were selected as the test function set, including ZDT1–4, ZDT6, DTLZ1–7 [28], and UF1-10 [35]. Table 4 lists the relevant parameters of each test function, where \( D \) represents the dimensionality of the decision variables, \( M \) represents the number of objectives, and FE represents the maximum number of evaluations.

4.2. Evaluation Index. To test the performance of all algorithms, the HV [26] and IGD [27] indices were used to evaluate the performance of all algorithms. The IGD measures the average value of the minimum distance between the real Pareto frontier and the noninferior solution obtained by the algorithm. The smaller the IGD value, the better is the comprehensive performance of the algorithm. Furthermore, HV measures the ratio of the noninferior solution obtained by the algorithm covering the space enclosed by the real Pareto front and the reference point (RP). The larger the HV value, the better the comprehensive performance of the algorithm.

4.2.1. Comparison of IDMOPSO with Four MOPSOs. Tables 5 and 6 show the average and standard deviation values of IGD and HV obtained by running 30 times independently. The best mean values obtained by all algorithms on the test function are indicated in bold. It can be concluded that the comprehensive performance of IDMOPSO is better than that of MPSO/D, SMPSO, NMPSO, and dMOPSO, where the number of the best IGD (14) and HV (14) of IDMOPSO are higher than those of the other algorithms. MPSO/D had no best IGD and HV values, SMPSO obtained three best IGD and four best HV values; NMPSO obtained three best IGD and four best HV values, and dMOPSO obtained two best IGD values.

On UF1-10, IDMOPSO had the best performance, where it obtained the nine best IGD and ten best HV values. In the ZDT series, IDMOPSO obtained the best IGD values for ZDT1, ZDT2, and ZDT4, and the best HV values were obtained for ZDT1, ZDT2, ZDT3, and ZDT4. Even if the performance on ZDT6 is not the best, it is better than that of MPSO/D and dMOPSO. In the DTLZ, IDMOPSO obtained the best IGD on DTLZ1 and DTLZ6 but did not perform best on other DTLZ test functions. From the above experimental analysis, it can be found that IDMOPSO achieved good results in the optimization of ZDT and UF.

When optimizing the irregular three-dimensional DTLZ2–4, IDMOPSO has an average performance, which indicates that IDMOPSO needs further improvement in optimizing the irregular MOPs. In other words, in 22 test functions, IDMOPSO achieved the number of better IGD and HV than MPSO/D, SMPSO, NMPSO, and dMOPSO in most test functions means that the overall performance of IDMOPSO is better than that of other MOPSOs.

For further observation, Figs 7–10 show the spatial distribution of the noninferior solution sets obtained by MPSO/D, SMPSO, dMOPSO, and NMPSO, respectively. Figure 7 shows that the distribution and convergence of IDMOPSO and SMPSO are better than those of MPSO/D, dMOPSO, and NMPSO on ZDT1. Figure 8
shows that IDMOPSO have better convergence and distribution, and the other three algorithms still need to be improved in terms of convergence and distribution on ZDT2. Additionally, Figures 9 and 10 show that the distribution and convergence of IDMOPSO are significantly better than those of the other four MOPSOs on UF9 and UF10. From the above analysis, it can be concluded that the convergence and diversity of IDMOPSO are better than those of the other four MOPSOs, which fully demonstrates the effective use of an adaptive grid and ideal distance to maintain external archive.

4.2.2. Comparison of IDMOPSO with Four MOEAs. Tables 7 and 8 show the mean and standard deviation of the IGD and HV obtained by independently running 30 times. The best mean values obtained by all algorithms are indicated in bold. Tables 7 and 8 show that the comprehensive
Table 3: Related parameters of nine algorithms.

| Algorithms | Parameters setting |
|------------|--------------------|
| MPSO/D [16] | $w \in [0, 1, 0.5], c_1, c_2 \in [1, 2]$, $p_1 = 0.9, F = 0.5, CR = 0.5$, $p_m = 1/n, \eta_m = 20$ |
| dMPSO [17] | $w \in [0, 1, 0.5], c_1, c_2 \in [1, 2]$, $p_1 = 1/n, \eta_m = 20$ |
| SMPSO [21] | $w \in [0, 1, 0.5], c_1, c_2 \in [1, 2]$, $p_1 = 1/n, \eta_m = 20$ |
| NMPSO [30] | $w \in [0, 1, 0.5], c_1, c_2 \in [1, 2]$, $p_1 = 1/n, \eta_m = 20$, $T = 20$ |
| MOEA/D [3] | $a = 0.5, e = 0.001, r = 5$ |
| MOEADD [31] | $p_1 = 1, p_m = 1/n, \eta_m = 20$, $T = 20$ |
| MOMBI-II [32] | $c_1 = c_2 = 2, \text{div} = 50$ |
| IMMOEA [33] | |
| IDMOPSO | |

Table 4: The relevant parameter settings of each test function.

| Problems   | $D$ | $M$ | FEs |
|------------|-----|-----|-----|
| ZDT1-ZDT3  | 30  | 2   | 10000 |
| ZDT4 and ZDT6 | 10  | 2   | 10000 |
| UF1-UF7    | 30  | 2   | 10000 |
| UF8-UF10   | 30  | 3   | 10000 |
| DTLZ1      | 7   | 3   | 10000 |
| DTLZ2-DTLZ6 | 12  | 3   | 10000 |
| DTLZ7      | 22  | 3   | 10000 |

Table 5: IGD value of IDMOPSO and four MOPSOs on 22 test functions.

| Problem      | dMPSO | SMPSO | NMPSO | IDMOPSO |
|--------------|-------|-------|-------|---------|
| ZDT1         | 5.0918e-2 | 9.9807e-2 | 4.69e-2 | 2.7099e-2 | 5.2661e-3 | 7.95e-4 |
| ZDT2         | 6.1058e-2 | 1.6617e-1 | 1.14e-1 | 1.87e-1 | 4.3591e-2 | 8.82e-2 | 5.5884e-3 | 1.19e-3 |
| ZDT3         | 3.6097e-2 | 2.92e-3 | 5.41e-2 | 1.04e-1 | 8.7305e-2 | 2.76e-2 | 2.0127e-1 | 5.33e-3 |
| ZDT4         | 5.5545e-1 | 6.05e+0 | 6.88e+0 | 3.63e+0 | 9.7871e+1 | 1.20e+1 | 5.1332e+3 | 6.78e-4 |
| ZDT5         | 4.9331e-3 | 3.68e-3 | 1.22e-3 | 1.92e-3 | 2.9325e-3 | 2.79e-4 | 2.4514e-3 | 5.03e-4 |
| UF1          | 6.5387e-1 | 2.6322e-1 | 4.34e-2 | 3.68e-1 | 1.37e-2 | 5.37e-2 | 8.86e-3 | 6.68e-3 |
| UF2          | 9.1537e-2 | 1.1246e-1 | 8.56e-3 | 1.08e-1 | 3.35e-2 | 6.26e-3 | 2.97e-3 | 4.98e-3 |
| UF3          | 3.3142e-1 | 5.39e-3 | 1.61e-2 | 4.50e-1 | 3.76e-1 | 6.65e-2 | 1.62e-2 | 2.71e-2 |
| UF4          | 1.3784e-1 | 6.13e-3 | 3.84e-3 | 1.13e-1 | 6.25e-2 | 7.14e-3 | 5.60e-4 | 5.08e-4 |
| UF5          | 3.2476e+0 | 2.23e-1 | 2.85e-1 | 2.90e-1 | 1.85e+0 | 5.46e-1 | 6.27e-1 | 1.74e-1 |
| UF6          | 2.3970e+0 | 4.89e-1 | 2.65e-1 | 1.27e+0 | 6.98e-1 | 1.65e-1 | 3.90e-2 | 5.05e-2 |
| UF7          | 3.6894e-1 | 9.06e-2 | 1.68e-2 | 3.68e-1 | 2.45e-1 | 1.90e-1 | 4.99e-2 | 6.84e-3 |
| UF8          | 3.3731e-1 | 3.45e-2 | 4.22e-2 | 3.87e-1 | 4.58e-1 | 1.31e-1 | 4.27e-1 | 7.98e-2 |
| UF9          | 5.8466e-1 | 4.08e-2 | 1.64e-2 | 5.62e-1 | 4.72e-1 | 5.32e-1 | 1.33e-1 | 1.59e-2 |
| UF10         | 9.4445e-1 | 5.56e-3 | 4.24e-1 | 2.78e+0 | 1.58e+0 | 3.17e-1 | 3.49e+0 | 9.14e-2 |
| DTLZ1        | 1.1921e+1 | 6.22e+0 | 1.80e+0 | 4.20e+0 | 6.49e+0 | 3.73e+0 | 1.46e+0 | 7.29e+0 |
| DTLZ2        | 1.0408e+1 | 5.15e+0 | 7.65e+0 | 6.17e+0 | 1.87e+1 | 2.29e+2 |
| DTLZ3        | 9.8334e+1 | 6.53e+1 | 1.24e+1 | 5.47e+1 | 1.14e+2 | 2.82e+1 | 1.73e+2 | 9.38e+0 |
| DTLZ4        | 3.3180e+1 | 1.88e-2 | 1.45e-1 | 3.94e-1 | 7.33e-2 | 1.77e-2 |
| DTLZ5        | 2.4644e-2 | 2.83e-2 | 4.48e-2 | 3.55e-3 | 6.83e-4 | 6.19e-3 |
| DTLZ6        | 2.0473e-2 | 2.65e-4 | 1.13e+0 | 1.31e+0 | 6.68e-3 | 1.93e-3 |
| DTLZ7        | 1.1578e-1 | 1.01e-2 | 5.67e-1 | 7.6518e-1 | 4.6851e-2 | 6.15e-1 | 1.06e-2 |

For Pareto frontiers, Figures 7–10 show the spatial distribution of the noninferior solution sets obtained by the four MOEAs and IDMOPSO. Figure 7 shows that the distribution and convergence of IDMOPSO are better than that of MOEA/D, MOEADD, IMMOEA, and MOMBI-II. In 22 test functions, IDMOPSO obtained fourteen best IGD and fifteen best HV values, MOEA/D obtained one best IGD and HV values, MOEADD obtained the two best IGD and zero best HV values and, IMMOEA obtained one best IGD and zero best HV values. MOMBI-II obtained the four best IGD and five best HV values. On UF1-10, IDMOPSO obtained the nine best IGD and HV values, and on the ZDT series, IDMOPSO obtained the best IGD on ZDT1, ZDT2, and ZDT4, and the best HV on ZDT1–4 and ZDT6. As can be seen from the above analysis, IDMOPSO is significantly better than MOEA/D, MOEADD, MOMBI-II, and IMMOEA on ZDT, UF, and DTLZ series test functions.
Table 6: HV value of IDMOPSO and four MOPSOs on 22 test functions.

| Problem | dMOPSO | MPSO/D | SMPSO | NMPSO | IDMOPSO |
|---------|--------|--------|-------|-------|---------|
| ZDT1    | 6.596e−1 (1.92e−2) | 5.7862e−1 (6.0e−2) | 5.6116e−1 (1.14e−1) | 6.9255e−1 (1.23e−2) | 7.1867e−1 (1.55e−3) |
| ZDT2    | 3.7124e−1 (5.83e−2) | 2.5873e−1 (1.01e−1) | 3.2765e−1 (1.40e−1) | 4.1357e−1 (7.11e−2) | 4.4188e−1 (1.95e−3) |
| ZDT3    | 6.0079e−1 (1.64e−2) | 4.4572e−1 (5.33e−2) | 5.4371e−1 (8.07e−2) | 5.7214e−1 (9.98e−3) | 6.5568e−1 (3.522e−4) |
| ZDT4    | 5.9017e−2 (9.06e−2) | 0.00e+0 + 0 (0.0e+0) | 0.00e+0 + 0 (0.0e+0) | 0.00e+0 + 0 (0.0e+0) | 7.1812e−1 (9.95e−4) |
| ZDT6    | 3.8696e−1 (3.85e−3) | 3.8736e−1 (1.15e−3) | 3.9002e−1 (8.08e−5) | 3.8977e−1 (2.46e−4) | 3.8329e−1 (4.99e−3) |
| UF1     | 5.9678e−2 (4.70e−2) | 3.6761e−1 (4.81e−2) | 2.7276e−1 (5.93e−2) | 5.2244e−1 (2.99e−2) | 6.5244e−1 (1.29e−2) |
| UF2     | 6.1895e−1 (5.91e−3) | 5.7948e−1 (9.59e−3) | 6.0805e−1 (7.53e−3) | 6.1960e−1 (6.21e−3) | 6.7699e−1 (9.26e−3) |
| UF3     | 3.0681e−1 (6.70e−3) | 1.7175e−1 (1.37e−2) | 2.1139e−1 (4.66e−2) | 2.6576e−1 (5.71e−2) | 5.1894e−1 (3.98e−2) |
| UF4     | 2.5270e−1 (6.59e−3) | 3.0822e−1 (4.61e−3) | 2.9231e−1 (1.04e−2) | 3.6091e−1 (9.86e−3) | 3.7028e−1 (4.72e−3) |
| UF5     | 0.0000e+0 + 0 (0.0e+0) | 0.0000e+0 + 0 (0.0e+0) | 0.0000e+0 + 0 (0.0e+0) | 0.0000e+0 + 0 (0.0e+0) | 1.8268e−2 (3.53e−2) |
| UF6     | 0.0000e+0 + 0 (0.0e+0) | 0.0000e+0 + 0 (0.0e+0) | 0.0000e+0 + 0 (0.0e+0) | 0.0000e+0 + 0 (0.0e+0) | 1.1397e−2 (1.74e−2) |
| UF7     | 1.8356e−1 (7.12e−2) | 2.7689e−1 (7.01e−2) | 1.9024e−1 (7.32e−2) | 3.4454e−1 (1.22e−1) | 5.1251e−1 (9.89e−3) |
| UF8     | 2.7161e−1 (1.72e−2) | 6.2497e−2 (2.39e−2) | 1.6792e−1 (3.78e−2) | 2.7576e−1 (5.79e−2) | 3.3345e−1 (5.59e−3) |
| UF9     | 2.2201e−1 (1.68e−2) | 1.0374e−1 (3.02e−2) | 2.0497e−1 (4.35e−2) | 2.9983e−1 (5.09e−2) | 6.3271e−1 (2.01e−2) |
| UF10    | 9.0882e−2 (8.05e−5) | 0.0000e+0 + 0 (0.0e+0) | 0.0000e+0 + 0 (0.0e+0) | 0.0000e+0 + 0 (0.0e+0) | 2.0274e−2 (9.15e−2) |
| DTLZ1   | 1.0427e−2 (5.71e−2) | 0.0000e+0 + 0 (0.0e+0) | 1.0487e−1 (1.94e−1) | 0.0000e+0 + 0 (0.0e+0) | 0.0000e+0 + 0 (0.0e+0) |
| DTLZ2   | 4.3721e−1 (8.86e−3) | 5.4689e−1 (2.55e−3) | 5.1098e−1 (9.30e−3) | 5.6944e−1 (9.10e−4) | 3.5096e−1 (1.97e−2) |
| DTLZ3   | 0.0000e+0 + 0 (0.0e+0) | 0.0000e+0 + 0 (0.0e+0) | 1.7041e−2 (4.55e−2) | 0.0000e+0 + 0 (0.0e+0) | 0.0000e+0 + 0 (0.0e+0) |
| DTLZ4   | 2.7558e−1 (3.23e−2) | 4.1325e−1 (4.24e−2) | 3.2850e−1 (9.37e−2) | 5.6139e−1 (4.05e−2) | 4.3257e−1 (1.74e−2) |
| DTLZ5   | 1.7380e−1 (6.20e−2) | 1.4482e−1 (1.06e−2) | 1.9988e−1 (3.54e−4) | 1.9819e−1 (5.32e−4) | 1.7304e−1 (1.06e−2) |
| DTLZ6   | 1.8996e−1 (5.95e−4) | 6.2695e−3 (1.98e−2) | 2.5917e−2 (6.66e−2) | 1.9813e−1 (5.63e−4) | 1.9385e−1 (5.25e−3) |
| DTLZ7   | 2.4415e−1 (4.69e−3) | 6.0860e−2 (2.78e−2) | 1.0836e−1 (7.69e−2) | 2.8129e−1 (9.39e−4) | 2.6198e−1 (4.88e−3) |

Best/All 0/22 0/22 4/22 4/22 14/22

Figure 7: Continued.
Figure 7: Approximate Pareto frontiers of IDMOPSO, four MOPSOs, and four MOEAs on the test function ZDT1. (a) MPSO/D; (b) SMPSO; (c) dMOPSO; (d) NMPSO; (e) MOEA/D; (f) MOEADD; (g) MOMBI-II; (h) IMMOEA; and (i) IDMOPSO.

Figure 8: Continued.
Figure 8: Approximate Pareto frontiers of IDMOPSO and four MOPSOs and four MOEAs on the test function ZDT2. (a) MPSO/D; (b) SMPSO; (c) dMOPSO; (d) NMPSO; (e) MOEA/D; (f) MOEADD; (g) MOMBI-II; (h) IMMOEA; and (i) IDMOPSO.

Figure 9: Approximate Pareto frontiers of IDMOPSO and four MOPSOs and four MOEAs on the test function UF9. (a) MPSO/D; (b) SMPSO; (c) dMOPSO; (d) NMPSO; (e) MOEA/D; (f) MOEADD; (g) IMMOEA; (h) MOMBI-II; and, (i) IDMOPSO.
SMPSO; (c) dMOPSO; (d) NMPSO; (e) MOEA/D; (f) MOEADD; (g) IMMOEA; (h) MOMBI-II; and (i) IDMOPSO.

Figure 10: Approximate Pareto frontiers of IDMOPSO, four MOPSOs, and four MOEAs on the test function UF10. (a) MPSO/D; (b) SMPSO; (c) dMOPSO; (d) NMPSO; (e) MOEA/D; (f) MOEADD; (g) IMMOEA; (h) MOMBI-II; and (i) IDMOPSO.

Table 7: IGD value of IDMOPSO and four MOEAs on 22 test functions.

| Problem | MOEA/D | MOEADD | IMMOEA | MOMBIII | IDMOPSO |
|---------|--------|--------|--------|---------|---------|
| ZDT1    | 1.9086e−1 (8.50e−2) | 1.1875e−1 (1.67e−2) | 1.8570e−1 (2.14e−2) | 3.9878e−2 (7.83e−3) | 5.2661e−3 (7.95e−4) |
| ZDT2    | 5.6377e−1 (6.08e−2) | 1.5870e−1 (2.14e−2) | 1.8925e−1 (3.40e−2) | 9.9762e−2 (8.98e−2) | 5.5884e−3 (1.19e−3) |
| ZDT3    | 1.8116e−1 (8.21e−2) | 1.7292e−1 (1.58e−2) | 2.1431e−1 (2.39e−2) | 3.2327e−2 (1.10e−2) | 2.1027e−1 (5.33e−3) |
| ZDT4    | 4.9738e−1 (1.64e−1) | 9.7793e−1 (4.00e−1) | 1.9289e−2 (5.63e−3) | 9.9516e−1 (3.71e−1) | 5.5132e−3 (6.78e−4) |
| ZDT6    | 8.4019e−2 (3.06e−2) | 6.1306e−1 (1.72e−1) | 1.6727e+0 (8.36e−2) | 4.0900e−1 (1.26e−1) | 2.4514e−3 (5.03e−4) |
| UF1     | 3.3551e−1 (1.05e−1) | 1.5652e−1 (3.35e−2) | 2.8179e−1 (6.29e−2) | 1.0765e−1 (1.99e−2) | 4.8899e−2 (6.68e−3) |
| UF2     | 1.9302e−1 (6.56e−2) | 7.4088e−2 (6.64e−3) | 7.1500e−2 (8.04e−3) | 6.8361e−2 (4.93e−3) | 2.9779e−2 (3.48e−3) |
| UF3     | 3.4274e−1 (3.80e−2) | 4.6760e−1 (1.01e−2) | 3.8362e−1 (2.46e−2) | 4.3167e−1 (2.13e−2) | 1.6243e−1 (2.71e−2) |
| UF4     | 1.1401e−1 (5.27e−3) | 9.0784e−2 (3.38e−3) | 1.2932e−1 (3.95e−3) | 8.1514e−2 (2.87e−3) | 5.6045e−2 (5.08e−3) |
| UF5     | 1.4316e+0 (3.16e−1) | 1.4234e+0 (3.01e−1) | 1.9335e+0 (2.64e−1) | 7.2150e−1 (1.60e−1) | 6.2702e−1 (1.74e−1) |
| UF6     | 6.0568e−1 (2.02e−1) | 8.6673e−1 (1.45e−1) | 1.2514e+0 (2.32e−1) | 5.0743e−1 (7.51e−2) | 3.9029e−1 (5.05e−2) |
| UF7     | 4.4728e−1 (1.61e−1) | 1.5309e−1 (5.60e−2) | 3.3829e−1 (6.77e−2) | 1.7709e−1 (1.03e−1) | 4.9987e−2 (6.84e−3) |
| UF8     | 4.9214e−1 (2.29e−1) | 3.3501e−1 (1.92e−2) | 2.7630e−1 (1.25e−2) | 3.8478e−1 (4.43e−2) | 4.2629e−1 (7.98e−2) |
those of the four MOEAs on ZDT1. Figure 8 shows that IDMOPSO has better convergence and distribution, and the other algorithms have poor performance on ZDT2. Additionally, Figures 9 and 10 show that the distribution and convergence of IDMOPSO are significantly better than those of the other four MOEAs on UF9 and UF10. From the above empirical results, IDMOPSO achieves a significantly better convergence than the other four MOEAs, which can conclude that IDMOPSO has a better performance in solving MOPs.
4.2.3. Comprehensive Performance Analysis of the Algorithm.
To compare the comprehensive performance of IDMOPSO and the other eight algorithms, Table 9 shows the ranking of IGD mean values on 22 test functions by sorting the dictionary order. Here, IDMOPSO obtained thirteen best IGD, MOEA/D obtained one best IGD, MOEADD obtained two best IGD, MOMBI-II obtained two best IGD, dMOPSO and MPSO/D had not best IGD, SMPSO obtained the two best IGD, and NMPSO obtained one best IGD. From Table 9, it can be concluded that the comprehensive performance of IDMOPSO is better than that of the other eight algorithms.

5. Conclusion
Multiobjective particle swarm optimization (MOPSO) has been widely used in science and engineering; however, providing a reasonable solution among many noninferior solutions is critical in all applications. In this paper, we propose a multiobjective particle swarm optimization based on ideal distance (IDMOPSO), where an ideal point and the degree of approximation is introduced as the criterion to measure the quality of the particle and is applied to the control of the external archive size and the selection of global learning exemplars. In addition, the cosine control factor is used to dynamically adjust the flying of particles in the swarm, which effectively improves the algorithm’s optimization ability and convergence ability. Furthermore, the cosine factor is applied to mutate the position of the particles to prevent the algorithm from falling into the local optimum. In the experiment simulation, the nine algorithms IDMOPSO, MPSO/D, SMPSO, NMPSO, dMOPSO, MOEA/D, MOEADD, IMMOEA, and MOMBI-II were tested on the ZDT, UF, and DTLZ series functions. The experimental results show that IDMOPSO has stronger competitiveness than other algorithms in IGD and HV indicators. Therefore, IDMOPSO can be regarded as an effective algorithm for solving multiobjective optimization problems.

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 no. 71461027) and Innovative Talent Team in Guizhou Province (Qian Ke HE Pingtai Rencai [2016]5619).

References
[1] S. Nguyen, M. Zhang, M. Johnston, and K. C. Tan, “Automatic design of scheduling policies for dynamic multi-objective job shop scheduling via cooperative coevolution genetic programming,” IEEE Transactions on Evolutionary Computation, vol. 18, no. 2, pp. 193–208, 2014.
[2] L. Li, W. Wang, and X. Xu, “Multi-objective particle swarm optimization based on global margin ranking,” Information Sciences, vol. 375, pp. 30–47, 2017.
[3] Q. F. Zhang and H. Li, “Moea/d: a multi-objective evolutionary algorithm based on decomposition,” IEEE Transactions on Evolutionary Computation, vol. 11, no. 6, pp. 712–731, 2007.
[4] K. Zou, Y. Liu, S. Wang, N. Li, and Y. Wu, “A multiobjective particle swarm optimization algorithm based on grid technique and multistrategy,” Journal of Mathematics, vol. 2021, pp. 1–17, 2021.
[5] J. E. Kennedy and R. Eberhart, “Particle swarm optimization,” in Proceedings of the IEEE International Conference on Neural Networks, pp. 1942–1948, New York, NY, USA, 1995.
[6] C. A. C. Coello, G. T. Pulido, and M. S. Lechuga, “Handling multiple objectives with particle swarm optimization,” IEEE Transactions on Evolutionary Computation, vol. 8, no. 3, pp. 256–279, 2004.
[7] H. Han, W. Lu, and J. Qiao, “An adaptive multiobjective particle swarm optimization based on multiple adaptive methods,” IEEE Transactions on Cybernetics, vol. 47, no. 9, pp. 2754–2767, 2017.
[8] Y. Zhang, D.-W. Gong, and Z. Ding, “A bare-bones multiobjective particle swarm optimization algorithm for environmental/economic dispatch,” Information Sciences, vol. 192, pp. 213–227, 2012.
[9] R. Leng, A. Ouyang, Y. Liu, L. Yuan, and Z. Wu, “A multiobjective particle swarm optimization based on grid distance,” International Journal of Pattern Recognition and Artificial Intelligence, vol. 34, no. 3, 2020.
[10] X. Zhang, X. Zheng, R. Cheng, J. Qiu, and Y. Jin, “A competitive mechanism based multi-objective particle swarm optimizer with fast convergence,” Information Sciences, vol. 427, pp. 63–76, 2018.
[11] H. Han, W. Lu, L. Zhang, and J. Qiao, “Adaptive gradient multi-objective particle swarm optimization,” IEEE Transactions on Cybernetics, vol. 48, no. 11, pp. 1–13, 2017.
[12] W.-N. Chen, J. Zhang, Y. Lin et al., “Particle swarm optimization with an aging leader and challengers,” IEEE Transactions on Evolutionary Computation, vol. 17, no. 2, pp. 241–258, 2013.
[13] W. Deng, J. J. Xu, X. Gao, and H. Zhao, “An enhanced msiqe algorithm with novel multiple strategies for global optimization problems,” IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 52, no. 3, pp. 1–10, 2020.
[14] Y. Zhang, D.-w. Gong, J.-y. Sun, and B.-y. Qu, “A decomposition-based archiving approach for multi-objective evolutionary optimization,” Information Sciences, vol. 430–431, pp. 397–413, 2018.
[15] W. Deng, S. F. Shang, and C. X. Xing, “Quantum differential evolution with cooperative coevolution framework and hybrid mutation strategy for large scale optimization,” Knowledge Based Systems, vol. 224, 2021.
[16] C. Dai, Y. Wang, and M. Ye, “A new multi-objective particle swarm optimization algorithm based on decomposition,” Information Sciences, vol. 325, pp. 541–557, 2015.
[17] S. Z. Martínez and C. A. Coello Coello, “A multi-objective particle swarm optimizer based on decomposition,” in Proceedings of the Conference on Genetic and Evolutionary Computation, pp. 69–76, Dublin, Ireland, July 2011.
[18] L. Chen, H. L. Liu, C. Lu, Y. M. Cheung, and J. Zhang, “A novel multi-objective particle swarm optimization with
multiple search strategies,” *European Journal of Operational Research*, vol. 247, no. 3, pp. 732–744, 2015.

[19] D. Brockhoff, T. Wagner, and H. Trautmann, “On the properties of the R2 indicator,” *Academic Medicine*, Philadelphia, PA, USA, July 2012.

[20] K. Deb, S. Agarwal, A. Pratap, and T. Meyarivan, “A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 849–858, 2002.

[21] A. J. Nebro, J. J. Durillo, J. Garcia-Nieto, C. A. Coello Coello, F. Luna, and E. Alba, “SMPSO: a new PSO-based metaheuristic for multi-objective optimization,” in *Proceedings of the Computational Intelligence in Multi-Criteria Decision-Making*, pp. 66–73, Nashville, TN, USA, April 2009.

[22] P. Wei and Q. Zhang, “A decomposition-based multi-objective particle swarm optimization Algorithm for continuous optimization problems,” in *Proceedings of the IEEE International Conference on Granular Computing*, pp. 534–537, Hangzhou, China, August 2008.

[23] F. Han, W.-T. Chen, Q.-H. Ling, and H. Han, “Multi-objective particle swarm optimization with adaptive strategies for feature selection,” *Swarm and Evolutionary Computation*, vol. 62, no. 6, Article ID 100847, 2021.

[24] B. Wu, W. Hu, J. Hu, and G. G. Yen, “Adaptive multi-objective particle swarm optimization based on evolutionary state estimation,” *IEEE Transactions on Cybernetics*, vol. 51, no. 7, pp. 1–14, 2019.

[25] K. Alkebsi and W. Du, “A fast multi-objective particle swarm optimization algorithm based on a new archive updating mechanism,” *IEEE access*, vol. 8, p. 1, 2020.

[26] E. Zitzler and L. Thiele, “Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach,” *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[27] C. A. C. Coello and N. C. Curz, “Solving multiobjective optimization problems using an artificial immune system,” *Genetic Programming and Evolvable Machines*, vol. 6, no. 2, pp. 163–190, 2005.

[28] S. Huband, P. Hingston, L. Barone, and L. White, “A review of multiobjective test problems and a scalable test problem toolkit,” *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 5, pp. 477–506, 2006.

[29] O. Schütze, X. Esquivel, A. Lara, and C. A. C. Coello, “Using the averaged hausdorff distance as a performance measure in evolutionary multiobjective optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 4, pp. 504–522, 2012.

[30] Z. L. Qiu, L. Songbai, Z. Qingling, C. Tang, and R. Song, “Particle swarm optimization with a balanceable fitness estimation for many-objective optimization problems,” *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 1, pp. 32–46, 2018.

[31] K. Li, K. Deb, Q. Zhang, and S. Kwong, “An evolutionary many-objective optimization algorithm based on dominance and decomposition,” *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 5, pp. 694–716, 2015.

[32] R. Hernandez Gomez and C. A. Coello Coello, “Improved metaheuristic based on the R2 indicator for many-objective optimization,” in *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 679–686, Madrid, Spain, July 2015.

[33] R. Cheng, Y. Jin, K. Narukawa, and B. Sendhoff, “A multi-objective evolutionary algorithm using Gaussian process-based inverse modeling,” *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 6, pp. 838–856, 2015.

[34] T. Ye, C. Ran, X. Zhang, and Y. J. Platemo, “A MATLAB platform for evolutionary multi-objective optimization,” *IEEE Computational Intelligence Magazine*, vol. 12, no. 4, pp. 73–87, 2017.

[35] Q. Zhang, A. Zhou, S. Zhao, P. N. Suganthan, and S. Tiwari, *Multi-objective Optimization Test Instances for the CEC 2009 Special Session and Competition*, Mechanical Engineering, New York, NY, USA, 2008.