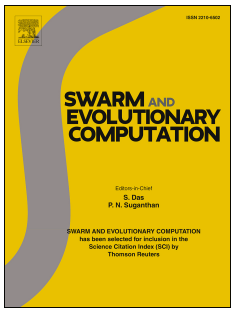

A random dynamic grouping based weight optimization framework for large-scale multi-objective optimization problems

Ruochen Liu, Jin Liu, Yifan Li, Jing Liu

PII: S2210-6502(19)30226-3

DOI: https://doi.org/10.1016/j.swevo.2020.100684

Reference: SWEVO 100684

To appear in: *Swarm and Evolutionary Computation BASE DATA*

Received Date: 29 March 2019

Revised Date: 12 March 2020

Accepted Date: 13 March 2020

Please cite this article as: R. Liu, J. Liu, Y. Li, J. Liu, A random dynamic grouping based weight optimization framework for large-scale multi-objective optimization problems, *Swarm and Evolutionary Computation BASE DATA* (2020), doi: https://doi.org/10.1016/j.swevo.2020.100684.

CRediT author statement

Ruochen Liu： Conceptualization, Methodology , Supervision, Project administration, Funding acquisition.

Jin Liu： Software, Validation, Writing - Original Draft.

Yifan Li：Formal analysis, Data curation, Writing - Review & Editing.

Jing Liu: Writing - Review & Editing, Project administration, Funding acquisition.

A Random Dynamic Grouping Based Weight Optimization Framework for Large-scale Multi-Objective Optimization Problems

Ruochen Liu[1], Jin Liu[1], Yifan Li[1], Jing Liu[1]

[a]*Key Lab of Intelligent Perception and Image Understanding of Ministry of Education, International Center of Intelligent Perception and Computation, Xidian University, Xi'an, 710071, China*

Abstract

For large-scale multi-objective problems (LSMOPs), it is necessary to get a good grouping strategy or another way to reduce dimensions because of "the curse of dimensions". In this paper, a weighted optimization framework with random dynamic grouping is proposed for large-scale problems. A weight optimization framework utilizes a problem transformation scheme in which weights are chosen to be optimized instead of the decision variables in order to reduce the dimensionality of the search space. Random dynamic grouping is used to determine sizes of each group adaptively. And multi-objective particle swarm optimization with multiple search strategies (MMOPSO) is employed as an optimizer for both original variables and weight variables. The proposed algorithm is performed on 28 benchmark test problems with 1000 dimensions, and the experimental results show that it can get better performance than some the-state-of-art

*Corresponding author
Email address: ruochenliu@xidian.edu.cn (Ruochen Liu)

algorithms in fewer function evaluations. In addition, it can be extended to solve LSMOPs with 5000 dimensions.

Keywords: large-scale multi-objective problem, weighted optimization framework, random dynamic grouping, MMOPSO

1. Introduction

Up to now, a lot of methods have been proposed for multi-objective optimization problems (MOPs), among which the most common one is evolutionary algorithm (EA) [1, 2, 3, 4, 5, 6]. For the past few years, the continuous development of optimization algorithms makes it possible to solve more complex tasks like dealing with MOP with a large number of objectives or decision variables that is often referred to as large-scale optimization (LSO). When the dimension of the search space, that is, the number of variables increases, the performance of the classical evolutionary computing method tends to decrease. In both single-objective optimization (SOP) and multi-objective optimization, it is an ongoing challenge to discover the optimal solution for a problem with a large number of variables. Cooperative Coevolution (CC) [7] and its variants based methods [8, 9, 10, 11, 12] have been developed to cope with the issue, and these methods usually involve a special mechanism to divide variables into multiple groups.

In CC based algorithms, it is necessary to choose a suitable decomposition strategy for obtaining satisfactory performance. For a good decomposition strategy, interactive variables are assigned to the same group, while independent variables are classified to different groups. There are two decomposition strategies, i.e. fixed grouping method [13, 14] and dynamic

grouping method [15, 16]. In fixed grouping method [13, 14], it needs a large number of function evaluations to determine the correlation between variables and divide them. After that, the variables will keep in the same group until the optimization process ends. While in dynamic grouping method, how to group the variables are determined dynamically. In general, the dynamic method does not need extra function evaluations, and the variables will be regrouped before every optimization. Two latest multi-objective LSO methods, i.e., evolutionary algorithm based on decision variable clustering approach (MOEA/DVA) [8] and large-scale many-objective evolutionary algorithm (LMEA) [9], are inspired from the idea of fixed decomposition strategy.

In MOEA/DVA, a new grouping mechanism is proposed to determine whether decision variables contribute to convergence, diversity, or both first. Then, the interaction between the variables is found. Through this information, a number of clusters of decision variables are obtained and they are maybe related to convergence, related to diversity, and mixed. The following MOEA/DVA optimization process first focuses on those convergence variables, and then on these diverse ones. In LMEA, a clustering method is applied to divided the decision variables into distance groups and diversity groups, which is the similar as MOEA/DVA. The results in [8] show that MOEA/DVA have good performance when the dimensions of decision variables are up to 200. Similarly, LMEA also can get good performance for LSO problems with 1000-dimensional decision variables [9].

However, both of the two methods above have the same disadvan-

tage, i.e., they all need a lot of function evaluations to divide the variables. To solve this problem, Zille et al. proposed a weighted optimization framework (WOF) [11], which is relied on the methods of grouping variables, the transformation function, and weight vectors optimization. The grouping methods, such as linear grouping, ordered grouping and different grouping, are used to group the decision variables into some subgroups. After this, the problem transformation is designed to represent the decision variables in terms of the weights. The whole variables in the same group are assigned a same weight vector, which will be optimized in the following process. And then through EAs, such as NSGA-II [17] and SMPSO [18], the weight vectors rather than the original decision variables are optimized to reduce the dimensionality of the variables. A random dynamic grouping strategy (RDG) [19] is proposed. In this strategy, both the size of groups and the variables in which groups are dynamically decided. At the beginning, a group pool with different sizes is designed. And then, the size of group is selected by a probability, which is determined by the non-domination relation in each iteration, and then the variables are grouped randomly to divide the population into some different groups.

In this paper, we propose a new algorithm for large-scale multi-objective problems (LSMOPs) by introducing WOF and RDG into a multi-objective particle swarm optimization with multiple search strategies (MMOPSO) [20], which is denoted as WOF-MMOPSO-RDG. Firstly, we use MMOPSO to optimize the initial population just with very few evaluations to obtain some better solutions. Then, we choose the group size by their probability using RDG, and separate the variables obtained. After that, we assign a

weight for each set of variables, and optimize the weight vectors instead of the original variables by MMOPSO. In the end, the rest of function evaluations are used to optimize the final decision variables by MMOPSO for maintaining the diversity. The contributions of this paper are summarized as follows:

(1) The weighted optimization framework divides the variables into groups, and each group has their own weight vector. The weight vectors are optimized, rather than the decision variables, to reduce the dimensions to a certain extent.

(2) In each iteration, the size of each group is selected adaptively by a probability which is depended on its quality. Through this strategy, the algorithm can find more suitable groups for itself at different stages, and then linear grouping, ordered grouping or different grouping can be used to divide the decision variables into the selected group size.

(3) Since we choose to optimize the weight vector, this method is essentially a kind of "reducing dimension" method which aims to transform the large search space into relatively small one. Thus the time complexity will be greatly reduced, and a large number of function evaluation times will be saved.

In the remainder of the paper, we firstly retrospect some correlated knowledge in Section 2. Section 3 provides some details of the main procedure of the proposed algorithm. Section 4 elaborates the experimental design and simulation results, and empirically evaluates the performance of proposed algorithm. Section 5 comes to conclusion and discusses our future work.

2. Background

In this section, some basic definitions of multi-objective optimization are proposed, and a brief review of existing multi-objective evolution algorithms is given later. In the end, we review some of the work on large-scale optimization.

2.1. Multi-objective Optimization

Many practical problems can be expressed as mathematical problems with certain objectives that must be optimized. In addition, some of these problems have constraints. The problem is called multi-objective problem, that if there are more than one objective to be optimized. A MOP without constraints can be formulated as follows:

$$\begin{aligned} &Z: \min F(x) = (f_1(x), f_2(x), ..., f_m(x)) \\ &s.t. x \in \Omega \in R^D \end{aligned} \tag{1}$$

where *F(x)* is the objective space with m dimensions, expressed as $\{f_1(x), f_2(x), ..., f_m(x)\}$, and Ω is the decision variable space, which contain *D* dimensional decision variables, expressed as $x \in \Omega \in R^D$.

In MOPs, there are a number of solutions that do not dominate each other, called no-dominate set. Without loss of generality, we can define this relationship between the solutions as follows:

$$\forall i \in (1, 2, ..., M) : F_i(x) \leq F_i(y) \wedge \exists j \in (1, 2, ..., M) : F_j(x) < F_j(y) \tag{2}$$

If *x* and *y* satisfy this inequality, we can say that *x* dominates *y*, or *y* is dominated by *x*, expressed as $x \prec y$. Pareto optimal solutions are those that are independent of other solutions. Pareto set (PS) is a collection of all

Pareto optimal solutions in variable space, and Pareto front (PF) is a set of corresponding objective vectors.

It is important to solve such problems as precisely as possible with or without constraints. For MOPs, it is no longer possible to determine a single best solution. In addition, due to the complexity of MOPs in practical applications and the limitations of computing resources, the true Pareto-front is difficult to analyze or calculate accurately. To overcome the shortcomings of traditional algorithms, heuristic algorithms were exploited to find the best possible non-domination solution set to approximate the true PF.

2.2. Multi-objective Evolution Algorithm

Since many decades, multi-objective evolution algorithms (MOEAs) are a popular way to solve MOPs[21, 22, 23, 24, 25]. All these MOEAs aims to obtain a set close to the true PF while maintaining diversity. The non-dominance sort used in NSGA-II [17], SPEA2 [26], and SMPSO [18] is to obtain a solution set that is convergent to the true PF as close as possible. Meanwhile, a way to maintain the diversity of the set is utilized in these algorithms. Some algorithms are based on performance indicators, like hyper volume (HV) [27, 28] and inverted generational distance metric (IGD) [29] which are used to measure the fitness of individuals, is proposed, such as IBEA [30] and SIBEA [31]. Recently, the decomposition strategy is developed to solve a MOP, in which a multi-objective problem is divided into a number of single objective sub-problems, and the widely used algorithms based on decomposition are MOEA/D [32], MPSOD [33] and MMOPSO [20]. MMOPSO aims to improve the performance of tradi-

tional particle swarm optimization (PSO), in which two search strategies are proposed to update the velocity of each particle, which are useful to accelerate the convergence and maintain the diversity respectively in the population. Different from the normal PSO, MMOPSO uses decomposition approach to divide the MOP into a set of single objective problems (SOPs), and the personal-best and the global-best particles are the best of each sub-problem and all SOPs respectively. Therefore, MMOPSO can optimize the whole problem by optimizing the sub-problem separately.

However, most of MOEAs mentioned above are not good at solving large-scale problems, because they will lose their effectiveness and efficiency rapidly with the increase of dimensionality and the exponential growth in search space, which is called "the curse of dimensionality".

2.3. Large-Scale Optimization

LSO handles optimization problems involving a mass of variables. Recently, various single-objective optimizers with a lot of variables have been proposed in [34]. Among them, a competitive single-objective PSO algorithm was developed in [35] in 2015, which achieved good performances for large-scale benchmarking of 5000 decision variables. But the cooperative co-evolution (CC) is a more popular choice. Followed the work of Porter and De [7, 36] , CC has been applied to all kinds of LSO algorithms [16]. For CC, the composition of each new candidate solution must come from different independent sub-populations that are optimized. And these sub-populations contain a group of decision variables.

CC is now applied widely to handle large-scale SOPs [16, 37, 38, 39]. [16] applied the CC method to SOPs using two characteristics. The first

one is to iteratively divide variables and regroup them into subcomponents. The second one is the weighting strategy, that applies the same weight to each variable in the same subcomponent, which is obtained by the random, the best and the worst members in the entire population and then optimizes the weight. The frequency of regrouping is random in [16]. They simultaneously optimize several variables (in each subcomponent) by changing only one weight value. In [37], an approach for selecting the suitable upper and lower bounds of weights is employed. Lately, Li [38] pointed out that using weighting methods is not as efficient as regrouping the improved variables frequently in CC. These methods based on CC works mentioned above are tested on the benchmark problems with 1000-dimensional decision variables.

Compared with the other researches above mentioned on large-scale SOPs, there are few studies on MOPs with lots of variables. The algorithm developed in [40] has a good performance on ZDT [41] problem, in which CC combines a differential evolution algorithm called GDE3 [42]. Authors test on LSMOPs with 200 to 5000 decision variables. However, they did not report the experimental results on more complex benchmarks like WFG [43] or DTLZ [44]. In addition, although the good approximation of PF of ZDT is better than the traditional method so far, it still requires a mass of function calculations ranged from 100,000 to 2,000,000. Iorio and Li [45] earlier combined CC with NSGA-II [17], and tested on ZDT issues with ten and thirty variables.

In [46], the inseparable problems are considered and researched about the distribution of variables in subcomponents or groups. In order to find

the variables' optimal distribution in subcomponents, a learning method is proposed to determine the interactions between variables. In [13], differential grouping (DG) was used to exploit a modified distribution of variables in a single-objective algorithm with CC. Other development methods of grouping such variables have been proposed in [14] and [47].

And the idea of the DG was also used in the LSMOPs, that is MOEA/DVA [8] and LMEA [9]. MOEA/DVA used decision variables analysis to divide the decision variables, and decomposed a LSMOP into several low-dimensional sub-MOPs, which was optimized independently. Similar to MOEA/DVA, LMEA also divided the decision variables into two types, i.e. the convergence-related and the diversity-related. And the two types of variables were optimized by two different methods, respectively. But both of them spent a large number of function evaluations finding the interactions for grouping. To save computing resources, WOF [11] was proposed, which was based on the problem transformation. All the decision variables were divided to some subgroups and each group was assigned with a weight vector. And then optimizing the weight vectors replaced optimizing the decision space of the original problem.

3. The proposed Algorithm

In this section, we introduce the proposed algorithm, which uses a weighted optimization framework with random dynamic grouping and MMOPSO as an optimizer, referred to as WOF-MMOPSO-RDG. At the beginning, we utilize random dynamic grouping strategy to set up a group size pool, and each group size has its own probability, which will be the

basis of roulette. After that, MMOPSO is used to optimize the initial population with very few function evaluations, and then we pick a number of solutions. For each picked solution, we divide it into some groups and each group will be assigned a weight vector by the transformation function, and use MMOPSO to optimize these weight variables. After optimization the best weights are applied to the original population to obtain the weighted population, and the duplicate solutions will be eliminated, and the non-dominated sorting process is performed on the union solution of the parent population and the weighted population. In each cycle except the first iteration, we calculate and change the possibility of each group size by the C-metric [48], and reselect the group size. When the end of the cycle condition is satisfied, MMOPSO is optimized the population to maintain the diversity until the rest of function evaluations used. We present the primary steps of WOF-MMOPSO-RDG in Algorithm 1.

3.1. The weighted optimization framework

The weighted optimization framework was developed in [10] and [11] with the initial aim of overcoming some of the disadvantages of cooperative coevolution. These methods essentially reduce problem's dimensionality by changing a large part of the decision variables simultaneously, and their extent of change is the same, which is achieved by a so-called transformation function. The transformation function can assign each weight value to a set of decision variables. The number of weight vectors is identical to an amount of groups and they are then updated by a separate optimization step.

The n-dimensional decision variables $(x_1, ..., x_n)$ are separated to r groups,

Algorithm 1 WOF-MMOPSO-RDG

Input: the size of population N, the maximal number of function evaluations FE_{max}, the dimension n, the group size pool gs, the selected probability of group size P_r

Output: the final population *S*

1: Initialization, random initial population *S*;
2: **Repeat**
3: Select a group size r from gs by their possibility;
4: $S \leftarrow$ Utilize MMOPSO to optimize the initial population S for t_1 evaluations;
5: $S \leftarrow$ use Algorithm 2 to obtain the population after weight optimization;
6: Calculate P_r of each group size by Algorithm 5, reselect the group size r by *roulette options*;
7: **until** $\delta * FE_{max}$ used
8: $S \leftarrow$ Apply MMOPSO to optimize the final population *S*, until all the evaluations are used;
9: **return** the final population *S*

and a new weight variable is assigned into each group, called ω_j, $j = 1, ..., r$. The values of the decision variable x_i with the corresponding weight variable ω_j, by mean of a transformation function, is gotten to evaluate the optimization problem, instead of only the original decision variables x_i. Thus, the new weight variable $\omega = (\omega_1, ..., \omega_r)$ is generated and it can be used to optimize the problem independently.

The pseudocode for the algorithm is shown in Algorithm 2.

Algorithm 2 WOF

Input: The group size r, the grouping method G, the transformation function ψ , the population S

Output: the optimized population *S*

1: $\{x_1, ..., x_q\} \leftarrow$ Select *q* solutions based on the Crowding Distance from the first non-dominated front of *S*.
2: **for** $k = 1 : q$ **do**
3: $W_k \leftarrow$ Apply the Algorithm 3 to optimize the weight variables and the solutions.
4: **end for**
5: $S \leftarrow$ Update the population $\{W_1, W_2, ..., W_q, S\}$ by Algorithm 4.

WOF applies the existing meta-heuristic in the framework, and by reducing n variables to r variables, the algorithm can search more efficiently in a smaller space. Since the optimal solution cannot be guaranteed to be contained in a small search space, optimizing the weight ω alternates with optimizing the original decision variables. For remaining number of evaluations, the original decision variables will be optimized to obtain a more diverse solution set, which is shown in Step 8 of Algorithm 1.

3.1.1. Grouping mechanism

In order to divide the n-dimensional decision variables into r groups, we need to use grouping strategy, which usually place these variables that with strong interactions in the same group (in the inseparable optimization problem). Here, we briefly explain the different grouping methods used in this paper. Although the first three methods are very simple and do not use any information of the objective function, differential grouping (DG) includes an intelligent mechanism based on problem analysis. We will use DG method for comparison, although it was developed for single-objective optimization.

(1) Linear grouping: It is to arrange all n variables in natural order, and then divide the first n/r variables into the first group according to a fixed group size r, and so on.

(2) Random grouping: It means that the whole variables are separated into fixed r groups, and variables in each group are randomly assigned.

(3) Ordered grouping: In ordered grouping, all the variables are sorted by their absolute values, and then they are assigned to different groups with a fixed grouping size r, that is the first n/r variables into the first group, and so on.

(4) Differential grouping: DG was developed by [47] for CC based single-objective optimization with the goal of detecting variable interactions before optimization. The number and size of groups are automatically set through DG mechanism. in simple terms, DG compares the difference values before and after the change of other variables $x_{h(h=1,2,\ldots,n)}$.

When the value of x_i is changed, the amount of change in $f(x)$ remains unchanged regardless of the value of the other variable x_h, so the variables x_i and x_h are not interact, so they can be divided into different groups. Otherwise, it means they are interactive, which will be distributed to the same group. Two disadvantages should be mentioned: 1) Differential grouping uses a lot of computing resources, which makes the algorithm expensive; 2) The DG mechanism was developed for SOPs. Its design does not take into account MOPs, so it can not directly apply to MOPs.

These four grouping mechanisms take different computational cost effort required in each iteration. During the whole process, linear grouping will not change, so there is no need to recalculate in each iteration. For DG, it is pre-computed before optimization start, so the recalculate is no need, which is the same as linear grouping. However, random grouping and ordered grouping need update at each time when a problem need to transform. In the former, a new random allocation is assigned to the *r* group, and in the latter each variable that selects x_k is ordered in each problem conversion step.

3.1.2. Transformation functions

In this part, we will introduce some different transformation functions $\psi(\omega, x)$, and all of them have their own benefits and drawbacks. All the decision variables have been divided into *r* groups before the transformation.

(1) Produce Transformation (ψ_1):the basic transformation function is $\psi(\omega, x) :=$

$(\omega_1 x_1, ..., \omega_r x_n)$, so for each of decision variables, the first function is as follows:

$$x_{i,new} = \omega_j \cdot x_{i,old} \tag{3}$$

Where $\omega_j \in [0, 2]$. The drawback for ψ_1 is the positive or negative values of $x_{i,old}$ can only obtain the positive or negative new values. Furthermore,the progress that can be made by changing a weight variable ω_j is determined by the absolute value of x_i , which is not what we excepted. For example, there is a variable x_i with a domain of [0,5], and the vector ω_j with a domain of [0,2]. If $x_i = 0.2$, the original search space is explored in the interval [0,0.4] maximally for the optimization algorithm, while $x_i = 4$, the whole domain of x_i can be covered. This shows that due to we do not know the degree of the variables in advance, the absolute value is not relied on in a search.

(2) p-Value Transformation (ψ_2): It aims to decouple the relationship between the absolute value of the original variable and the reachable domain space, and it is given in the following.

$$x_{i,new} = x_{i,old} + p \cdot (x_{\max} - x_{\min}) \cdot (\omega_j - 1.0) \tag{4}$$

Where $\omega_j \in [0, 2]$ and $p \in [0, 1]$. In Eq.(4), the value of x_i always changes within a certain range of its original value, where the range is decided by p. $x_{\max}$ and $x_{\min}$ represent the upper and lower bounds of the variables x_i. As the range of the value x_i is [0,2], we can see that ψ_2 can translate the variables into the interval [-1,1], which describes the change in the original value in one direction or the other. The ac-

tual amounts of change are not determined by ω_j, instead we can alter the parameter p to change it. For example, if we set $p = 0.5$, The value of x_i changes by 50% of the width of the variable's field, centered on the original value ω_j.

The weighted optimization is shown in Algorithm 3.

Algorithm 3 Weighted Optimization

Input: The group size r, the selected solution x_k, the grouping method G, the transformation function ψ

Output: the population of weights $\mathbf{W}_k$

1: $\mathbf{W}_k \leftarrow$ Randomly initialize the weighted population in its domain;
2: Divide n variables into r groups by the grouping method G;
3: Transform the problem with x_k and $\mathbf{W}_k$ by the function ψ;
4: $\mathbf{W}_k \leftarrow$ Using MMOPSO to optimize the $\mathbf{W}_k$ for t_2 evaluations;
5: **return** $\mathbf{W}_k$

After the weighted optimization, the optimized weight vectors are used to the original population, which is shown in Algorithm 4. For each $\mathbf{W}_k$, we choose the one with the largest crowding distance when it is applied on the original population *S*. And then the obtained solutions S'_k are combined with *S*, and sort them by non-dominated sorting. Certainly, duplicate solutions are deleted before sorting for preserving the diversity of population.

3.2. Random Dynamic Grouping Strategy

At the beginning of this strategy, a group size pool is set, which include k different group size, expressed as GS $= \{g_1, g_2, ..., g_k\}$. How many

Algorithm 4 Update Population

Input: the weight population $\mathbf{W}_1, ..., \mathbf{W}_q$, the population S

Output: the updated population S

1: **for** $k = 1 : q$ **do**
2: $\omega_k \leftarrow$ Select one individual that have the largest crowding distance from $\mathbf{W}_k$;
3: $S_k^{'} \leftarrow$ Apply ω_k to the population S;
4: **end for**
5: $S \leftarrow$ Apply non-dominated sorting in $S \cup \{S_k^{'}\}_{k=1,...,q}$;
6: **return** S

groups the whole decision variables will be divided into are determined by the probability of the group size in the pool. In order to make the right choice, we use the performance improvement brought by using group size to determine its probability.

Corresponding to the group size pool, the probability selection list is supposed as R $= \{r_1, r_2, ..., r_k\}$, which store the improvement of performance for each group size. Each group size has the same probability in the beginning, and the elements in R is initialized to 1. During evolution, the relative performance improvement of the selected group size is computed as:

$$r_i = C(A, B) = \frac{|\{u \in B | \exists v \in A : v \prec u\}|}{N} \tag{5}$$

where N represents the size of population , A and B are two solution sets, that are closed to the PF of an MOP.

In this paper, C-metric introduced in [48] is used to compute the relative increase of performance, which defined as the percentage of the solu-

tion in *B* that are dominated by at least one in *A*, and both *A* and *B* have the same size. $C(A, B) = 1$ represents that all the solutions in B can be dominated by some of the solutions in *A*. Similarly, if there is no solution in *B* that can be dominated by one of the solutions in *A*, the value of $C(A, B)$ is zero.

So, the probability of each group size being selected P $= \{p_1, p_2, ..., p_k\}$ can be calculated as follows:

$$p_i = \frac{e^{5*r_i}}{\sum_{j=1}^{k} e^{5*r_j}}, (i = 1, 2, ..., k) \tag{6}$$

Finally, for selecting the group size from GS by the probability computed above, roulette wheel selection method is utilized.

The procedure of the proposed RDG is presented in Algorithm 5.

Algorithm 5 RDG

Input: the group size pool GS

Output: group size *r*

1: Calculate the probability selection list R by the *C-metric*;
2: Calculate the probability P of each group size by the Eq.(6);
3: Select a group size r from GS utilizing the roulette wheel selection method.

3.3. MMOPSO

MMOPSO [20] decomposes a MOP into a set of SOP and then each SOP is optimized by a particle. Two strategies for updating particle's velocity were proposed in MMOPSO to improve the convergence rate and

maintain the diversity and their co-operation is governed by predetermined thresholds. An external archive with a finite size will be established to memory all non-domination solutions. Once the external archives are filled in, only the non-domination solutions that have higher values of crowding distance are left, which are considered as elite solutions and on behalf of the whole PF well. In MMOPSO, individual optimal and global optimal particles are the optimal values of each aggregation problem and all SOP, respectively. The new updating strategies of the velocity of particle is shown as follows:

$$v_i(t) = \omega \cdot v_i(t-1) + C_1 \cdot r_1 \cdot (x_{pbesti} - x_i) \tag{7}$$

$$v_i(t) = \omega \cdot v_i(t-1) + C_2 \cdot r_2 \cdot (x_{gbesti} - x_i) \tag{8}$$

Where $v_i(t)$ represents the velocity of particle *i* at time *t*, ω is the inertia weight, C_1 and C_2 are acceleration coefficient, x_{pbesti} and x_{gbesti} represents individual cognition and social cognition of particle *i* respectively, r_1 and r_2 are random numbers between 0 and 1. When a random number generated is less than a threshold, the velocity of the particle is updated by using Eq.(7), instead, updated by Eq.(8).

4. Experiments and Results

In this part, the proposed algorithm is performed on WFG [43] and UF [49] test suits with 1000-dimentional to 5000-dimentional decision variables. And the Wilcoxon signed rank [50] is used to evaluate the difference of the solution sets with comparing algorithms. "+", "-" and "≈" denote the performance of WOF-MMOPSO-RDG is better than, worse than and similar to that of the corresponding comparing algorithms, respectively.

4.1. Performance Measurements

We use the relative value of the hyper volume (rHV) [11], i.e. the obtained hyper-volume divided by the hyper-volume of true PF. By multiplying the nadir point of the real PF sample for each question by 2.0 for each dimension, a reference point for calculating the hypervolume in all cases can be obtained. In experiments, the higher the index is, the better the performance is.

The other performance indicator is inverted generational distance (IGD) [51]. It can assess convergence and uniformity performance at the same time. IGD is calculated as follows:

$$IGD(P^*, P) = \frac{\sum_{v \in P^*} d(v, P)}{|P^*|} \tag{9}$$

P^* presents a number of uniformly distributed solutions on the true PF, and P is the approximate solution set of the PF. $d(v, P)$ means the minimal Euclidean distance form v in P^* to the solution P. To get the values of metrics, for two-objective test problems the amount of solutions in P^* is set as 500, and for three-objective problems that is set as 2500. In test experiments, the smaller the index, the better the performance.

4.2. Parameter Settings

In the proposed algorithm, the maximum number of function evaluations is set to 100,000. The number of evaluations to optimize the original problem, denoted as t_1, is set to 1000, and the number of evaluations to optimize the transformed problem, denoted as t_2, is set to 500. The parameter q, which is used to represent the original population for weight optimization, is set to 3. The population size is set as 100. The parameter δ in

Algorithm 1 is set to 0.5. We choose the ordered grouping method and the p-value transformation function ψ_2, and the parameter p is set to 0.2. The group size pool includes five different size, i.e. 2, 4, 8, 10, 20. The parameters mentioned above is the same as [11]. The threshold in MMOPSO is set to 0.7, which is the same as [20]. The parameters of comparing algorithms, i.e. MOEA/DVA [8], LMEA [9], WOF [11], and MOEA/D-RDG[19]are set the same as the original papers.

Each algorithm runs 20 times independently to calculate the statistical values of metrics in the following experiments. The value of the best result among the compared algorithms is highlighted in bold in the experimental studies.

4.3. Experimental Result and Analysis

4.3.1. Experimental Result of problems with 1000 variables in WFG test problems

In this part, 2-objective and 3-objective WFG test problems are used to validate the algorithm we proposed. To compare the performance of our algorithm, we choose MOEA/DVA, LMEA, WOF and MOEA/D-RDG for comparison. The average rHV and its standard variance, the average IGD and its standard over 20 runs are presented in Table 1.

Table 1:The average and standard deviation value of rHV and IGD obtained by the compared algorithms with 1000 variables in WFG test problems

m			WOF-MMOPSO-RDG	WOF-MMOPSO		MOEA/DVA		LMEA		MOEA/D-RDG	
2	WFG1	rHV	**0.98765 (8.5147e-06)**	0.98684 (1.7053e-05)	+	0.57156 (2.8459e-07)	+	0.57172 (5.2604e-07)	+	0.59506 (5.6302e-06)	+
		IGD	**1.4221e-1 (2.5126e-04)**	1.5234e-01 (1.3333e-03)		2.3680e+00 (2.5316e-06)		2.3673e+00 (4.7161e-06)		2.2668e+00 (3.6579e-06)	
2	WFG2	rHV	**0.97300 (1.4704e-05)**	0.94632 (3.7323e-06)	+	0.47015 (4.3804e-04)	+	0.45746 (5.7961e-04)	+	0.71973 (3.5268e-04)	+
		IGD	**5.8106e-2 (1.7078e-05)**	1.1983e-1 (2.5639e-05)		1.4512e+0 (1.9131e-02)		1.5570e+0 (8.5946e-02)		6.9986e-1 (6.2586e-02)	

2	WFG3	rHV	**0.97524**	0.97525	+	0.51078	+	0.51092	+	0.69365	+
			(1.6499e-05)	(3.7323e-06)		(2.2754e-05)		(3.3702e-05)		(4.3995e-05)	
		IGD	**7.3221e-2**	8.0026e-2		1.2738e+0		1.2720e+0		7.6843e-1	
			(2.5478e-04)	(1.5994e-04)		(2.9733e-04)		(1.2863e-03)		(2.6523e-03)	
2	WFG4	rHV	**0.96218**	0.96155	+	0.45973	+	0.45830	+	0.64208	+
			(4.1370e-06)	(5.8670e-06)		(2.2387e-05)		(1.5115e-05)		(6.3256e-05)	
		IGD	**9.5765e-2**	9.5926e-1		1.5385e+0		1.5432e+0		8.6853e-1	
			(9.3469e-06)	(1.5211e-05)		(4.1199e-04)		(8.3773e-04)		(9.3622e-04)	
2	WFG5	rHV	**0.95435**	0.95306	≈	0.40787	+	0.40652	+	0.61653	+
			(6.1724e-06)	(6.7125e-06)		(1.7070e-05)		(1.7580e-05)		(6.2668e-05)	
		IGD	**7.2056e-2**	7.2368e-2		1.5279e+0		1.5337e+0		8.3096e-1	
			(8.4100e-07)	(8.1477e-07)		(3.9089e-04)		(1.8653e-03)		(3.2899e-03)	
2	WFG6	rHV	**0.99856**	0.99695	+	0.40651	+	0.36609	+	0.55763	+
			(3.1643e-08)	(1.6636e-08)		(6.3982e-05)		(8.5336e-05)		(9.6328e-05)	
		IGD	**1.7268e-2**	5.7543e-2		1.5337e+0		1.6141e+0		1.0113e+0	
			(7.1633e-07)	(4.5691e-07)		(2.3369e-04)		(3.3163e-04)		(6.1202e-04)	
2	WFG7	rHV	0.97681	**0.97802**	-	0.50058	+	0.49991	+	0.70561	+
			(1.1498e-05)	**(8.0761e-06)**		(5.5595e-04)		(1.2099e-04)		(5.2314e-04)	
		IGD	6.5210e-2	**6.1926e-2**		1.3589e+0		1.3624e+0		7.3781e-1	
			(8.0179e-05)	**(5.0896e-05)**		(2.1130e-03)		(1.5197e-03)		(5.3652e-03)	
2	WFG8	rHV	**0.95437**	0.95362	≈	0.40638	+	0.40701	+	0.64179	+
			(2.0349e-05)	(1.7850e-05)		(2.3651e-04)		(1.2911e-05)		(3.2569e-05)	
		IGD	**1.2241e-1**	1.2453e-1		1.5753e+0		1.5752e+0		7.8075e-1	
			(1.3751e-04)	(1.2311e-04)		(9.2584e-04)		(5.9442e-04)		(5.3651e-04)	
2	WFG9	rHV	**0.97256**	0.97130	≈	0.36415	+	0.36890	+	0.55641	+
			(5.0872e-06)	(6.6772e-06)		(2.1332e-04)		(1.6140e-05)		(3.2564e-05)	
		IGD	**1.7268e-2**	5.7543e-2		1.5337e+0		1.6141e+0		1.0615e+0	
			(7.1633e-07)	(4.5691e-07)		(2.3369e-04)		(3.3163e-04)		(3.2658e-04)	
3	WFG1	rHV	**0.99725**	0.99690	+	0.54097	+	0.54098	+	0.60993	+
			(7.0882e-07)	(6.8479e-07)		(3.1555e-04)		(2.0036e-04)		(3.6985e-04)	
		IGD	**3.4492e-1**	3.7534e-1		2.7530e+0		2.7628e+0		2.3908e+0	
			(3.1645e-04)	(4.0979e-03)		(1.3242e-03)		(6.4123e-03)		(6.6332e-03)	
3	WFG2	rHV	**0.96156**	0.86172	+	0.30548	+	0.30600	+	0.66862	+
			(1.3758e-05)	(1.3792e-04)		(4.3566e-04)		(1.0632e-05)		(6.3665e-05)	
		IGD	**2.9201e-1**	2.9826e-1		2.7530e+0		2.3292e+0		1.1129e+0	
			(3.7148e-04)	(4.2841e-04)		(3.2231e-03)		(5.3906e-03)		(6.3215e-03)	
3	WFG3	rHV	**0.96156**	0.95733	+	0.49755	+	0.48975	+	0.54743	+
			(1.5084e-05)	(2.5934e-05)		(6.2321e-04)		(9.3569e-03)		(9.0232e-03)	
		IGD	**1.8612e-1**	2.0416e-1		1.6457e+0		1.6500e+0		8.8865e-1	
			(5.6374e-04)	(5.4470e-04)		(3.5770e-03)		(1.6980e-04)		(1.6980e-04)	
3	WFG4	rHV	**0.93986**	0.93825	≈	0.37302	+	0.37176	+	0.68432	+
			(5.6524e-05)	(2.9224e-06)		(5.4216e-04)		(1.0957e-05)		(5.3202e-05)	
		IGD	**3.2880e-1**	3.2954e-1		2.8834e+0		2.9199e+0		1.0819e+0	
			(8.1219e-05)	(1.6793e-04)		(6.2136e-04)		(1.8513e-04)		(1.2036e-04)	
3	WFG5	rHV	**0.95249**	0.95190	≈	0.31986	+	0.31801	+	0.63083	+
			(1.4912e-05)	(1.7610e-05)		(3.2598e-05)		(1.3145e-05)		(2.3025e-05)	
		IGD	**2.9761e-1**	2.9783e-1		2.5360e+0		2.5730e+0		1.0498e+0	
			(1.8578e-04)	(1.9957e-04)		(6.2158e-04)		(2.2353e-04)		(3.6521e-04)	
3	WFG6	rHV	0.98586	**0.98747**	-	0.28542	+	0.28838	+	0.61528	+
			(3.5264e-04)	**(2.8919e-04)**		(2.2231e-03)		(3.7925e-03)		(5.3236e-03)	

		IGD	2.9883e-1 (9.4427e-04)	**2.8974e-1 (6.3692e-04)**		2.3623e+0 (5.3217e-03)		2.3586e+0 (3.0280e-03)		1.3183e+0 (6.3258e-03)	
3	WFG7	rHV	**0.92499 (1.3721e-05)**	0.92460 (2.4139e-05)	≈	0.40765 (9.2325e-05)	+	0.41513 (1.3995e-04)	+	0.69355 (8.3256e-04)	+
		IGD	**3.5971e-1 (1.2678e-04)**	3.6256e-1 (2.1444e-04)		2.5040e+0 (6.3215e-04)		2.5051e+0 (2.3685e-04)		1.0101e+0 (9.3685e-03)	
3	WFG8	rHV	0.92068 (4.8973e-06)	**0.92341 (3.8604e-06)**	≈	0.31862 (2.3256e-04)	+	0.32416 (2.6582e-05)	+	0.67043 (6.3258e-05)	+
		IGD	3.6746e-1 (1.1063e-04)	**3.6176e-1 (1.1939e-04)**		2.6678e+0 (3.2136e-04)		2.6552e+0 (2.8406e-04)		1.0926e+0 (4.3025e-04)	
3	WFG9	rHV	**0.93590 (5.5661e-05)**	0.92958 (7.3077e-05)	+	0.29492 (3.2568e-05)	+	0.28939 (2.6582e-05)	+	0.59560 (5.3210e-05)	+
		IGD	**3.2993e-1 (1.0064e-04)**	3.4161e-1 (1.6844e-04)		2.3453e+0 (3.6665e-04)		2.4136e+0 (1.7478e-04)		1.3459e+0 (8.3202e-04)	
+/-/≈				8/2/8		18/0/0		18/0/0		18/0/0	

In Table 1, we can see that out of the 9 problems, 8 results of our proposed algorithm are the best, in which there are 3 problems obtained the similar solution set with WOF-MMOPSO, and the best of the remaining one problem is the WOF-MMOPSO in 2-objective problems. While in 3-objective problems, there are 7 results of our proposed algorithm are the best out of the 9 problems, in which there are 5 problems obtained the similar solution set with WOF-MMOPSO.

From the statistical results, it can be seen that under these 18 benchmark problems, the proposed WOF-MMOPSO-RDG is significantly better than the other four comparison algorithms on the 8 instances according to the Wilcoxon signed-rank test, and performs equally well on 8 instances when comparing with WOF-MMOPSO. These results confirm that WOF-MMOPSO-RDG has a better performance on WFG test problems.

4.3.2. Experimental Result of problems with 1000 variables in UF test problems

In this part, 2-objective UF1-UF7 test problems and 3-objective UF8-UF10 test problems are used to validate the algorithm we proposed. For

comparing the performance of our algorithm, the same comparing algorithms in Section 4.3.1 is chose. The average rHV and its standard variance, the average IGD and its standard over 20 runs are presented in Table 2.

Table 2:The average and standard deviation value of rHV and IGD obtained by the compared algorithms with 1000 variables in UF test problems

m			WOF-MMOPSO-RDG	WOF-MMOPSO		MOEA/DVA		LMEA		MOEA/D-RDG	
2	UF1	rHV	**0.95548** **(2.2200e-04)**	0.95027 (3.8147e-04)	+	5.4321e-04 (4.3566e-04)	+	8.4865e-04 (1.1753e-06)	+	4.8595e-04 (6.3256e-06)	+
		IGD	**9.7306e-2** **(4.2924e-04)**	1.0691e-1 (4.8599e-04)		2.1662e+0 (2.5639e-04)		2.1606e+0 (1.2509e-03)		2.1714e+0 (1.6036e-03)	
2	UF2	rHV	0.92844 (5.7029e-06)	**0.92960** **(5.5787e-06)**	≈	0.34304 (1.1381e-030)	+	0.34358 (2.4900e-05)	+	0.33842 (3.6200e-05)	+
		IGD	8.8820e-2 (3.9157e-06)	**8.7943e-2** **(4.8593e-06)**		9.4843e-0 (1.2468e-02)		9.4728e-0 (7.6521e-05)		9.4816e-0 (8.0232e-05)	
2	UF3	rHV	**0.98991** **(1.2122e-06)**	0.98573 (2.7524e-07)	+	0.19835 (3.8808e-05)	+	0.19756 (8.1975e-04)	+	0.20336 (7.9503e-04)	+
		IGD	**2.1565e-2** **(8.0608e-06)**	2.3951e-2 (1.9059e-06)		1.1245e+0 (3.1083e-03)		1.1248e+0 (3.1377e-03)		1.1269e+0 (4.0325e-03)	
2	UF4	rHV	**0.94959** **(7.6736e-05)**	0.94692 (1.5363e-04)	+	0.79652 (1.9394e-06)	+	0.79803 (1.0155e-05)	+	0.79546 (3.0635e-05)	+
		IGD	**6.1906e-2** **(1.0010e-04)**	6.4256e-2 (1.5941e-04)		2.2486e-0 (2.9819e-07)		2.2468e-0 (5.5588e-07)		2.2413e-0 (6.3258e-07)	
2	UF5	rHV	0.15583 (5.4583e-03)	**0.19365** **(5.0874e-03)**	-	/	+	/	+	/	+
		IGD	1.2162+0 (3.2026e-03)	**1.2006e+0** **(6.3542e-02)**		7.2516e+0 (6.3122e-03)		7.2368e+0 (2.7879e-03)		7.3246e+0 (2.6302e-03)	
2	UF6	rHV	**0.95576** **(8.1418e-03)**	0.90102 (1.2604e-02)	+	0.66079 (1.1785e-03)	+	0.75936 (1.4003e-03)	+	/	+
		IGD	**7.2021e-2** **(5.2032e-03)**	1.3125e-1 (7.6041e-02)		3.7922e-01 (8.3700e-03)		2.0191e-01 (4.1582e-04)		8.6466e+0 (4.1582e-04)	
2	UF7	rHV	**0.93845** **(1.7641e-03)**	0.89786 (7.0151e-05)	+	/	+	/	+	/	+
		IGD	**9.0395e-2** **(1.2188e-03)**	1.3321e-1 (9.7837e-05)		2.3123e+0 (9.9316e-04)		2.2319e+0 (1.2092e-03)		2.2458e+0 (1.6032e-03)	
3	UF8	rHV	**0.86481** **(5.4229e-04)**	0.85802 (4.4350e-03)	≈	/	+	/	+	/	+
		IGD	**3.7816e-1** **(6.8207e-03)**	3.8864e-1 (1.6734e-02)		4.4150e+0 (9.3654e-03)		4.4104e+0 (8.4396e-03)		4.4437e+0 (8.0230e-03)	
3	UF9	rHV	0.67394 (6.9510e-04)	**0.67395** **(1.4406e-03)**	≈	/	+	/	+	/	+
		IGD	5.2235e-1 (5.7035e-03)	**5.1574e-1** **(6.5216e-03)**		4.5379e+0 (3.9615e-03)		4.5186e+0 (5.6821e-03)		4.5717e+0 (5.0369e-03)	
3	UF10	rHV	**0.85455** **(3.0118e-03)**	0.85221 (3.7759e-03)	≈	/	+	/	+	/	+
		IGD	**2.6147e-1** **(3.2950e-03)**	2.7096e-1 (4.0091e-03)		2.1152e+1 (1.1566e-02)		2.1286e+1 (5.3960e-02)		2.0163e+1 (5.6312e-02)	

+/-/≈			5/1/4	10/0/0	10/0/0	10/0/0

In Table 2, "/" represent that cannot calculate the result, means "0". It shows that we proposed algorithm has 5 problems better than the others in 2-objective UF test problems, while in 3-objective test problems, has better performance in UF8 and UF10 problems.

Similarly, we can discover from the Wilcoxon signed-rank test, that the proposed WOF-MMOPSO-RDG is significantly better than the other four comparison algorithms on the 5 instances under these 10 test problems, while 4 instances are performed similarly comparing with WOF-MMOPSO. So, we proposed algorithm also can obtain better performance in UF test problems.

4.3.3. The result analysis of the test problems with 1000 variables

As we can see from Section 4.3.1 and Section 4.3.2, by contrast, the performance of MOEA/D-RDG is better than MOEA/DVA and LMEA in WFG test suits, and MOEA/D-RDG cannot deal with most of UF problems well, but all the performance of them are very poor. The reason is they cannot be convergent when we set 100 000 function evaluations. The algorithm we proposed do not determine the independence of the decision variables by the difference method, so it can save much more computing resources and get better results.

To compared with the WOF, the algorithm we proposed has better performance, because the group size is depended on the test problems rather than a fixed number, which means the algorithm can adjust the group size adaptively.

We also do another experiments in this section in order to see whether our algorithm's performance will be better when the number of function evaluations increases. Therefore, we set 100 000 function evaluations to 1 000 000 function evaluations for the proposed algorithm when it is used to test UF problems with 1000 variables. Fig.1 show the experimental result.

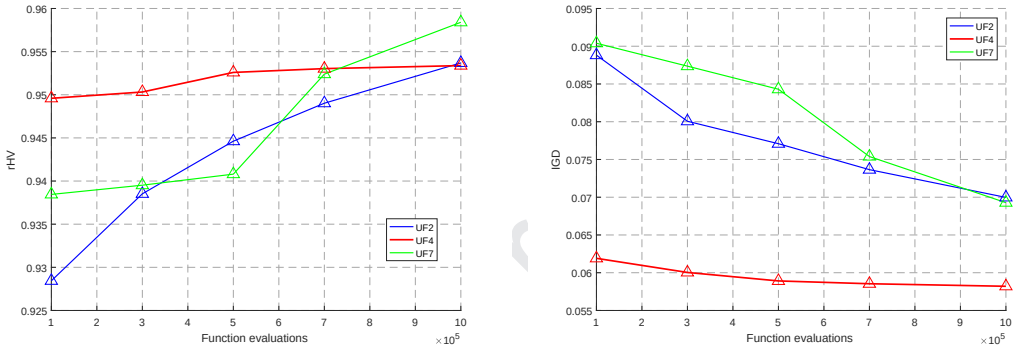

Figure 1: the variation of average value of rHV and IGD with different function evaluations on UF2, UF4 and UF7 test problems

As can be seen from Fig.1, with the increase of evaluation times, the performance of our algorithm is also improved. So, we have reason to believe that at 8 million evaluations, which all of MOEA/DVA, LMEA and MOEA/D-RDG take to convergent, our algorithm is still competitive. This also proves the superiority of our algorithm in saving computing resources.

4.3.4. Experimental Result of problems with 2000 and 5000 variables

In the previous section, we have applied the proposed algorithm to 28 1000-dimensional problems, and our proposed algorithm has very good performance on these problems when compared with other three com-

parative algorithms. In order to further test whether our proposed algorithm can be extended to solve the problems with higher dimensions, we conducted another experiment on 3-objective 2000-dimensional and 5000-dimensional problems with 100 000 function evaluations in this section. Since MOEA/DVA, LMEA and MOEA/D-RDG lose its convergence among most of 1000-dimensional problems, only WOF is selected for comparison. The parameters are consistent with Section 4.3.1, and the experiment results are shown in Table 3.

Table 3: The average and standard deviation value of rHV and IGD obtained by the compared algorithms with 2000 and 5000 variables

		WOF-MMOPSO-RDG		WOF-MMOPSO			
		2000 variables	5000 variables	2000 variables		5000 variables	
WFG1	rHV	**0.99913 (1.0462e-07)**	**0.99897 (7.0882e-07)**	0.99911 (4.3173e-07)	≈	0.99891 (2.5942e-07)	+
	IGD	**2.2301e-1 (1.2070e-04)**	**2.2705e-1 (2.5245e-04)**	2.2524e-1 (6.3756e-04)	≈	2.3558e-1 (5.9856e-04)	+
WFG2	rHV	**0.94042 (1.1996e-06)**	**0.93668 (1.5084e-05)**	0.93898 (1.9483e-04)	≈	0.93504 (2.7760e-05)	+
	IGD	**3.4099e-1 (6.7902e-06)**	**3.4908e-1 (4.8233e-04)**	3.4423e-1 (9.8494e-04)	≈	3.5479e-1 (5.8271e-04)	+
WFG3	rHV	**0.79451 (6.9815e-05)**	**0.78986 (6.1700e-05)**	0.79383 (3.6252e-04)	+	0.78765 (3.3279e-05)	≈
	IGD	**4.7734e-1 (5.7014e-04)**	**4.9079e-1 (5.6374e-04)**	4.8251e-1 (7.1236e-04)	+	5.0083e-1 (2.2533e-04)	≈
WFG4	rHV	0.91801 (1.3053e-04)	0.91693 (1.0709e-05)	**0.91895 (1.1680e-04)**	≈	**0.91793 (1.0010e-05)**	≈
	IGD	3.7280e-1 (1.1943e-04)	3.7593e-1 (1.3663e-04)	**3.6632e-1 (3.1666e-03)**	≈	**3.6873e-1 (9.5630e-05)**	≈
WFG5	rHV	0.93613 (1.9837e-04)	0.93540 (1.5664e-05)	**0.93684 (1.3485e-03)**	≈	**0.93554 (1.7351e-05)**	≈
	IGD	3.2691e-1 (1.6206e-03)	3.3163e-1 (1.3062e-04)	**3.2511e-1 (6.6303e-03)**	≈	**3.3289e-1 (1.7400e-04)**	≈

WFG6	rHV	**0.98822 (9.2662e-04)**	**0.98991 (1.5240e-03)**	0.98764 (3.6441e-04)	≈	0.98908 (1.2231e-03)	+
	IGD	**3.1363e-1 (1.8792e-03)**	**2.7418e-1 (3.0130e-03)**	3.1390e-1 (3.2642e-04)		3.1598e-1 (2.9130e-03)	
WFG7	rHV	**0.89298 (4.4556e-05)**	**0.88789 (1.7693e-05)**	0.89085 (1.4201e-04)	≈	0.88783 (2.4949e-05)	≈
	IGD	**4.4056e-1 (3.7322e-04)**	**4.4956e-1 (2.0220e-04)**	4.4222e-1 (3.9727e-03)		4.5052e-1 (2.3067e-04)	
WFG8	rHV	0.90953 (1.0954e-05)	0.90663 (7.0892e-06)	**0.91141 (1.6204e-05)**	≈	**0.91378 (5.3027e-06)**	-
	IGD	3.9962e-1 (1.2082e-04)	4.0599e-1 (5.6987e-05)	**3.9765e-1 (2.0206e-04)**		**3.8999e-1 (7.8123e-05)**	
WFG9	rHV	**0.94815 (1.5414e-04)**	**0.95140 (4.0212e-03)**	0.94810 (2.2544e-05)	≈	0.93812 (3.0277e-04)	+
	IGD	**3.0570e-1 (6.8152e-04)**	**3.0326e-1 (2.7776e-03)**	3.0834e-1 (1.1060e-04)		3.5350e-1 (2.3324e-03)	
+/-/≈				1/0/8		4/1/4	

As shown in Table 3, in 2000-dimensional problems, there are 6 results of our proposed algorithm are the best out of the 9 problems, that is the same as 5000-dimensional problems. According to the Wilcoxon signed rank test, our algorithm is significantly better than WOF-MMOPSO in 1 of 9 test problems, and there are 4 problems in which they can obtained the solution set with little difference in 2000-dimensional problems, but in 5000-dimensional problems, there are 4 problems significantly better than WOF-MMOPSO.

It can be seen from the results that the algorithm we proposed can maintain good performance. And we also can know that the most of the results are not good as the problems with 1000 dimensions, and there are two main reasons for this: First, with the increase of the dimension, the

computational complexity of algorithm will also increase, which may result in slower convergence, so the performance of the algorithm will go down; Second, the function evaluations are just 100 000 for fair comparison, and the results can be improved with the increase of the evaluations. So, it is obvious that our algorithm can obtain satisfactory results on higher dimension problems, which up to 5000, with fewer function evaluation times, which just are 100 000.

Fig.2 also shows the variation of average value of rHV and IGD of WFG1, WFG2, WFG8 and WFG9 problems from 1000 to 5000 dimension.

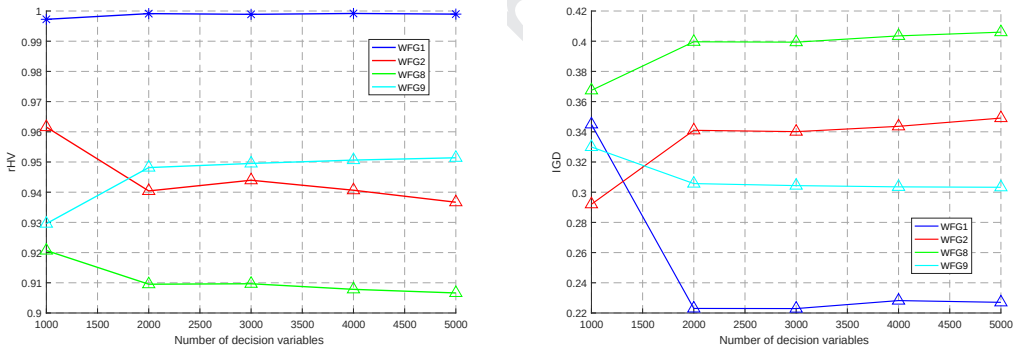

Figure 2: The variation of average value of rHV and IGD with different number of decision variables on WFG1, WFG2, WFG8 and WFG9 test problems

In Fig.2, we can see that the performance of the proposed algorithm on some problems, such as WFG1 and WFG9, can be improved a litter, but for WFG2 and WFG8, its performance would be reduced, but not much. However, in general, the performance of our algorithm on most high-dimensional problems is still competitive, and there are still many deficiencies that need to be further improved.

5. Conclusion

In this paper, we proposed a random dynamic grouping based weight optimization framework for solving large-scale multi-objective problems. For testing the performance of our algorithm, some test benchmark functions are adopted, i.e. UF test problems and WFG test problems, and some comparing algorithms are selected, i.e. MOEA/DVA, LMEA, WOF and MOEA/D-RDG. For 1000-dimensional test problems, our algorithm can get the ideal results and also can save the computing resource when compared with MOEA/DVA, LMEA and MOEA/D-RDG, while compared with WOF, our algorithm also can get better results on the most problems with the same function evaluations. In the end, we try to use our algorithm to solve 2000-dimensional even 5000-dimensional problems, and the good news is that our algorithm also can get a solution set which can convergence to the true PF. It is also worth to notice the way that you choose the q solutions, which represent the entire population optimized by MMOPSO at the beginning of the algorithm, will directly affect the outcome. Since q is a very small number, and the way that we used in this paper is Crowding Distance method, the boundary points are always selected through this way. It does nothing to further increase the diversity of the final solutions.

As mentioned above, exploring a better way to select the q solutions is an interesting direction in the future. If there is a way that can choose some more diverse solutions instead of just the boundary points, the population can be better represented which is more advantageous to the next step, that is weight optimization, and the final solution set will be more satis-

fying, and then the proposed algorithm can be extended to more complex test problems, like [52]. Another point of the future work is parallelization, because the q solutions are independent, each of them can optimize on its own, so that the calculation time will be further reduced.

6. Acknowledgments

This work was supported by the National Natural Science Foundation of China (Nos. 61876141 and 61373111), and the Provincial Natural Science Foundation of Shaanxi of China (No.2019JZ-26).

References

[1] Z. Wang, Q. Zhang, A. Zhou, M. Gong, L. Jiao, Adaptive replacement strategies for MOEA/D, IEEE transactions on cybernetics 46 (2) (2015) 474–486.

[2] R. Cheng, C. He, Y. Jin, X. Yao, Model-based evolutionary algorithms: a short survey, Complex & Intelligent Systems 4 (4) (2018) 283–292.

[3] Y. Tian, R. Cheng, X. Zhang, F. Cheng, Y. Jin, An indicator-based multiobjective evolutionary algorithm with reference point adaptation for better versatility, IEEE Transactions on Evolutionary Computation 22 (4) (2017) 609–622.

[4] X. Ma, Q. Zhang, G. Tian, J. Yang, Z. Zhu, On tchebycheff decomposition approaches for multiobjective evolutionary optimization, IEEE Transactions on Evolutionary Computation 22 (2) (2017) 226–244.

[5] Y. Tian, C. He, R. Cheng, X. Zhang, A multistage evolutionary algorithm for better diversity preservation in multiobjective optimization, IEEE Transactions on Systems, Man, and Cybernetics: Systems (2019) 1–15.

[6] J. Del Ser, E. Osaba, D. Molina, X.-S. Yang, S. Salcedo-Sanz, D. Camacho, S. Das, P. Suganthan, C. Coello, F. Herrera, Bio-inspired computation: Where we stand and what's next, Swarm and Evolutionary Computation 48 (2019) 220 – 250.

[7] M. A. Potter, K. A. D. Jong, A cooperative coevolutionary approach to function optimization, Third Parallel Problem Solving Form Nature 866 (1994) 249–257.

[8] X. Ma, L. Fang, Y. Qi, X. Wang, L. Li, L. Jiao, M. Yin, M. Gong, A multiobjective evolutionary algorithm based on decision variable analyses for multiobjective optimization problems with large-scale variables, IEEE Transactions on Evolutionary Computation 20 (2) (2016) 275–298.

[9] X. Zhang, Y. Tian, R. Cheng, Y. Jin, A decision variable clustering based evolutionary algorithm for large-scale many-objective optimization, IEEE Transactions on Evolutionary Computation 22 (1) (2018) 97–112.

[10] H. Zille, H. Ishibuchi, S. Mostaghim, Y. Nojima, Weighted optimization framework for large-scale multi-objective optimization, in: Pro-

ceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion, 2016, pp. 83–84.

[11] H. Zille, H. Ishibuchi, S. Mostaghim, Y. Nojima, A framework for large-scale multi-objective optimization based on problem transformation, IEEE Transactions on Evolutionary Computation 22 (2) (2018) 260–275.

[12] W. Hong, K. Tang, A. Zhou, H. Ishibuchi, X. Yao, A scalable indicator-based evolutionary algorithm for large-scale multiobjective optimization, IEEE Transactions on Evolutionary Computation 23 (3) (2018) 525–537.

[13] X. Li, Y. Mei, X. Yao, M. N. Omidvar, Cooperative co-evolution with differential grouping for large scale optimization, IEEE Transactions on Evolutionary Computation 18 (3) (2014) 378–393.

[14] Y. Sun, M. Kirley, S. K. Halgamuge, Extended differential grouping for large scale global optimization with direct and indirect variable interactions, Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation. ACM (2015) 313–320.

[15] M. N. Omidvar, X. Li, X. Yao, Cooperative co-evolution with delta grouping for large scale non-separable function optimization, in: IEEE Congress on Evolutionary Computation, IEEE, 2010, pp. 1–8.

[16] Z. Yang, T. Ke, Y. Xin, Large scale evolutionary optimization using cooperative coevolution, Information Sciences 178 (15) (2014) 2985–2999.

[17] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Transactions on Evolutionary Computation 6 (2) (2002) 182–197.

[18] A. J. Nebro, J. J. Durillo, J. Garcia-Nieto, C. C. Coello, F. Luna, E. Alba, SMPSO: A new pso-based metaheuristic for multi-objective optimization, in: 2009 IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making (MCDM), IEEE, 2009, pp. 66–73.

[19] A. Song, Q. Yang, W. Chen, J. Zhang, A random-based dynamic grouping strategy for large scale multi-objective optimization, in: 2016 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2016, pp. 468–475.

[20] Q. Lin, J. Li, Z. Du, J. Chen, M. Zhong, A novel multi-objective particle swarm optimization with multiple search strategies, European Journal of Operational Research 247 (3) (2015) 732–744.

[21] Y. Guo, P. Zhang, J. Cheng, C. Wang, D. Gong, Interval multi-objective quantum-inspired cultural algorithms, Neural Computing and Applications 30 (3) (2018) 709–722.

[22] X. Ma, Q. Zhang, G. Tian, J. Yang, Z. Zhu, On tchebycheff decomposition approaches for multiobjective evolutionary optimization, IEEE Transactions on Evolutionary Computation 22 (2) (2017) 226–244.

[23] F. Wang, H. Zhang, Y. Li, Y. Zhao, Q. Rao, External archive matching strategy for MOEA/D, Soft Computing 22 (23) (2018) 7833–7846.

[24] G. Wu, R. Mallipeddi, P. Suganthan, Ensemble strategies for population-based optimization algorithms – a survey, Swarm and Evolutionary Computation 44 (2019) 695 – 711.

[25] S. Wang, M. Gong, Y. Wu, M. Zhang, Multi-objective optimization for location-based and preferences-aware recommendation, Information Sciences 513 (2020) 614–626.

[26] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: Improving the strength pareto evolutionary algorithm, in: Evolutionary Methods for Design, Optimisation and Control, Barcelona, Spain: CIMNE, 2002, pp. 95–100.

[27] J. Bader, E. Zitzler, HypE: an algorithm for fast hypervolume-based many-objective optimization, Evolutionary Computation 19 (1) (2011) 45–76.

[28] L. M. Russo, A. P. Francisco, Quick hypervolume, IEEE Transactions on Evolutionary Computation 18 (4) (2013) 481–502.

[29] Q. Zhang, A. Zhou, Y. Jin, RM-MEDA: A regularity model-based multiobjective estimation of distribution algorithm, IEEE Transactions on Evolutionary Computation 12 (1) (2008) 41–63.

[30] E. Zitzler, S. Künzli, Indicator-based selection in multiobjective search, Lecture Notes in Computer Science 3242 (2004) 832–842.

[31] D. Brockhoff, E. Zitzler, Improving hypervolume-based multiobjective evolutionary algorithms by using objective reduction methods,

in: 2007 IEEE congress on evolutionary computation, IEEE, 2007, pp. 2086–2093.

[32] Q. Zhang, L. Hui, MOEA/D: A multiobjective evolutionary algorithm based on decomposition, IEEE Transactions on Evolutionary Computation 11 (6) (2007) 712–731.

[33] C. Dai, Y. Wang, M. Ye, A new multi-objective particle swarm optimization algorithm based on decomposition, Information Sciences 325 (2015) 541–557.

[34] S. Mahdavi, M. E. Shiri, S. Rahnamayan, Metaheuristics in large-scale global continues optimization: A survey, Information Sciences 295 (2015) 407–428.

[35] R. Cheng, Y. Jin, A competitive swarm optimizer for large scale optimization, IEEE Transactions on Cybernetics 45 (2) (2015) 191–204.

[36] M. A. Potter, K. A. De Jong, Cooperative coevolution: an architecture for evolving coadapted subcomponents, Evolutionary Computation 8 (1) (2014) 1–29.

[37] X. Li, X. Yao, Tackling high dimensional nonseparable optimization problems by cooperatively coevolving particle swarms, in: 2009 IEEE Congress on Evolutionary Computation, IEEE, 2009, pp. 1546–1553.

[38] X. Li, X. Yao, X. Li, X. Yao, Cooperatively coevolving particle swarms for large scale optimization, IEEE Transactions on Evolutionary Computation 16 (2) (2012) 210–224.

[39] X. Ma, X. Li, Q. Zhang, K. Tang, Z. Liang, W. Xie, Z. Zhu, A survey on cooperative co-evolutionary algorithms, IEEE Transactions on Evolutionary Computation 23 (3) (2018) 421–441.

[40] L. M. Antonio, C. A. C. Coello, Use of cooperative coevolution for solving large scale multiobjective optimization problems, in: 2013 IEEE Congress on Evolutionary Computation, IEEE, 2013, pp. 2758–2765.

[41] E. Zitzler, K. Deb, L. Thiele, Comparison of multiobjective evolutionary algorithms: Empirical results, Evolutionary computation 8 (2) (2000) 173–195.

[42] S. Kukkonen, J. Lampinen, Performance assessment of generalized differential evolution 3 (GDE3) with a given set of problems, in: 2007 IEEE Congress on Evolutionary Computation, IEEE, 2007, pp. 3593–3600.

[43] S. Huband, P. Hingston, L. Barone, R. L. While, A review of multiobjective test problems and a scalable test problem toolkit, IEEE Transactions on Evolutionary Computation 10 (5) (2006) 477–506.

[44] K. Deb, L. Thiele, M. Laumanns, E. Zitzler, Scalable multi-objective optimization test problems, in: Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600), Vol. 1, 2002, pp. 825–830.

[45] A. W. Iorio, X. Li, A cooperative coevolutionary multiobjective al-

gorithm using non-dominated sorting, in: Genetic and Evolutionary Computation Conference, Springer, 2004, pp. 537–548.

[46] W. Chen, T. Weise, Z. Yang, K. Tang, Large-scale global optimization using cooperative coevolution with variable interaction learning, in: International Conference on Parallel Problem Solving from Nature, Springer, 2010, pp. 300–309.

[47] M. N. Omidvar, Y. Mei, X. Li, Effective decomposition of large-scale separable continuous functions for cooperative co-evolutionary algorithms, in: 2014 IEEE congress on evolutionary computation (CEC), IEEE, 2014, pp. 1305–1312.

[48] J. E. Fieldsend, R. M. Everson, S. Singh, Using unconstrained elite archives for multiobjective optimization, IEEE Transactions on Evolutionary Computation 7 (2003) 305–323.

[49] Q. Zhang, A. Zhou, S. Zhao, P. Suganthan, W. Liu, S. Tiwari, Multiobjective optimization test instances for the CEC 2009 special session and competition, University of Essex, Colchester, UK and Nanyang Technological University (2008) 1–30.

[50] F. Wilcoxon, S. Katti, R. A. Wilcox, Critical values and probability levels for the Wilcoxon rank sum test and the Wilcoxon signed rank test, Selected tables in mathematical statistics 1 (1970) 171–259.

[51] A. Zhou, Y. Jin, Q. Zhang, B. Sendhoff, E. Tsang, Combining model-based and genetics-based offspring generation for multi-objective op-

timization using a convergence criterion, in: 2006 IEEE International Conference on Evolutionary Computation, IEEE, 2006, pp. 892–899.

[52] H. Li, K. Deb, Q. Zhang, P. Suganthan, L. Chen, Comparison between moea/d and nsga-iii on a set of many and multi-objective benchmark problems with challenging difficulties, Swarm and Evolutionary Computation 46 (2019) 104 – 117.

Declaration of interests

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: