IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 26, 2019, accepted December 15, 2019, date of publication December 18, 2019,
date of current version December 31, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2960531

An Improved NSGA-III Algorithm Using Genetic K-Means Clustering Algorithm

QINGGUO LIU, XINXUE LIU, JIAN WU, AND YAXIONG LI
Xi'an High-Tech Institute, Xi'an 710025, China

Corresponding author: Qingguo Liu (teamalpha@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61603398, and in part by the Natural Science Basic Research Plan in Shaanxi Province of China under Grant 2018JM6007.

ABSTRACT The non-dominated sorting genetic algorithm III (NSGA-III) has recently been proposed to solve many-objective optimization problems (MaOPs). While this algorithm achieves good diversity, its convergence is unsatisfactory. In order to improve the convergence, we propose an improved NSGA-III using a genetic K-means clustering algorithm (NSGA-III-GKM), which can also ensure diversity and automatically provide the number and direction vector of the subspaces. Compared with the NSGA-III, the proposed NSGA-III-GKM has two key features. First, the initial reference points are clustered using a GKM clustering algorithm, which realizes automatic learning of the number of clusters. Second, as the reference points are replaced by cluster centers, a penalty-based boundary intersection (PBI) aggregation function is introduced to replace the perpendicular distance. The proposed NSGA-III-GKM and other similar optimization algorithms (NSGA-III, MOEA/D, U-NSGA-III, DC-NSGA-III and B-NSGA-III) are tested on DTLZ test problems and UF test problems. The simulation results demonstrate that the NSGA-III-GKM exhibits better diversity and convergence performance than the other algorithms.

INDEX TERMS Many-objective optimization, genetic K-means clustering algorithm, NSGA-III, automatic learning.

I. INTRODUCTION

Optimization problems with more than three objectives are called many-objective optimization problems (MaOPs) [1], [2]. These problems frequently appear in many research fields, and are typically solved by a special class of evolutionary algorithms called many-objective evolutionary algorithms (MOEAs) [3]–[5], especially the basic non-dominated sorting genetic algorithm (NSGA) and its variants. This basic NSGA proposed by Srinivas and Deb [6] has been widely used to solve MaOPs, but it has a high computational complexity. Therefore, Deb et al. [7] proposed the NSGA-II to reduce the computational complexity through an elite reserved strategy that is based on a quick sorting of non-dominated solutions. To solve MaOPs, the NSGA-II can obtain many non-dominated solutions, but its evolutionary pressure decreases. Therefore, Deb and Jain [8] proposed the NSGA-III, in which the crowded distance of the NSGA-II is replaced by reference points. The performance of the NSGA-III is better than that of the NSGA-II. Indeed, the NSGA-III is currently widely recognised as the best available algorithm for MaOPs [9]–[11]. However, the NSGA-III convergence is still not totally satisfactory due to its exclusive consideration of solutions that are closest to the reference points in the niche-preservation operation phase [12].

Many improved NSGA-III variants have been proposed in the literature. Yuan *et al.* [13] introduced the θ-NSGA-III, which exploits a θ-dominance relation and a penalty-based boundary intersection (PBI) aggregation function. While the θ-NSGA-III framework has diverged significantly from that of the NSGA-III, the θ-NSGA-III typically outperforms the NSGA-III. Seada and Deb [14] proposed a unified NSGA-III called the U-NSGA-III. The U-NSGA-III maintains a constant preference of convergence over diversity. Abouhawwash *et al.* [15] integrated the Karush Kuhn Tucker proximity measure (KKTPM) with the NSGA-III to enhance its convergence properties towards the true Pareto optimal front. Seada *et al.* [16] focused on the diversity and convergence of the NSGA-III (DC-NSGA-III); hence, a local search and KKTPM were used to improve the performance

The associate editor coordinating the review of this manuscript and approving it for publication was Hongwei Du.

of the NSGA-III. Based on the U-NSGA-III and KKTPM, Seada *et al.* [17] proposed a multi-phased NSGA-III capable of automatically balancing convergence and the diversity of the population members, which is called the B-NSGA-III. To solve MaOPs, an alternative to the NSGA-III is the framework based on the decomposition strategy (MOEA/D) [18]. The weaknesses of MOEA/D are that the number and direction vectors of the subspaces are hard to determine and its convergence is not good. Based on the framework of the NSGA-III and the idea of MOEA/D, we propose a new algorithm that combines the genetic K-means clustering algorithm [19]–[21] and NSGA-III to separate the objective space. Our proposed algorithm, named the NSGA-III-GKM, adopts PBI aggregation functions to replace the perpendicular distances. The advantages of the NSGA-III-GKM are as follows: (a) The GKM clustering algorithm strengthens the capacity of developing and exploring the whole objective space, and hence improves the convergence; (b) the introduction of the PBI aggregation function in the niche-preservation operation phase improves the convergence; (c) genetic operations are executed independently in subspaces, which can ensure diversity; (d) the GKM clustering algorithm uses non-subjective reference point data to automatically determine the number and direction vectors of the subspaces.

This paper is organized as follows. First, we briefly review the NSGA-III framework in Section 2. Then, we propose the NSGA-III-GKM and provide its details in Section 3. Finally, in Section 4, we conduct simulations to compare the performance of the NSGA-III-GKM, NSGA-III, MOEA/D, U-NSGA-III, DC-NSGA-III and B-NSGA-III on DTLZ test problems and UF test problems. The simulation results demonstrate that the NSGA-III-GKM has better diversity and convergence performance than the other algorithms.

II. BRIEF REVIEW OF THE NSGA-III FRAMEWORK

The NSGA-III framework is similar to that of the NSGA-II. Let us assume that an MaOP has M objectives. In the t^{th} generation, let the size of the parent population P_t be N and the size of its offspring population Q_t, which is obtained through selection, crossover and mutation operations, be N. The population members are the points that express the object values of the solutions. If two points x_A and x_B satisfy the following conditions, x_A is the non-dominated point [22].

$$\begin{cases} \forall i \in \{1, 2, \cdots, M\}, & f_i(x_A) \leq f_i(x_B) \\ \exists j \in \{1, 2, \cdots, M\}, & f_j(x_A) < f_j(x_B) \end{cases} \tag{1}$$

where $f_i(x_A)$ and $f_i(x_B)$ are the i^{th} objective values of x_A and x_B, respectively; $f_j(x_A)$ and $f_j(x_B)$ are the j^{th} objective values of x_A and x_B, respectively.

The NSGA-III obtains the $t+1^{\text{th}}$ generation by combining the parent and offspring populations, i.e., $R_t = P_t \cup Q_t$ where the size of R_t is $2N$. According to the non-dominated sorting rules, R_t is then divided into different levels, denoted by F_1, $F_2, \ldots$, etc. Starting from F_1, each level is selected one at a time to construct a new population S_t, and the size of S_t is equal to or larger than N for the first time. If the last level included is the l^{th} level, solutions in S_t/F_l (levels before F_l) are chosen for the next parent population P_{t+1} while solutions in the remaining levels are rejected. The key implementation steps of the NSGA-III are as follows.

Step 1 (Normalization of the Objective Values): The objective values of the population members are normalized using the ideal and extreme points. In a population S_t, use the minimum values of all objectives to construct the ideal point [23] $\boldsymbol{z}^{\min} = (z_1^{\min}, z_2^{\min}, \cdots, z_M^{\min})$. The objective value of each solution is translated by subtracting the ideal point $z^{\min}$,

$$f_i'(\boldsymbol{x}_j) = f_i(\boldsymbol{x}_j) - \boldsymbol{z}^{\min} \tag{2}$$

where $i = 1, 2, \cdots, M, j \geq N, \boldsymbol{x}_j$ is the j^{th} solution, and $f_i(\boldsymbol{x}_j)$ is the i^{th} objective value of the j^{th} solution $\boldsymbol{x}_j$.

The extreme point is identified by finding the solution that minimizes the following achievement scalarization function (ASF) with the weight vector $\boldsymbol{w}$ [24]:

$$\min\ ASF(\boldsymbol{x}_j, \boldsymbol{w}_i) = \max_{i=1}^{M} f_i'(\boldsymbol{x}_j)/\boldsymbol{w}_i \tag{3}$$

where $\boldsymbol{w} = (w_1, w_2, \cdots, w_M)$ is a weight vector. For finding the k^{th} extreme point, we set $w_k = 1$, while the other weights are set to a small value, e.g., 10^{-6}. We use M extreme points to obtain an M dimensional linear hyperplane. The $f_i'(\boldsymbol{x}_j)$ objective can be normalized as

$$f_i^n(\boldsymbol{x}_j) = \frac{f_i'(\boldsymbol{x}_j)}{a_i - z_i^{\min}} \tag{4}$$

where a_i is the intercept of the i^{th} objective axis.

Step 2 (Generation of the Initial Reference Points): The initial reference points are commonly generated on a normalized hyperplane using Das and Dennis's systematic approach [25]. For M objectives and p divisions of each objective, the total number H of reference points is

$$H = \binom{p + M - 1}{p} \tag{5}$$

Step 3 (Perpendicular Distance Computation): After normalizing the objective values and generating reference points, the perpendicular distance between the objective value of each solution and a reference line (joining the origin with a reference point) is computed. For a population S_t, a solution is associated with the reference point of the minimum perpendicular distance.

Step 4 (Niche-Preservation Operation): The niche count ρ_i is equal to the number of solutions in S_t/F_l, associated with the i^{th} reference point. The minimum niche count is $J_{\min} = \min \rho_i, i = 1, 2, \cdots, M$. The reference point with $J_{\min}$ is chosen. If $J_{\min} > 1$, one reference point is chosen randomly. We set the chosen reference point as the l^{th} reference point.

If $\rho_l \geq 1$ and the l^{th} reference point is associated with one or more solutions in F_l, a solution in F_l is randomly selected into population P_{t+1}, and the value of ρ_i is incremented by one. If $\rho_l \geq 1$ and no solution in F_l is associated

with the l^{th} reference point, this reference point is not considered in the t^{th} generation.

If $\rho_i = 0$ and the l^{th} reference point is associated with one or more solutions in F_l, the solution with the minimum perpendicular distance is selected into population P_{t+1} and the value of ρ_i is incremented by one. If $\rho_i = 0$ and no solution in F_l is associated with the l^{th} reference point, this reference point is not considered in the t^{th} generation.

Step 5 (Genetic Operations): Genetic operations include selection [26], simulated binary crossover (SBX) [27] and polynomial mutation [28]. After the parent population P_{t+1} is obtained, the offspring population Q_{t+1} is obtained by these genetic operations.

III. THE PROPOSED NSGA-III-GKM

The GKM clustering algorithm is combined with the NSGA-III to obtain the cluster centers of the initial reference points and replace these points by cluster centers. Specifically, after clustering the reference points with the GKM clustering algorithm, the objective space is partitioned into several subspaces $\{S_1, S_2, \cdots, S_k\}$, where k is the cluster count. Each cluster center is the direction vector of one subspace. Objective space partitioning can improve the convergence by overcoming the problem only solutions closest to the reference points are considered in the niche-preservation operation phase. The PBI aggregation function [29] is used to replace the perpendicular distance and hence further improve the convergence of the NSGA-III. In this paper, the performance of the NSGA-III-GKM and other state-of-the-art algorithms is evaluated by the inverted generation distance (IGD) indicator [30], [31] and hypervolume (HV) indicator [32].

A. ALGORITHMIC DETAILS OF THE NSGA-III-GKM

The NSGA-III-GKM algorithm is as follows:

Step 1 (Initialization): Set the size N of the population P_t, and the maximum number TM of functions evaluations (FEs).

Step 2 (Generation of the Populations): Randomly generate the initial population based on the aforementioned genetic operations of Section 2. Note that the genetic operations are executed for each individual subspace. Then perform non-dominated sorting of the populations $P_t = \sum_{i=1}^{k} P_t^{S_i}$ and $Q_t = \sum_{i=1}^{k} Q_t^{S_i}$. $P_t^{S_i}$ is the size of the parent population in the subspace S_i, $Q_t^{S_i}$ is the size of the offspring population in the subspace S_i, and $P_t^{S_i}$ is equal to $Q_t^{S_i}$. Combine the parent and offspring populations, i.e., $R_t = P_t \cup Q_t$.

Step 3 (Normalization): This normalization step is the same as the one for the basic NSGA-III scheme (see Section 2).

Step 4: (Clustering of the Reference Points): The goal of clustering the reference points is to partition the objective space. The K-means algorithm is sensitive to the initial cluster centers, unable to determine the optimal number of clusters, and easily trapped into local optima. Genetic algorithms can overcome the shortcomings of the K-means clustering algorithm and realize automatic learning of the cluster count. After the reference point generation following the basic NSGA-III scheme (Section 2), the points are clustered by the GKM clustering algorithm. First, we review the conventional K-means clustering algorithm as follows. k reference points are randomly selected as the initial cluster centers. Then, each of the remaining reference points is assigned to the cluster center of the nearest distance. The cluster centers are recalculated to minimize the squared error criterion E during convergence.

$$E = \sum_{j=1}^{k} \sum_{\boldsymbol{p} \in C_j} \left\| \boldsymbol{p} - \boldsymbol{m}_j \right\|^2 \tag{6}$$

where k is the number of clusters, $\boldsymbol{p}$ is the solution, C_j is the j^{th} cluster and $\boldsymbol{m}_j$ is the cluster center of cluster C_j.

The genetic variant of the K-means clustering algorithm involves the evolution of chromosomes, and the final result is obtained by genetic operations such as selection, crossover, and mutation. The steps of the GKM clustering algorithm are as follows:

a) Real-number encoding is adopted to transform the cluster centers into genes $G_1, G_2, \cdots, G_k$ on a chromosome, as shown in Fig. 1 The size of a chromosome varies with the number of clusters.

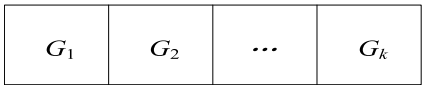

FIGURE 1. Encoding of cluster centers as genes of a chromosome.

b) The clustering results should satisfy tightness and separability requirements. Tightness means that the reference points within one cluster are as similar as possible, while separability means that the reference points in different clusters are as different as possible. Therefore, we define the fitness function *fitD* of the GKM clustering algorithm as

$$fitD = \frac{G_b}{b + aE} \tag{7}$$

where a and b are positive constant coefficients, and G_b is the sum of the distances between different clusters:

$$G_b = \frac{2}{k\,(k-1)} \sum_{i=1}^{k} \sum_{j=i+1}^{k} \left\| \boldsymbol{m}_i - \boldsymbol{m}_j \right\|^2 \tag{8}$$

which is used to quantify the separability. The fitness function *fitD* indicates that the clustering is better when the reference points in the same cluster are closer to each other (the value of E is smaller) and the cluster centers of different clusters are farther from each other (the value of G_b is larger).

Using genetic operations including roulette-based selection and single-point crossover, we propose a new mutation operation that leads to automatic learning of the optimal number of clusters k. The population chromosome with the largest fitness value is selected as the model chromosome of

the optimal cluster number. Other chromosomes in the same population should learn from this model to achieve better fitness by decreasing and increasing the genes of chromosomes. The chromosomes in the initial population have the same length, a small number of initial clusters is set, and an increasing trend is assumed when the first mutation occurs. With the reoccurrence of mutation operation, the offspring chromosomes decrease or increase based on whether their lengths are longer or shorter than the model chromosome, respectively. On one hand, decreasing the number of genes is achieved by eliminating the nearest genes to cluster centers of the model chromosome. On the other hand, the genes are increased by adding the farthest reference point to the cluster centers of the model chromosome.

After the cluster centers are determined, solutions in R_t are assigned to clusters based on a criterion of the minimum Euclidean distance. The crossover and mutation probabilities of GKM are denoted as P^c_{GKM} and P^m_{GKM}, respectively. The population and the maximum number of iterations are denoted by P_{GKM} and TM_{GKM}, respectively.

Step 5 (Niche-Preservation Operation Based on the PBI Aggregation Function): Based on the niche-preservation operation (Section 2), solutions with small PBI aggregation function values are added to the next generation until the target population size is attained. The PBI aggregation function value is

$$d(\boldsymbol{x}_j) = d_{i,1}(\boldsymbol{x}_j) + \theta d_{i,2}(\boldsymbol{x}_j) \tag{9}$$

where $\boldsymbol{x}_j$ is the j^{th} solution, $d_{i,1}(\boldsymbol{x}_j)$ and $d_{i,2}(\boldsymbol{x}_j)$ are the projection and vertical distances of $\boldsymbol{x}_j$ in the i^{th} cluster center direction, respectively, and θ is a penalty parameter. A smaller penalty parameter θ is beneficial for selecting solutions with strong convergence. The distances $d_{i,1}(\boldsymbol{x}_j)$ and $d_{i,2}(\boldsymbol{x}_j)$ are defined as, respectively

$$d_{i,1}(\boldsymbol{x}_j) = \left\| (f^n(\boldsymbol{x}_j))^T \boldsymbol{\lambda}_i \right\| / \|\boldsymbol{\lambda}_i\| \tag{10}$$

$$d_{i,2}(\boldsymbol{x}_j) = \left\| f^n(\boldsymbol{x}_j) - d_{i,1}(\boldsymbol{x}_j)(\boldsymbol{\lambda}_i / \|\boldsymbol{\lambda}_i\|) \right\| \tag{11}$$

where $\boldsymbol{\lambda}_i$ is the direction vector from the ideal point to the i^{th} cluster center.

Step 6 (Terminal Conditions): If the maximum number of FE TM is reached, output the current solutions and terminate the program. Otherwise, repeat steps 2-6.

Figs. 2 and 3 illustrate the differences between the proposed NSGA-III-GKM and the basic NSGA-III.

Fig. 2 shows reference points on a normalized hyperplane with $M = 3$. After applying the GKM clustering algorithm, the reference points are assigned to clusters whose centers are taken as new reference points (as shown in Fig. 3). For example, the solution A in cluster 1 has a smaller value of the PBI aggregation function (with one reference point in cluster 2) than solution B in cluster 2 (under the condition that solution B is the only solution in cluster 2). Thus, the solution A will be selected into the next generation according to the NSGA-III rules. However, this is not good for searching in cluster 2 and leads to a break of the search equilibrium. If we use the GKM clustering algorithm, the solution B will be selected into the next generation because this solution is in cluster 2 and has a higher priority. This example shows that the proposed NSGA-III-GKM can overcome the problem in which only the solutions closest to the reference points are selected into the next generation. Indeed, the NSGA-III-GKM strengthens the capacity of developing and exploring the whole objective space and hence leads to improve convergence. In addition, the genetic operations are executed in each subspace independently, which can ensure diversity.

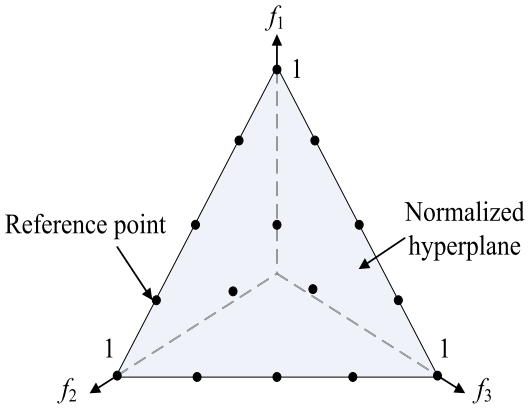

FIGURE 2. Reference points on a normalized hyperplane with $M = 3$.

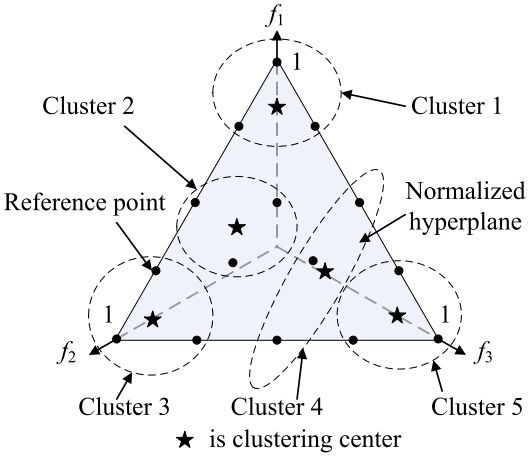

FIGURE 3. Reference points are assigned to clusters based on the GKM clustering algorithm.

B. INDICTORS

The NSGA-III-GKM, NSGA-III, MOEA/D, U-NSGA-III, DC-NSGA-III and B-NSGA-III are tested on DTLZ test problems and UF test problems. The IGD indicator and HV indicator are used to evaluate the performance, including the diversity and convergence of each of these multi-objective evolutionary algorithms. Let P denote the Pareto front (PF) obtained by these algorithms, P^* be the true PF, and $z = (z_1, z_2, \cdots, z_M)^T$ be a reference point in the objective space that is dominated by all Pareto-optimal points.

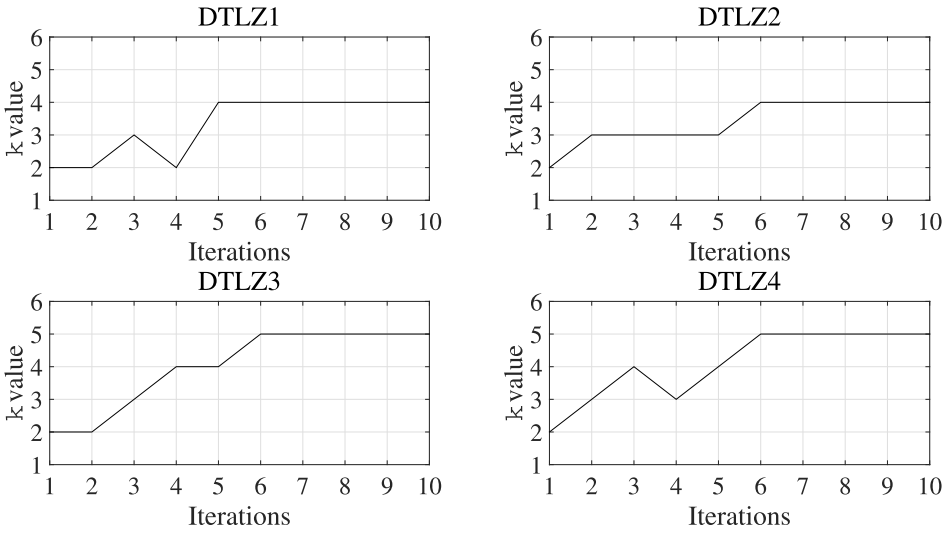

FIGURE 4. Changing curves of k value in the tenth experiment of each of the DTLZ1-4 problems.

The IGD indicator is defined as

$$IGD(P, P^*) = \frac{\sum_{v \in P^*} d(v, P)}{|P^*|} \tag{12}$$

where $d(v, P)$ is the minimum Euclidean distance between a solution v that belongs to P^* and P, while $|P^*|$ is the size of P^*. The IGD value will only be small when both the convergence and diversity of the solutions in P are good. Therefore, the smaller the IGD value is, the better the overall performance of an evolutionary algorithm is.

The HV indicator is defined as

$$HV(P, z) = Volume\left(\bigcup_{F \in P} [f_1, z_1] \times \cdots \times [f_M, z_M]\right) \tag{13}$$

The larger the HV value is, the better the overall performance of an evolutionary algorithm is.

IV. SIMULATIONS AND ANALYSIS OF THE RESULTS

A. SIMULATIONS

In this paper, the DTLZ test problems with 3 to 10 objectives include DTLZ1, DTLZ2, DTLZ3 and DTLZ4 problems, and UF test problems include UF1, UF2, UF3 and UF4 problems. These test problems are used to test the performance of the NSGA-III-GKM, NSGA-III, MOEA/D, U-NSGA-III, DC-NSGA-III and B-NSGA-III evolutionary algorithms. A total of 30 independent runs are performed for each algorithm. The parameters of these five algorithms are as follows. The population size and the maximum number TM of FE are set in Table 1. The genetic crossover and mutation probabilities are set as $P_c = 0.85$ and $P_m = 0.1$, respectively. The penalty parameter θ of the NSGA-III-GKM is set to 5. The other parameters related to the MOEA/D, U-NSGA-III, DC-NSGA-III and B-NSGA-III are adopted from references [14], [16]–[18].

The parameters of the GKM clustering algorithm are as follows: the crossover probability P^c_{GKM} is 0.85, the mutation probability P^m_{GKM} is 0.1, the population P_{GKM} is 5, the maximum number of iterations TM_{GKM} is 10, the initial value of k is 2, and the positive coefficients a and b are 2 and 1.2, respectively.

TABLE 1. Population size and FES.

Problem	Population size	Total FE
DTLZ1(3)	91	40000
DTLZ2(3)	91	30000
DTLZ3(3)	91	90000
DTLZ4(3)	91	60000
DTLZ1(5)	210	120000
DTLZ2(5)	210	70000
DTLZ2(5)	210	210000
DTLZ4(5)	210	210000
DTLZ1(8)	156	120000
DTLZ2(8)	156	80000
DTLZ3(8)	156	160000
DTLZ4(8)	156	200000
DTLZ1(10)	275	280000
DTLZ2(10)	275	210000
DTLZ3(10)	275	400000
DTLZ4(10)	275	550000
UF1	200	300000
UF2	200	300000
UF3	200	300000
UF4	200	300000

The parameters in parentheses are the count number of objectives.

TABLE 2. Average and standard deviation of IGD values obtained by six algorithms on the DTLZ and UF test problems.

Problem		NSGA-III-GKM	NSGA-III	MOEA/D	U-NSGA-III	DC-NSGA-III	B-NSGA-III
DTLZ1(3)	AVG	**6.333e-4[0]**	1.409e-3[4]	1.261e-3[2]	1.499e-3[5]	1.335e-3[3]	7.225e-4[1]
	SD	5.002e-5	2.895e-4	2.669e-4	4.222e-4	6.001e-4	3.012e-5
DTLZ1(5)	AVG	4.169e-4[1]	1.225e-3[5]	6.018e-4[2]	1.200e-3[4]	1.101e-3[3]	**3.332e-4[0]**
	SD	4.020e-5	2.669e-4	3.405e-5	3.255e-4	4.890e-4	3.658e-5
DTLZ1(8)	AVG	**1.292e-3[0]**	2.240e-3[4]	4.251e-3[5]	1.987e-3[3]	1.850e-3[2]	1.420e-3[1]
	SD	4.589e-4	5.023e-4	3.998e-4	2.658e-4	2.332e-4	1.020e-4
DTLZ1(10)	AVG	**1.443e-3[0]**	3.652e-3[4]	6.201e-3[5]	3.425e-3[3]	2.100e-3[2]	1.792e-3[1]
	SD	1.556e-4	2.002e-4	1.256e-4	1.687e-4	2.002e-4	1.290e-4
DTLZ2(3)	AVG	4.780e-4[1]	1.351e-3[4]	1.492e-3[5]	1.251e-3[3]	6.803e-4[2]	**4.631e-4[0]**
	SD	2.662e-5	2.523e-4	1.094e-4	1.189e-4	5.966e-5	3.968e-5
DTLZ2(5)	AVG	7.960e-4[1]	1.956e-3[5]	8.597e-4[2]	1.699e-3[4]	**7.890e-4[0]**	1.035e-3[3]
	SD	4.009e-5	1.052e-4	8.002e-5	2.700e-4	6.300e-5	9.256e-5
DTLZ2(8)	AVG	**1.996e-3[0]**	2.762e-3[4]	4.956e-3[5]	2.584e-3[3]	2.050e-3[2]	2.039e-3[1]
	SD	4.200e-4	1.935e-4	3.586e-4	9.665e-5	8.653e-5	3.993e-5
DTLZ2(10)	AVG	**2.762e-3[0]**	4.532e-3[3]	6.308e-3[5]	4.145e-3[2]	4.905e-3[4]	3.249e-3[1]
	SD	4.001e-4	8.256e-5	1.125e-4	2.336e-4	9.524e-5	1.025e-4
DTLZ3(3)	AVG	1.302e-3[1]	1.653e-3[4]	1.483e-3[2]	1.555e-3[3]	1.654e-3[5]	**1.210e-3[0]**
	SD	6.552e-5	9.036e-5	8.654e-5	1.702e-4	2.365e-4	5.203e-5
DTLZ3(5)	AVG	1.100e-3[1]	5.960e-3[4]	6.225e-3[5]	5.743e-3[3]	4.424e-3[2]	**1.003e-3[0]**
	SD	3.257e-4	1.036e-4	2.654e-4	1.222e-4	2.459e-4	3.224e-5
DTLZ3(8)	AVG	**4.223e-3[0]**	1.265e-2[4]	1.733e-2[5]	1.185e-2[2]	1.246e-2[3]	4.562e-3[1]
	SD	2.223e-4	4.965e-4	3.586e-3	2.220e-3	4.653e-4	4.201e-4
DTLZ3(10)	AVG	**4.800e-3[0]**	1.566e-2[4]	1.777e-2[5]	1.430e-2[3]	1.021e-2[2]	7.402e-3[1]
	SD	2.558e-4	4.247e-4	3.639e-4	1.222e-3	2.685e-4	3.785e-4
DTLZ4(3)	AVG	1.025e-4[1]	1.455e-3[4]	1.035e-1[5]	1.399e-3[3]	**1.002e-4[0]**	2.245e-4[2]
	SD	2.336e-5	7.589e-5	1.263e-3	2.963e-4	5.220e-5	1.583e-5
DTLZ4(5)	AVG	1.488e-4[1]	1.326e-3[4]	9.336e-2[5]	1.286e-3[3]	1.127e-3[2]	**1.205e-4[0]**
	SD	2.369e-5	4.522e-5	1.063e-3	4.69e-5	4.205e-5	2.366e-5
DTLZ4(8)	AVG	**1.700e-4[0]**	5.230e-3[4]	2.253e-1[5]	5.131e-3[3]	4.904e-3[2]	1.925e-4[1]
	SD	1.236e-5	4.005e-5	1.203e-3	2.001e-4	1.698e-4	2.369e-5
DTLZ4(10)	AVG	**1.235e-3[0]**	6.256e-3[3]	2.925e-1[5]	7.100e-3[4]	4.025e-3[2]	3.002e-3[1]
	SD	4.558e-5	2.368e-4	1.203e-3	2.680e-4	4.330e-4	5.061e-5
UF1	AVG	5.321e-3[2]	**3.358e-3[0]**	1.025e-2[3]	2.586e-2[5]	1.255e-2[4]	4.998e-3[1]
	SD	3.963e-4	5.002e-5	2.996e-4	1.023e-3	7.669e-4	1.325e-4
UF2	AVG	8.252e-3[1]	3.569e-2[5]	3.094e-2[3]	3.205e-2[4]	2.780e-2[2]	**7.952e-3[0]**
	SD	1.025e-4	5.336e-4	4.890e-4	1.002e-3	8.336e-4	2.698e-4
UF3	AVG	**1.021e-2[0]**	3.088e-2[5]	2.786e-2[2]	2.864e-2[3]	2.961e-2[4]	2.552e-2[1]
	SD	3.508e-4	1.558e-3	4.023e-3	7.995e-4	8.030e-4	1.056e-3
UF4	AVG	**2.112e-2[0]**	4.008e-2[4]	3.850e-2[3]	3.729e-2[2]	3.598e-2[1]	4.056e-2[5]
	SD	1.003e-3	2.558e-4	1.925e-3	4.539e-4	7.225e-4	1.052e-3

AVG and SD are the abbreviations of average and stand deviation, respectively. The best result of each test problem is in bold.

TABLE 3. Average and standard deviation of HV values obtained by six algorithms on the DTLZ and UF test problems.

Problem		NSGA-III-GKM	NSGA-III	MOEA/D	U-NSGA-III	DC-NSGA-III	B-NSGA-III
DTLZ1(3)	AVG	0.975[1]	0.871[4]	0.968[2]	0.852[5]	0.880[3]	**0.987[0]**
	SD	2.336e-3	1.589e-2	4.558e-3	3.698e-3	1.235e-2	5.369e-3
DTLZ1(5)	AVG	0.422[2]	0.403[4]	0.396[5]	0.412[3]	0.423[1]	**0.496[0]**
	SD	2.035e-2	1.968e-2	4.620e-3	6.220e-3	5.302e-3	7.998e-4
DTLZ1(8)	AVG	**0.528[0]**	0.445[5]	0.456[3]	0.449[4]	0.468[2]	0.502[1]
	SD	4.023e-3	5.269e-3	7.012e-4	5.630e-3	2.965e-3	4.368e-3
DTLZ1(10)	AVG	**0.988[0]**	0.765[5]	0.885[4]	0.906[3]	0.923[2]	0.974[1]
	SD	1.096e-2	2.556e-2	3.086e-3	4.880e-4	3.208e-3	2.362e-4
DTLZ2(3)	AVG	0.775[3]	0.803[1]	0.725[5]	0.764[4]	0.800[2]	**0.826[0]**
	SD	1.396e-3	2.502e-3	4.336e-3	2.052e-3	2.001e-2	5.925e-3
DTLZ2(5)	AVG	0.558[2]	0.458[5]	**0.672[0]**	0.603[1]	0.459[4]	0.526[3]
	SD	9.000e-3	2.934e-2	5.336e-3	5.287e-3	4.529e-3	6.382e-4
DTLZ2(8)	AVG	**0.975[0]**	0.769[5]	0.798[2]	0.772[4]	0.783[3]	0.925[1]
	SD	7.632e-4	8.021e-4	4.336e-3	2.015e-3	1.931e-3	2.064e-3
DTLZ2(10)	AVG	**0.730[0]**	0.556[4]	0.632[3]	0.664[2]	0.525[5]	0.699[1]
	SD	5.669e-4	4.826e-3	3.250e-2	2.635e-2	1.558e-2	4.025e-2
DTLZ3(3)	AVG	0.956[1]	0.896[3]	0.853[4]	0.775[5]	0.900[2]	**0.967[0]**
	SD	3.998e-2	9.025e-4	8.882e-4	7.936e-4	4.302e-3	2.014e-2
DTLZ3(5)	AVG	0.695[1]	0.663[5]	**0.768[0]**	0.685[4]	0.690[3]	0.694[2]
	SD	2.526e-2	4.589e-2	5.235e-2	1.025e-1	4.589e-3	3.002e-2
DTLZ3(8)	AVG	**0.649[0]**	0.536[4]	0.502[5]	0.546[3]	0.569[2]	0.630[1]
	SD	3.577e-2	7.256e-4	8.258e-4	4.036e-4	5.298e-3	4.225e-2
DTLZ3(10)	AVG	**0.933[0]**	0.621[4]	0.608[5]	0.665[3]	0.789[2]	0.901[1]
	SD	3.669e-3	2.568e-2	1.589e-1	5.689e-3	8.695e-2	4.005e-3
DTLZ4(3)	AVG	0.556[1]	0.502[4]	0.496[5]	0.520[3]	**0.566[0]**	0.541[2]
	SD	2.598e-2	4.568e-2	3.668e-2	9.025e-3	1.258e-1	4.506e-2
DTLZ4(5)	AVG	0.695[3]	0.663[5]	0.675[4]	0.702[2]	0.733[1]	**0.759[0]**
	SD	3.846e-3	4.895e-2	6.000e-2	8.565e-2	3.654e-2	9.058e-3
DTLZ4(8)	AVG	**0.736[0]**	0.551[4]	0.498[5]	0.589[3]	0.602[2]	0.677[1]
	SD	4.558e-2	5.025e-4	1.232e-4	6.556e-4	8.025e-3	1.033e-2
DTLZ4(10)	AVG	**0.795[0]**	0.625[4]	0.584[5]	0.648[3]	0.702[2]	0.746[1]
	SD	4.582e-3	6.583e-2	1.222e-1	7.965e-2	3.025e-3	2.001e-1
UF1	AVG	**0.902[0]**	0.856[4]	0.796[5]	0.860[3]	0.896[1]	0.884[2]
	SD	4.258e-2	3.025e-2	6.335e-3	1.698e-1	4.658e-2	9.025e-2
UF2	AVG	0.756[2]	0.695[4]	0.543[5]	0.740[3]	0.762[1]	**0.839[0]**
	SD	4.258e-3	7.265e-2	4.025e-2	3.025e-3	6.124e-3	7.025e-2
UF3	AVG	**0.946[0]**	0.885[3]	0.782[5]	0.866[4]	0.892[2]	0.925[1]
	SD	1.335e-2	5.369e-3	7.214e-3	4.980e-2	5.693e-2	1.250e-1
UF4	AVG	0.752[1]	0.668[5]	**0.880[0]**	0.705[4]	0.749[2]	0.712[3]
	SD	5.892e-2	4.698e-2	5.336e-2	7.598e-2	9.025e-2	7.025e-2

AVG and SD are the abbreviations of average and stand deviation, respectively. The best result of each test problem is in bold.

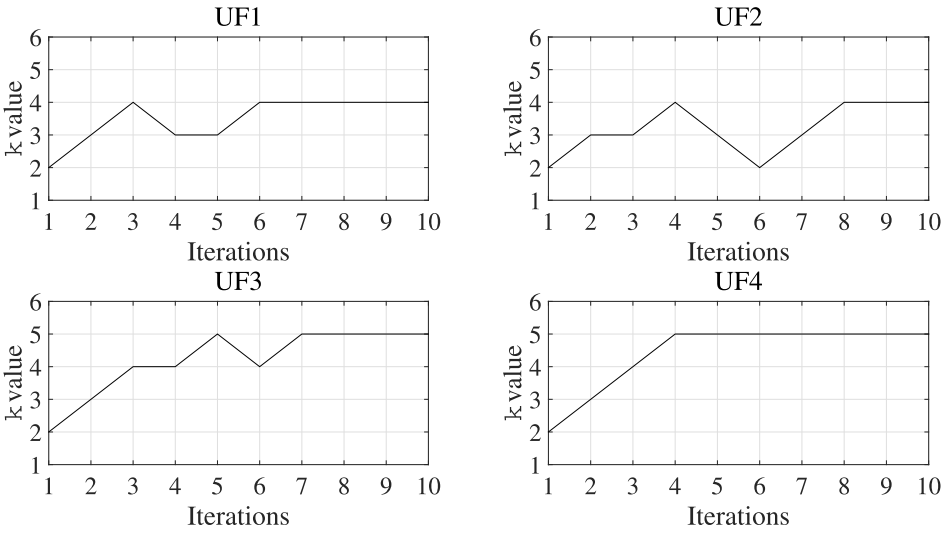

FIGURE 5. Changing curves of k value in the tenth experiment of each of the UF1-4 problems.

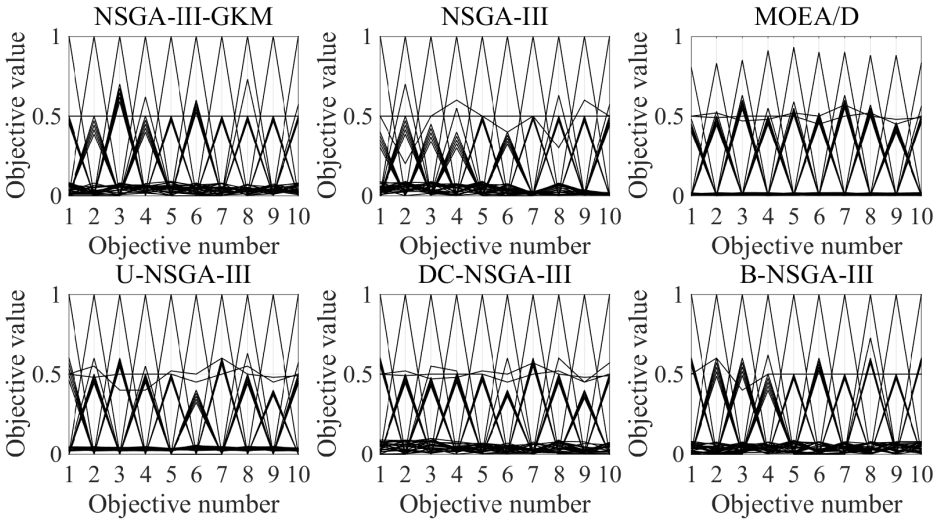

FIGURE 6. Parallel coordinates of the PFs obtained by six algorithms of the DTLZ1 problem with 10 objectives.

In order to test the differences for statistical significance, the Kruskal-Wallis test [33] with a 5% significance level is applied for all pairwise comparisons, and the performance score [34] is adopted to rank all the algorithms: the smaller the score is, the better the algorithm is.

B. ANALYSES OF RESULTS

Tables 2 and 3 show the average and standard deviation of IGD and HV values obtained by six algorithms on the DTLZ and UF problems. The numbers in brackets in Table 2 and 3 are ranks of the six algorithms for each test problem. We can see from Table 2 that the NSGA-III-GKM has better performance on eleven of the twenty test problems. We can see from Table 3 that the NSGA-III-GKM has better performance on ten of the twenty test problems. Tables 2 and 3 intuitively indicate that NSGA-III-GKM has better performance than the other five state-of-the-art algorithms when solving MaOPs with more than five objectives.

Figs. 4-5 show the changing curves of k value in the tenth experiments of DTLZ1-4 and UF1-4 problems. We can see from Figs. 4-5 that the k values of NSGA-III-GKM realize automatic learning.

Fig. 6 shows the parallel coordinates of the PFs obtained by five algorithms of DTLZ1(10) problem. The PFs obtained by NSGA-III-GKM, NSGA-III, U-NSGA-III, DC-NSGA-III and B-NSGA-III are similar in terms of diversity and convergence. However, the convergence of the PF obtained by MOEA/D is worse, which verifies the weakness of MOEA/D.

Fig. 7 shows the parallel coordinates of the PFs obtained by five algorithms of DTLZ2(10). The PFs obtained by the NSGA-III-GKM, DC-NSGA-III and B-NSGA-III are similar in terms of diversity and convergence. The convergence of the PF obtained by the NSGA-III is worse in the ninth objective.

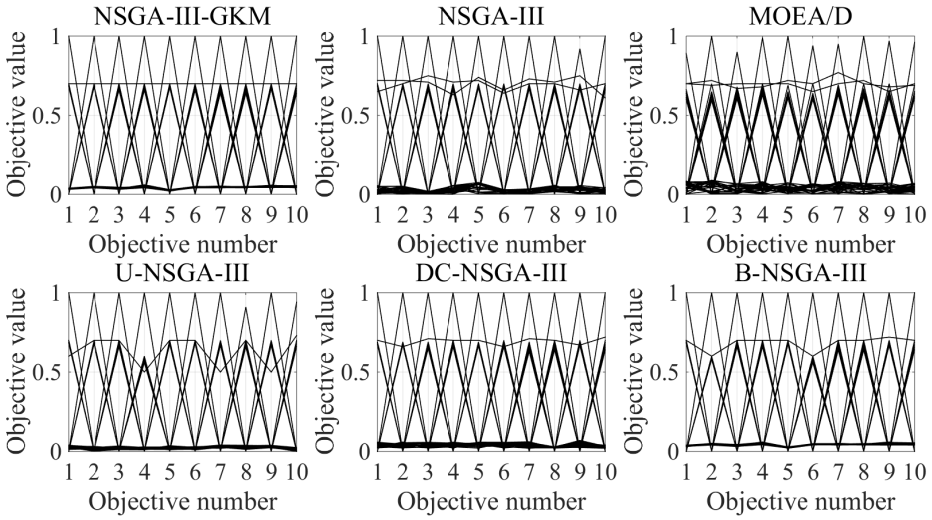

FIGURE 7. Parallel coordinates of the PFs obtained by six algorithms of the DTLZ2 problem with 10 objectives.

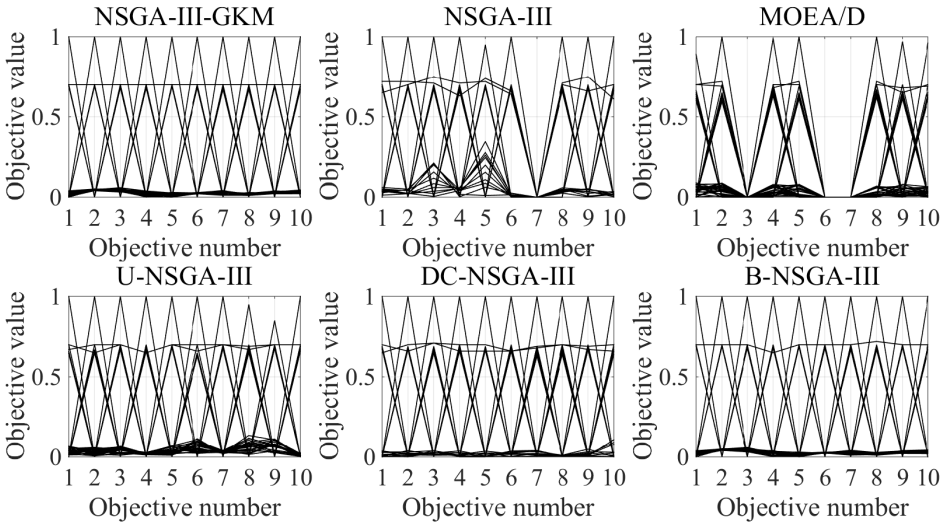

FIGURE 8. Parallel coordinates of the PFs obtained by six algorithms of the DTLZ3 problem with 10 objectives.

The convergence of results obtained by MOEA/D is worse in the first, third, sixth, seventh, ninth and tenth objectives. The convergence of the PF obtained by the U-NSGA-III is worse in the eighth and tenth objectives.

Figu. 8 shows the parallel coordinates of the PFs obtained by five algorithms of DTLZ3(10). The PFs obtained by the NSGA-III-GKM, DC-NSGA-III and B-NSGA-III are similar in terms of diversity and convergence. The PF obtained by the NSGA-III loses the part of the seventh objective and the PF obtained by MOEA/D loses the parts of the third, sixth and seventh objectives. The convergence of the PF obtained by the U-NSGA-III is worse in the eighth and ninth objectives.

Fig. 9 shows the parallel coordinates of the PFs obtained by the five algorithms of DTLZ4(10) problem. The PFs obtained by the NSGA-III-GKM and B-NSGA-III are better than those obtained by the NSGA-III, MOEA/D, U-NSGA-III and DC-NSGA-III. The PF obtained by the NSGA-III loses the parts of the second and third objectives and the PF obtained by MOEA/D loses the parts of the second, third and eighth objectives. The convergence of the results obtained by the U-NSGA-III is worse in the second, third and seventh objectives. The convergence of results obtained by DC-NSGA-III is worse in the second and seven objectives.

We can see from Figs. 6-9 that the NSGA-III-GKM and B-NSGA-III have better diversity and convergence than the other four algorithms when solving MaOPs with 10 objectives.

Although the GKM operation is added to the NSGA-III, the computational cost is lightly increased. According to the results, the NSGA-III-GKM increases the computational

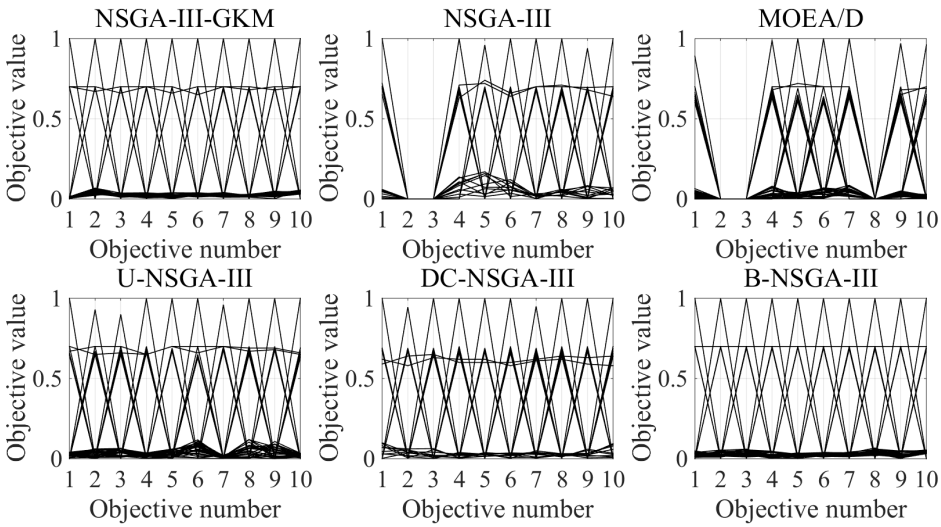

FIGURE 9. Parallel coordinates of the PFs obtained by six algorithms of the DTLZ4 problem with 10 objectives.

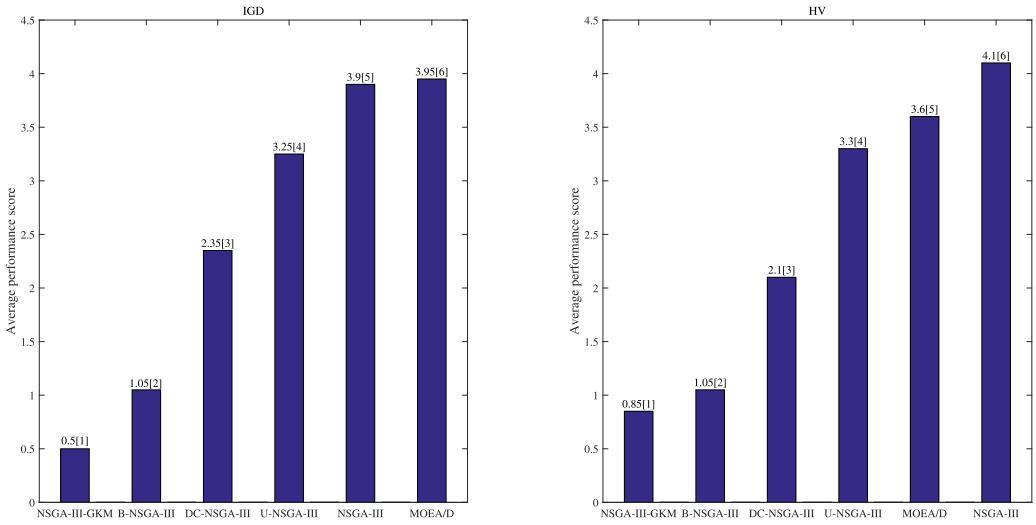

FIGURE 10. Ranking of the average performance score over all test problems for the six algorithms.

time (0.393s) by 1.2% compared to the average computational time of the other five comparison algorithms after 30 independent runs, which verifies that the addition of the GKM operation hardly increases the computational cost.

To rank these six algorithms, the average performance scores over all 20 test problems based on the IGD and HV indicators are presented in Fig. 10. The smaller the average performance score is, the better the overall performance of the algorithm is. The rank of each algorithm is given in the corresponding bracket. NSGA-III-GKM obtains the best average performance score based on the IGD and HV indicators, respectively. Then, the overall performance of the NSGA-III-GKM is better than that of the NSGA-III, MOEA/D, U-NSGA-III, DC-NSGA-III and B-NSGA-III on the DTLZ1-4 and UF1-4 problems.

V. CONCLUSION AND FUTURE RESEARCH

In this paper, we propose an improved NSGA-III using a genetic K-means clustering algorithm, called the NSGA-III-GKM, which improves the convergence and diversity by separating the space objective into subspaces and introducing the PBI aggregation function in the niche-preservation operation phase. We design a comparative simulation for the NSGA-III-GKM and five other state-of-the-art algorithms MOEAs. The results demonstrate that the proposed NSGA-III-GKM has better overall performance than the other four algorithms.

Although better results are obtained using the NSGA-III-GKM, there are still several issues worth studying for further improvements. Our future research work includes the following directions: 1) studying the adaptive positive coefficients of GKM clustering algorithm in different MaOPs;

2) designing local operations in different phases of the NSGA-III-GKM; 3) attempting to replace GKM with other clustering algorithms; 4) solving MaOPs with complex constraints; and 5) applying the NSGA-III-GKM to real-world applications.

REFERENCES

[1] G. N. Abel and A. López-Jaimes, "An investigation into many-objective optimization on combinatorial problems: Analyzing the pickup and delivery problem," *Swarm Evol. Comput.*, vol. 38, pp. 218–230, Feb. 2018.

[2] B. L. Kimia and H. Ali, "An evolutionary based framework for many-objective optimization problems," *Eng. Comput.*, vol. 35, no. 4, pp. 1805–1828, 2018.

[3] M. Asafuddoula, T. Ray, and R. Sarker, "A decomposition-based evolutionary algorithm for many objective optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 3, pp. 445–460, Jun. 2015.

[4] S. Bechikh, M. Elarbi, and L. B. Said, "Many-objective optimization using evolutionary algorithms: A survey," in *Recent Adv. Evol. Multi-objective Optim.* (Adaptation, Learning, and Optimization), vol. 20. Cham, Switzerland: Springer, 2017, pp. 105–137.

[5] F. Goulart and F. Campelo, "Preference-guided evolutionary algorithms for many-objective optimization," *Inf. Sci.*, vol. 329, pp. 236–255, Feb. 2016.

[6] N. Srinivas and K. Deb, "Multiobjective function optimization using nondominated sorting genetic algorithms," *Evol. Comput.*, vol. 2, no. 3, pp. 221–248, 1995.

[7] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[8] K. Deb and H. Jain, "An Evolutionary Many-objective optimization algorithm using reference-point based non-dominated sorting approach, part I: Solving problems with box constraints," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 577–601, Aug. 2014.

[9] J. Zhang, S. Wang, Q. Tang, Y. Zhou, and T. Zeng, "An improved NSGA-III integrating adaptive elimination strategy to solution of many-objective optimal power flow problems," *Energy*, vol. 172, pp. 945–957, Apr. 2019.

[10] F. Ruan, R. Gu, T. Huang, and S. Xue, "A big data placement method using NSGA-III in meteorological cloud platform," *EURASIP J. Wireless Commun. Netw.*, vol. 2019, no. 1, p. 143, 2019.

[11] X. Yuan, H. Tian, Y. Yuan, Y. Huang, and R. M. Ikram, "An extended NSGA-III for solution multi-objective hydro-thermal-wind scheduling considering wind power cost," *Energy Convers. Manage.*, vol. 96, pp. 568–578, May 2015.

[12] X. Bi and C. Wang, "An improved NSGA-III algorithm based on objective space decomposition for many-objective optimization," *Soft Comput.*, vol. 21, no. 15, pp. 4269–4296, 2017.

[13] Y. Yuan, H. Xu, and B. Wang, "An improved NSGA-III procedure for evolutionary many-objective optimization," in *Proc. Annu. Conf. Genetic Evol. Comput.*, 2014, pp. 661–668.

[14] H. Seada and K. Deb, "U-NSGA-III: A unified evolutionary optimization procedure for single, multiple, and many objectives: Proof-of-principle results," in *Proc. Int. Conf. Evol. Multi—Criterion Optim.*, 2015, pp. 34–49.

[15] M. Abouhawwash, H. Seada, and K. Deb, "Towards faster convergence of evolutionary multi-criterion optimization algorithms using Karush Kuhn Tucker optimality based local search," *Comput. Oper. Res.*, vol. 79, pp. 331–346, Mar. 2017.

[16] H. Seada, M. Abouhawwash, and K. Deb, "Towards a better balance of diversity and convergence in NSGA-III: First results," in *Proc. Int. Conf. Evol. Multi—Criterion Optim.*, 2017, pp. 545–559.

[17] H. Seada, M. Abouhawwash, and K. Deb, "Multiphase balance of diversity and convergence in multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 23, no. 3, pp. 503–513, Jun. 2019.

[18] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.

[19] N. Bouhmala, A. Viken, and J. B. Lønnum, "Enhanced genetic algorithm with K-averages for the clustering problem," *Int. J. Model. Optimization.*, vol. 5, no. 2, pp. 150–154, 2015.

[20] D. Mustafi and G. Sahoo, "A hybrid approach using genetic algorithm and the differential evolution heuristic for enhanced initialization of the k-averages algorithm with applications in text clustering," *Soft Comput.*, vol. 23, no. 15, pp. 6361–6378, 2019.

[21] S. Saharan, R. Baragona, M. E. Nor, R. M. Salleh, and N. M. Asrah, "Clustering for binary data sets by using genetic algorithm-incremental K-averages," *J. Phys. Conf. Ser.*, vol. 995, no. 1, pp. 12–38, 2018.

[22] T. Glasmachers, "A fast incremental BSP tree archive for non-dominated points," in *Proc. Int. Conf. Evol. Multi—Criterion Optim.*, 2017, pp. 252–266.

[23] X. Bi and C. Wang, "An improved NSGA-III algorithm based on elimination operator for many-objective optimization," *Memetic Comput.*, vol. 9, no. 4, pp. 361–383, Dec. 2017.

[24] A. B. Ruiz, R. Saborido, and M. Luque, "A preference-based evolutionary algorithm for multiobjective optimization: The weighting achievement scalarizing function genetic algorithm," *J. Global Optim.*, vol. 62, no. 1, pp. 101–129, 2015.

[25] I. Das and J. E. Dennis, "Normal-bounday intersection: A new method for generating Pareto optimal points in multicriteria optimization problems," *SIAM J. Optim.*, vol. 8, no. 3, pp. 631–657, Aug. 1998.

[26] W. R. Alkhayri, S. S. Owais, and M. Shkoukani, "A new selection operator-CSM in genetic algorithms for solving the TSP," *Int. J. Adv. Manuf. Tech.*, vol. 7, no. 10, pp. 62–66, 2016.

[27] K. Deb and H. G. Beyer, "Self-adaptive genetic algorithms with simulated binary crossover," *Evol. Comput.*, vol. 9, no. 2, pp. 197–221, 2001.

[28] L. M. Li, K. D. Lu, G. Q. Zeng, L. Wu, and M. R. Chen, "A novel real-coded population-based extremal optimization algorithm with polynomial mutation: A non-parametric statistical study on continuous optimization problems," *Neurocomputing*, vol. 174, pp. 577–587, Jan. 2016.

[29] J. Bai and H. Liu, "Multi-objective artificial bee algorithm based on decomposition by PBI method," *Appl. Intell.*, vol. 45, no. 4, pp. 976–991, 2016.

[30] H. Ishibuchi, R. Imada, Y. Setoguchi, and Y. Nojima, "Reference point specification in inverted generational distance for triangular linear Pareto front," *IEEE Trans. Evol. Comput.*, vol. 22, no. 6, pp. 961–975, Dec. 2018.

[31] H. Ishibuchi, H. Masuda, Y. Tanigaki, and Y. Nojima, "Modified distance calculation in generational distance and inverted generational distance," *Evolutionary Multi-Criterion Optimization* (Lecture Notes in Computer Science), vol. 9019. Cham, Switzerland: Springer, 2015, pp. 110–125.

[32] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Trans. Evol. Comput.*, vol. 3, no. 4, pp. 257–271, Nov. 1999.

[33] A. E. Bargagliotti and R. N. Greenwell, "Combinatorics and statistical issues related to the Kruskal-Wallis statistic," *Commun. Statist.—Simul. Comput.*, vol. 44, no. 2, pp. 533–550, 2015.

[34] J. Bader and E. Zitzler, "Hype: An algorithm for fast hyper, volume-based many-objective optimization," *Evol. Comput. J.*, vol. 19, no. 1, pp. 45–76, 2011.