Multi-objective Optimization Algorithm Based on Uniform Design and Differential Evolution

Jinrong He[1,2,3], Dongjian He[1,3(✉)], Aiqing Shi[1,3], and Guoliang He[4]

[1] College of Mechanical and Electronic Engineering, Northwest A&F University, Yangling 712100, China
hdjl68@nwsuaf.edu.cn
[2] College of Mathematics and Computer Science, Yan'an University, Yan'an 716000, China
[3] Key Laboratory of Agricultural Internet of Things, Ministry of Agriculture and Rural Affairs, Yangling 712100, China
[4] School of Computer, Wuhan University, Wuhan 430070, China

Abstract. The multi-objective optimization problem is an important research direction in the field of optimization. Because the traditional mathematical programming method often cannot achieve the optimal global solution, the researchers introduced the heuristic method into the multi-objective optimization problem. The heuristic method is a method of searching based on empirical rules, which can get the optimal solution or solution set of problems in the limited search space. In this paper, we proposed a multi-objective evolutionary algorithm based on uniform design and differential evolution, which use the uniform design table to construct the weight vector and utilize the crossover in differential evolution and mutation process to replace the simulated binary intersection and the simulated polynomial variation. Compared with the classical algorithm, the experimental results show that the improved algorithm is superior to the original algorithm.

Keywords: Multi-objective optimization · Heuristic method · Evolutionary algorithm · Uniform design · Differential evolution

1 Introduction

In general, optimization problem with more than one optimization goals need to be processed at the same time is called multi-objective optimization problem (MOP, [1]). A typical approach to directly solve a multi-objective optimization problem is to convert the multi-objective optimization problem into a single-objective optimization problem in a certain way (aggregation by weight or transformation multiple objectives into constraint conditions), to obtain an optimal solution. However, this kind of traditional method can only be applied to a small set of problems and has poor generalization. Secondly, most algorithms can only get a locally optimal solution. If the neighborhood is enlarged, the complexity of the algorithm will be increased.

https://doi.org/10.1007/978-981-15-5577-0_14

In 1967, Rosenberg proposed that genetic algorithms could be used to solve multi-objective optimization problems [2]. Eighteen years later, Schafferf proposed a vector-based fitness calculation method, called Vector Evaluated Genetic Algorithm (VEGA, [3]), which is an extension of SGA (Simple Genetic Algorithms). A multi-objective genetic algorithm combined with genetic algorithms is proposed for the first time, creating a precedent for multi-objective heuristic optimization algorithms. Since 1967, multi-objective optimization problems have begun to attract the attention of researchers in various fields. By the end of the 20th century, as a new method to solve multi-objective optimization problems, the evolutionary algorithm of multi-objective heuristic optimization algorithms has attracted much attention, and some have successfully applied to the project.

Multi-objective optimization algorithms based on evolutionary algorithms are mostly characterized by non-dominated selection and diversity preservation strategies based on shared functions. In 1993, Fonseca and Fleming proposed the Multi-Objective Genetic Algorithm (MOGA, [4]). Srinivas and Deb proposed the Non-Dominated Sorting Genetic Algorithm (NSGA, [5]). Horn and Nafpliotis proposed the Niched Pareto Genetic Algorithms (NPGA, [6]), which are commonly called the first-generation evolutionary multi-objective optimization algorithms. In 2002, Zitzler and Thiele proposed the Strength Pareto Evolutionary Algorithm (SPEA, [7]). In 2000, Knowles and Corne proposed the Pareto Archived Evolution Strategy (PAES, [8]), and in 2002, Deb et al. proposed the NSGA-II, which is still widely used up to now, obtained by improving NSGA [9]. In 2003, Moore and Chapman proposed the first method to solve the multi-objective optimization problem using PSO, in which they introduced the Pareto preference to change records and use individual optimal locations [10]. Ray et al. introduced the Pareto sorting strategy into PSO to solve multi-objective optimization problem, and adopted crowding degree to maintain the diversity of particles [11]. Coello and Lechuga introduced the Pareto archiving evolution strategy into PSO to obtain MOPSO [12], and based on artificial immunity, they proposed the multi-objective immune system algorithm (MISA, [13]), which first applied an artificial immune system to solve the multi-objective optimization problems. Luh, Chueh, and Liu proposed the multi-objective immune algorithm (MOIA, [14]), in which the antibody adopts binary string encoding. Based on distributed estimation, Khan et al. proposed the multi-objective Bayesian optimization algorithm (mBOA, [15]), which combined the non-dominant selection strategy in NSGA-II with Bayesian Optimization Algorithm (BOA) to solve the spoofing multi-objective optimization problem. Laumanns et al. also proposed a multi-objective optimization algorithm based on Bayesian optimization, which combined SPEA2 and BOA to solve the multi-objective knapsack problem [16].

Domestic researchers have also proposed a variety of multi-objective heuristic optimization algorithms. In response to the multi-objective optimization problem with constraints, Prof. Cai Zixing and Dr. Wang Yong from Central South University proposed a new evolutionary algorithm based on the existing multi-objective optimization technology solving the optimization problem with constraints and getting good results [17]. In the research of multi-objective algorithm based on particle swarm, Li Xiaodong introduced the non-dominant sorting mechanism of NSGA-II into PSO and adopted niche strategy to ensure the diversity of solutions, achieving excellent results [18]. In the research on multi-objective Optimization Algorithm based on artificial immunity, Prof. Jiao licheng, Prof. Gong Maoguo, and Dr. Shang Ronghua et al. proposed Immune Dominance Clone Multi-objective Optimization Algorithm (IDCMA, [19]). In 2008, Prof. Gong Maoguo and Prof. Jiao Licheng et al. presented the nonsorted neighbor immune algorithm (NNIA, [20]), which used an individual selection method based on non-dominant neighbors. In the study of multi-objective optimization algorithm based on density estimation, Zhang Qingfu and Zhou Aimin et al. proposed Regularity Model based Multi-objective Estimation of Distribution Algorithm (RM-MEDA) which is a classic algorithm combined with distributed estimation algorithm and multi-objective optimization algorithm [21]. In the study of multi-objective optimization algorithm based on co-evolution, Liu Jing proposed a multi-objective evolutionary algorithm based on co-evolution, in which a crossover operator and three co-evolution operators were designed to maintain the diversity of individuals in the population and accelerate the convergence rate [22]. Tan Kaichen et al. proposed a new distributed cooperative co-evolutionary algorithm (DCCEA, [23]) based on the idea of distributed cooperative coevolution. In 2010, Prof. Tan Ying et al. from Peking University proposed the fireworks algorithm, which is a new idea of optimization problem study [24]. In 2013, Dr. Zheng Yujun et al. proposed the application of multi-target fireworks algorithm to the variables in oil crop production [25]. In 2016, Prof. Xie Chengwang et al. proposed to introduce an elite reverse learning mechanism into multi-objective fireworks algorithm to make the algorithm better, which provided a new research direction for multi-objective optimization [26].

This paper proposes MOEA/D based on uniform design and differential evolution. According to the existing multi-objective heuristic optimization algorithm combined with existing mathematical knowledge and differential evolution algorithm, the improvement of MOEA/D algorithm was proposed, and compared with the original algorithm in experimental analysis.

2 Multi-objective Evolutionary Algorithm

2.1 Problem Formulation of Multi-objective Optimization

In general, the multi-objective optimization problem consists of n decision variables, M objective functions, and K constraints, which can be formulated as follows.

$$\begin{cases} \min \ \vec{y} = f(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), \cdots, f_M(\vec{x})] \\ s.t. \quad g_i(\vec{x}) \le 0 \qquad i = 1, 2, \ldots, p \\ \qquad\ h_i(\vec{x}) = 0 \qquad i = 1, 2, \ldots, q \end{cases} \tag{1}$$

where $\vec{x} = (x_1, x_2, \ldots, x_n) \in D$ is the decision vector; $\vec{y} = (f_1, f_2, \ldots, f_M) \in Y$ represents the target vector; D is the decision space formed by the decision vector; Y represents the target space formed by the target vector; $g_i\left(\vec{x}\right) \leq 0(i = 1, 2, \ldots, p)$ defines p inequality constraints; $h_i\left(\vec{x}\right) = 0(i = 1, 2, \ldots, q)$ defines q equality constraints.

Definition 1 (Pareto dominance). If $x_1, x_2 \in X_f$ is two feasible solutions to the multi-objective optimization problem (1), then x_1 Pareto dominates x_2 if and only if

$$\forall i = 1, 2, \ldots, M,\ f_i(x_1) \leq f_i(x_2) \wedge \exists j \in \{1, 2, \ldots, M\},\ f_i(x_1) < f_i(x_2) \tag{2}$$

For short, x_1 dominates x_2, which can be denoted as $x_1 \succ x_2$, also called compared with x_2, x_1 is Pareto dominant.

Definition 2 (Pareto optimal). Solution $x^* \in X_f$ is Pareto optimal (or non-dominated), if and only if

$$!\exists\, x \in X_f : x \succ x^* \tag{3}$$

Definition 3 (Pareto frontier). The target vector corresponding to all Pareto optimal solutions in the Pareto optimal set P_s constitutes the surface, which can be represented as:

$$PF = \left\{F(x^*) = (f_1(x^*), f_2(x^*), \cdots, f_M(x^*))^T \,|\, x^* \in P_s\right\} \tag{4}$$

2.2 Multi-objective Evolutionary Algorithm Based on Decomposition (MOEA/D)

MOEA/D decomposes the multi-objective optimization problem into N scalar sub-problems, which can be solved simultaneously by evolving the population of solutions. In each generation, the population is a collection of optimal solutions for each sub-problem selected from all generations. The degree of association of two adjacent sub-problem keys is determined by the distance between their aggregation coefficient vectors. For two adjacent sub-problems, the optimal solution should be very similar. For each sub-question, it can be optimized with information about its adjacent sub-problems. The details of MOEA/D are summarized in Algorithm 1.

Algorithm 1. Decomposition-based multi-objective evolutionary algorithm

```
Step 1. Uniformly generate the initial population $\{x^1, x^2, ..., x^N\}$ with size $N$ in the feasible space.
Step 2. Calculate the Euclidean distance between any two weight vectors, and find the $T$ weight vectors nearest to each weight vector. For any $i = 1,2,...,N$, let $B_i = \{i_1, i_2, ..., i_t\}$ and $\lambda^{i_1}, \lambda^{i_2}, ..., \lambda^{i_T}$ is the nearest $T$ weight vectors to $\lambda^i$.
Step 3. Initialize $z = \{z_1, z_2, ..., z_m\}^T$, where $z_i = \min\{f_i(x^1), f_i(x^2), ..., f_i(x^N)\}$; Set $EP$ to null.
Step4. Genetic recombination and improvement. Select two random serial Numbers $k, l$ from $B_i$, use genetic operator to generate a new solution $y$ from $x^k, x^l$, and use the repair and improvement of $y$ based on the test problem to inspire the generation of $y'$.
Step 5. Update $z$. For any $j = 1,2,...,m$, if $z_i < f_i(y')$, then $z_i = f_i(y')$.
Step 6. Update neighborhood solution. For any $j \in B_i$, if $g^{te}(y'|\lambda^j, z) \le g^{te}(x^j | \lambda^j, z)$, then $x^j = y', FV^j = F(y')$.
Step 7. Update EP. Remove all vectors that are dominated by $F(y')$ from EP. If the vector in EP does not dominate $F(y')$, add $F(y')$ to EP.
Step 8. Determine the termination condition. If it terminates, output EP as a result, otherwise go to Step 3.
```

3 MOEA/D Based on Uniform Design and Differential Evolution

3.1 Generate a Weight Vector Using Uniform Design Methods

In MOEA/D, the simplex lattice point design is used to set the weight vector, but the generated weight vector points are not distributed uniformly, and there are too many points on the boundary, so that the weight of some targets in the sub-question is zero. We use a uniform design method to generate uniformly distributed weight vectors. The basic idea behind uniform design is to make the experimental points in the factor space have better uniform dispersion, and to minimize some uniformity measure. Let $U_n(q^s)$ denotes the uniform design table, where *U* denotes a uniform design table, *S* denotes the number of factors, *q* is the number of experimental levels, and *n* denotes the number of experiments.

In this paper, we use the good lattice point method to construct a uniform design table $U_N(N^{m-1})$ and use the reliable deviation (CD_2) as the uniformity measure to generate *N* weight vectors with uniform distribution of *m* dimensions. The use of good lattice point method to construct a uniform design table involves the following two definitions. The results of uniform design in this paper are discussed on the basis of the following definitions.

Definition 4. If each column element of a $n \times s$ matrix $U = (u_{ij})$ is a permutation of a set $\{1, 2, \ldots, q\}$, it is called a U – matrix and is denoted as $U(n, n^s)$. Transform u_{ij} to $x_{ij} = \frac{2u_{ij}-1}{2n}$, the matrix consisting of x_{ij} is denoted $X_u = (x_{ij})_{n \times s}$, the n rows of X_u can be regarded as n points above $C^s = [0, 1]^s$, and X_u is the set of points on C^s.

The n points determined by the U-matrix or the corresponding X_u are not necessarily evenly distributed, but finding the most uniform one among them can be used as a uniform design. In this paper, the centralization L_2 – deviation is used to measure the uniformity of the point set X_u. The centralization L_2 – deviation of the point set X_u is "0", and the calculation method is

$$\begin{aligned} CD_2(X_u) = & \left[\left(\frac{13}{12}\right)^s - \frac{2^{1-s}}{n} \sum_{k=1}^{n} \sum_{i=1}^{s} \left(2 + \left| x_{ki} - \frac{1}{2} \right| - \left| x_{ki} - \frac{1}{2} \right|^2 \right) + \frac{1}{n^2} \right. \\ & \left. + \frac{1}{n^2} \sum_{k,l=1}^{n} \sum_{i=1}^{s} \left(1 + \frac{1}{2}\left| x_{ki} - \frac{1}{2} \right| + \frac{1}{2}\left| x_{li} - \frac{1}{2} \right| - |x_{ki} - x_{li}| \right)^{\frac{1}{2}} \right] \end{aligned} \tag{5}$$

Definition 5. The U-matrix U^* of size $n \times s$ is called a uniform design, if and only if its corresponding X_{U^*} has the smallest CD_2 – value in all X_U of the same type, and U^* is denoted as $U(n^s)$.

3.2 Differential Evolution Method

The evolutionary process of MOEA/D mainly adopts simulated polynomial variation (PM) and simulated binary crossover (SBX). In this paper, mutation and crossover strategies in differential evolutionary algorithm (DE) are adopted to evolve population. The basic idea of differential variability is to randomly select two individuals from the population to obtain the difference vector, and to weight the difference vector and then sum the third individual according to certain rules to produce the variant individual. The basic idea of differential crossing is to mix the mutated individuals with a predetermined objective individual to generate test individuals. The specific calculation method is as follows:

The new solution obtained by DE mutation is $u' = (u'_1, u'_2, \ldots, u'_n)$, where the component u'_k is calculated by:

$$u'_k = x_{r_1}(g) + F \cdot (x_{r_2}(g) - x_{r_3}(g)), k \neq r_1 \neq r_2 \neq r_3 \tag{6}$$

where $k = 1, 2, \ldots, n$, F is the control parameter of the mutation process.

The new solution generated by DE crossover is $\overline{y} = (\overline{y_1}, \overline{y_2}, \ldots, \overline{y_n})$, where the component $\overline{y_k}$ is calculated by:

$$\overline{y_k} = \begin{cases} v_k & \textit{if} \quad rand_1 < CR,\ k = rand_2, \\ x_k^i & \textit{otherwise.} \end{cases} \tag{7}$$

in which

$$v_k = x_k^i + F \times (x_k^{r_1} - x_k^{r_2}) \tag{8}$$

where $k = 1, 2, \ldots, n$, and $rand_1 \in [0, 1]$, $rand_2 \in \{1, \ldots, n\}$ are two random numbers, and CR is the control parameter of the crossing process.

3.3 Algorithm Description

UDEMOEA/D, like the general idea of MOEA/D, decomposes the multi-objective optimization problem into sub-problems of n scalars, which solves all sub-problems simultaneously by evolving a solution population. For each generation of populations, the population is a collection of optimal solutions for each sub-question selected from all generations. The degree of association of two adjacent sub-problem keys is determined by the distance between their aggregation coefficient vectors. For two adjacent sub-problems, the optimal solution should be very similar. For each sub-question, just optimize it with information about its neighboring sub-problems. The specific algorithm steps are shown in Algorithm 2.

Algorithm 2. Decomposition multi-objective evolutionary algorithm based on uniform design and differential evolution

```
Step 1. Uniformly generate the initial population $\{x^1, x^2, ..., x^N\}$ with size $N$ in feasible space.
Step 2. Calculate the Euclidean distance between any two weight vectors, and find the $T$ weight vectors nearest to each weight vector. For each $i = 1,2,...,N$, let $B_i = \{i_1, i_2, ..., i_t\}$ and $\lambda^{i_1}, \lambda^{i_2}, ..., \lambda^{i_T}$ is the nearest $T$ weight vectors to $\lambda^i$.
Step 3. Initialize the uniformly distributed weight vector $z = \{z_1, z_2, ..., z_m\}^T$, and let $z_i = \min\{f_i(x^1), f_i(x^2), ..., f_i(x^N)\}$; set EP to null.
Step 4. Genetic recombination and improvement. Three serial numbers $j, k, l$ are randomly selected from $B_i$, and selects a suitable genetic operator according to the objective function to generate a new solution y from $x^j, x^k, x^l$, and generates a $y'$ based on the repair and improvement of the test problem.
Step 5. Update $z$. For any $j = 1,2,...,m$, if $z_i < f_i(y')$, then $z_i = f_i(y')$.
Step 6. Update the neighborhood solution. For any $j \in B_i$, if $g^{te}(y' | \lambda^j, z) \le g^{te}(x^j | \lambda^j, z)$, then $x^j = y', FV^j = F(y')$.
Step 7. Update EP. Remove all vectors that are dominated by $F(y')$ from $EP$. If the vector in EP does not dominate $F(y')$, then add $F(y')$ to EP.
Step 8. Determine the termination condition. If it terminates, output EP as a result, otherwise go to Step 3.
```

In Algorithm 2, Step 1 uses a uniform design method of good lattice points to construct the weight vector. The specific steps are shown in Algorithm 3.

```
Algorithm 3.  Good lattice point uniform design method
Step 1. Given the number $n$ of experiments, look for an integer $h$ smaller than $n$ and let the greatest common divisor of $n$ and $h$ be 1. A positive integer that meets this condition constitutes a vector $h = (h_1, h_2, \ldots, h_m)$, where $m$ is determined by the Euler function $\varphi(n)$.
Step 2. The $j$th column of the uniform design table is generated by calculating $u_{ij} = ih[\bmod\ n]$, and then $U_{ij}$ is generated according to the recursion formula.
Step 3. Select the first of all generated uniform design tables as the weight vector.
```

In Algorithm 3, the $[\bmod\ n]$ mentioned in Step 2 represents the congruence operation. If ih exceeds n, it is subtracted by an appropriate multiple of n to make the difference fall in $[1, n]$. At the same time, the recursion formula used to generate U_{ij} is:

$$\begin{cases} u_{1j} = h_j \\ u_{i+1j} = \begin{cases} u_{ij} + h_j & if \quad u_{ij} + h_j \leq n \\ u_{ij} + h_j - n & if \quad u_{ij} + h_j > n \end{cases} \end{cases} \quad i = 1, 2, \ldots, n-1 \tag{9}$$

4 Experiment Results

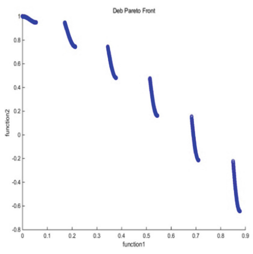
4.1 Experimental Settings

In the experiment, the initial population size is set to 100, the number of iterations is 1000, the simulated binary cross-distribution index used in the process is 15, the simulated polynomial variation distribution index is 20, the cross-index of differential evolution is 0.9, the variation index of differential evolution is 0.5. All simulation experiments were performed on a PC with Intel Core i3-2350M 2.30 GHz and 2G memory.

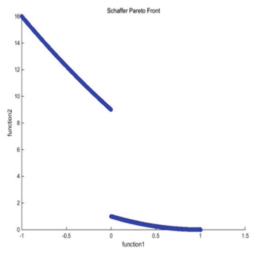
To evaluate the performance improvement of UDEMEA/D compared to MOEA/D, the coverage metric I_C is used to evaluate the optimal Pareto solution set. Coverage metric [7] is a relatively common measure of approximation that is used to quantitatively evaluate the dominance of two sets. Assume A and B are the two approximate Pareto optimal solution sets in the target space, the coverage index $I_C(A, B)$ is calculated as follows:

$$I_C(A, B) \overset{def}{=} \frac{|b \in B; \exists a \in A : a \rhd b|}{|B|} \tag{10}$$

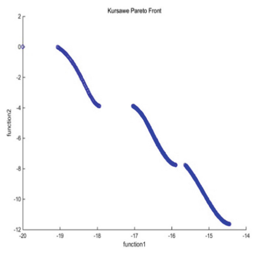
where $\rhd$ means that Pareto is not inferior, and $I_C(A, B) \in [0, 1]$, $I_C(A, B) = 1$ means that for all points in the set B, at least one point that is not inferior to it can be found in the set A. Instead, $I_C(B, A) = 0$ means that for all points in the set A, at least one point that is not inferior to A can be found in the set B. This indicator considers both $I_C(A, B)$ and $I_C(B, A)$, since $I_C(A, B)$ is not necessarily equal to $1 - I_C(B, A)$.



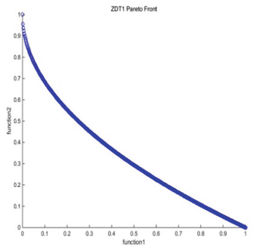
a. DEB



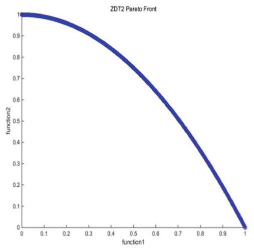
b. SCH



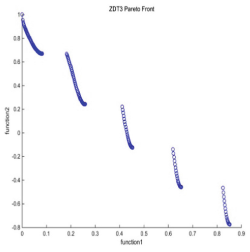
c. KUR



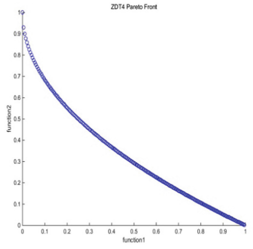
d. ZDT1



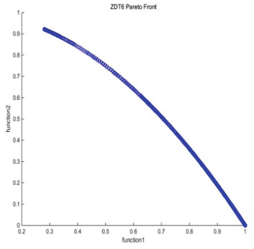
e. ZDT2



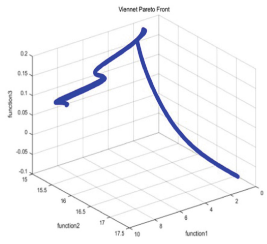
f. ZDT3



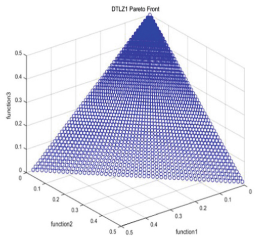
g. ZDT4



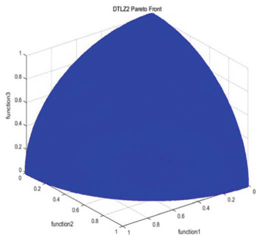
h. ZDT6



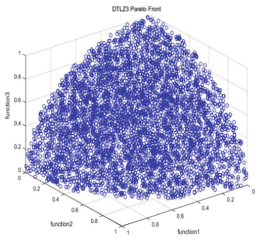
i. Viennet



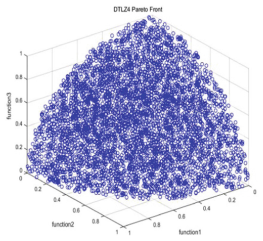
j. DTLZ1



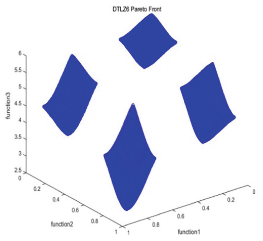
k. DTLZ2



l. DTLZ3



m. DTLZ4



n. DTLZ6

Fig. 1. The ideal Pareto frontier for 14 multi-target test problems

4.2 Benchmark Problems

The test functions used in the experiment were selected from the widely used test functions in the field of multi-objective optimization, including DEB, SCH, KUR, five ZDT problems, Viennet issues and five DTLZ issues [9]. Among them, the test functions DEB, KUR and SCH are two-objective optimization problems, and in five ZDT problems, ZDT1, ZDT2, ZDT3 have 30 decision variables, ZDT4 and ZDT6 have 10 decision variables, and Viennet problem is three-objective optimization problems, with 2 decisions Variables. The number of decision variables in five DTLZ problems and targets can be extended to any number. This article will set the value of k and $|x_k|$ according to Deb's suggestion. For DTLZ1, $k = 3$, $|x_k| = 5$ For DTLZ2, DTLZ3 and DTLZ4, $k = 3$, $|x_k| = 10$; for DTLZ6, $k = 3$, $|x_k| = 20$. Figure 1 is an ideal Pareto front for 14 test questions.

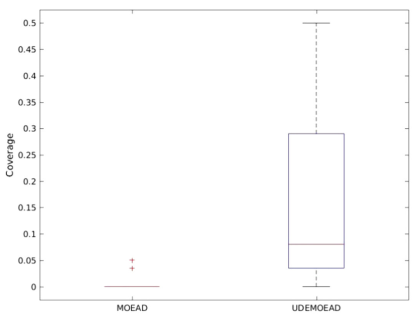
4.3 Experimental Results

UDEMOEA/D and MOEA/D were evaluated according to coverage index, and Table 1 and Fig. 2 were obtained respectively.

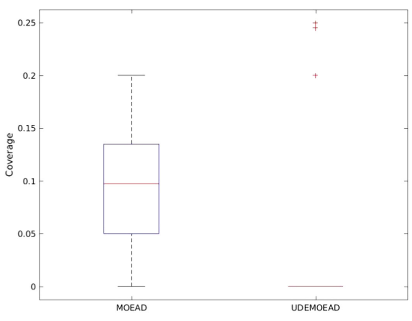
Table 1. UDEMOEA/D and MOEA/D coverage index evaluation box diagram

Coverage indicator value	UDEMOEA/D			MOEA/D		
	Min	Average	Max	Min	Average	Max
DEB	0	**0.1493**	0.5000	0	0.0028	0.0500
SCH	0	0.0498	0.2500	0	**0.0912**	0.2000
KUR	0.0450	**0.4360**	0.5000	0	0	0
ZDT1	0.4950	**0.4997**	0.5000	0	0	0
ZDT2	0.5000	**0.5000**	0.5000	0	0	0
ZDT3	0.5000	**0.5000**	0.5000	0	0	0
ZDT4	0.5000	**0.5000**	0.5000	0	0	0
ZDT6	0	**0.3665**	0.5000	0	0	0
Viennet	0	**0.0016**	0.0067	0	0.0012	0.0033
DTLZ1	0.3067	**0.3321**	0.3333	0	0	0
DTLZ2	0.0267	**0.0850**	0.2600	0	0	0
DTLZ3	0.0033	**0.2924**	0.3333	0	0.0046	0.1167
DTLZ4	0	**0.1439**	0.2767	0	0	0
DTLZ6	0	**0.2984**	0.3333	0	0	0

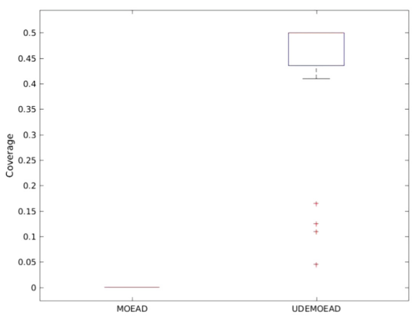
Table 1 shows the minimum, average, and maximum coverage of the solutions obtained by running the algorithm independently for 30 times. Figure 2 is the coverage index box graph of the two algorithms, in which MOEA/D represents the coverage of the solution obtained by MOEA/D for UDEMOEA/D, and UDEMOEA/D represents the coverage of the solution obtained by UDEMOEA/D to the solution obtained by MOEA/D. The display in Fig. 2 shows that in DEB, KUR, 5 ZDT problems, Viennet and 5 DTLZ problems, the box diagram of UDEMOEA/D is higher than the



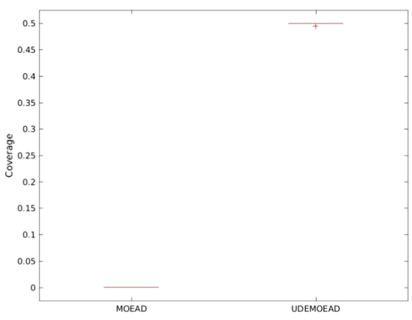
a. DEB



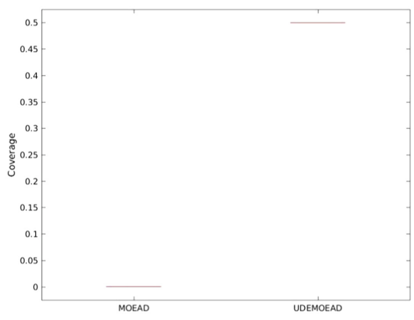
b. SCH



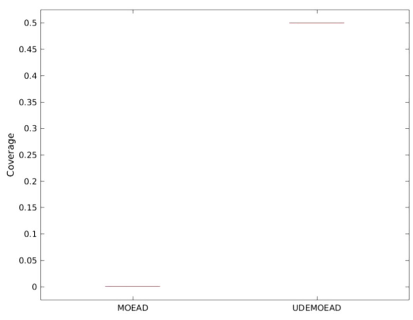
c. KUR



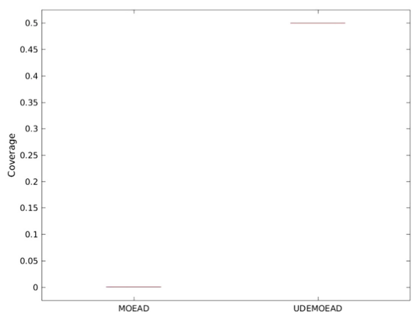
d. ZDT1



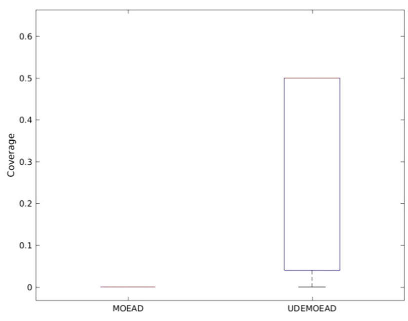
e. ZDT2



f. ZDT3

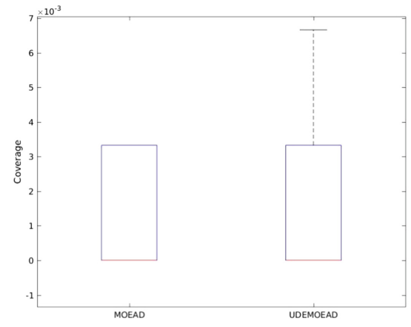


g. ZDT4

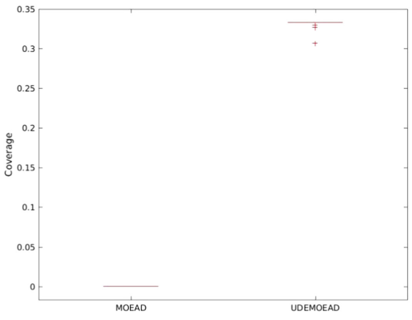


h. ZDT6

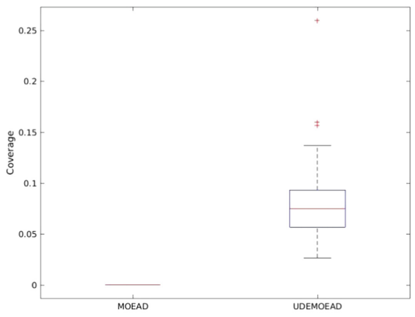
Fig. 2. MOEA/D and UDEMOEA/D cover the coverage index of the 14 test questions



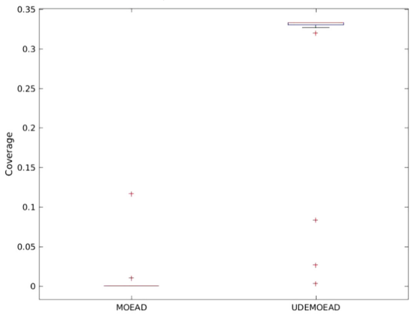
i. Viennet



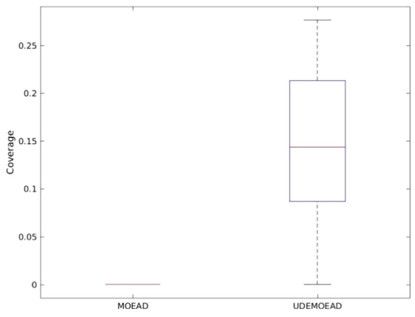
j. DTLZ1



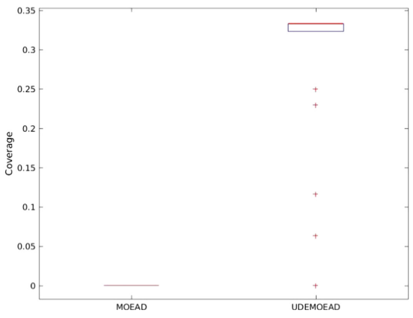
k. DTLZ2



l. DTLZ3



m. DTLZ4



n. DTLZ6

Fig. 2. (*continued*)

corresponding box diagram of MOEA/D, only MOEA in KUR/The box diagram of D is higher than the corresponding box diagram of UDEMOEA/D. At the same time, as can be seen from Table 1, among the four problems of ZDT1, ZDT2, ZDT3, and ZDT4, the coverage of the optimal Pareto solution set obtained by UDEMOEA/D for the optimal Pareto solution set of MOEA/D is 0.5. It can be concluded that under the same parameter setting, the optimal Pareto solution set obtained by UDEMOEA/D in solving 13 other test problems except SCH problem is better than the optimal Pareto solution set obtained by MOEA/D, and the solution set obtained by UDEMOEA/D is closer to the ideal Pareto frontier. UDEMOEA/D is superior to MOEA/D only considering approximation performance.

5 Conclusions

Based on that the distribution of weight vector points generated by the decomposition-based multi-objective evolutionary algorithm is not uniform, which leads to the fact that the final solution cannot be close to the ideal Pareto front, this paper proposed to use the uniform design table to construct the weight vector and utilize the crossover in differential evolution and mutation process to replace the simulated binary intersection and the simulated polynomial variation, and an improved algorithm is obtained—a decomposition-based multi-objective evolutionary algorithm based on uniform design and differential evolution. After the improved algorithm is implemented, it is compared with the original algorithm, and the obtained results are calculated to obtain the coverage index between the two algorithms to solve the test problems. By comparing the minimum, average and maximum of the two algorithms coverage index value, it can be seen that the improved algorithm is superior to the original algorithm in approximation performance.

Acknowledgement. This work was partially supported by National Natural Science Foundation of China (61902339, 61876136), the China Postdoctoral Science Foundation (2018M633585), Natural Science Basic Research Plan in Shaanxi Province of China (No. 2018JQ6060), and Google Supported Industry-University Cooperation and Education Project, Doctoral Starting up Foundation of Yan’an University (YDBK2019-06).

References

1. Anirban, M., Ujjwal, M., Sanghamitra, B., Carlos, A.C.C.: A survey of multi-objective evolutionary algorithms for data mining. IEEE Trans. Evol. Comput. **18**(1), 20–35 (2014)
2. Rosenberg, R.S.: Simulation of genetic populations with biochemical properties. Ph.D. thesis, University of Michigan, Michigan (1967)
3. Schaffer, J.D.: Multiple objective optimization with vector evaluated genetic algorithms. In: Proceedings of the International Conference on Genetic Algorithms and Their Applications, pp. 93–100. L. Erlbaum Associates, Hillsdale (1985)
4. Fonseca, C.M., Fleming, P.J.: Genetic algorithm for multi-objective optimization: formulation, discussion and generation. In: Forrest, S., (ed.) Proceedings of the 5th International Conference on Genetic Algorithms, pp. 416–423. Morgan Kauffman Publishers, San Mateo (1993)
5. Srinivas, N., Deb, K.: Multi-objective optimization using non-dominated sorting in genetic algorithms. Evol. Comput. **2**(3), 221–248 (1994)
6. Horn, J., Nafpliotis, N., Goldberg, D.E.: A niched Pareto genetic algorithm for multi-objective optimization. In: Fogarty, T.C., (ed.) Proceedings of the 1st IEEE Congress on Evolutionary Computation, pp. 82–87. IEEE, Piscataway (1994)
7. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: improving the strength Pareto evolutionary algorithm. In: Giannakoglou, K., Tsahalis, D.T., Périaux, J., Papailiou, K.D., Fogarty, T., (eds.) Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems, pp. 95–100. Springer, Berlin (2002)
8. Knowles, J.D., Corne, D.W.: Approximating the non-dominated front using the Pareto archived evolution strategy. Evol. Comput. **8**(2), 149–172 (2000)

9. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multi-objective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
10. Moore, J., Chapman, R.: Application of particle swarm to multi-objective optimization. In: International Conference on Computer Science and Software Engineering (2003)
11. Ray, T., Liew, K.M.: A swarm metaphor for multi-objective design optimization. Eng. Optim. **34**(2), 141–153 (2002)
12. Coello, C.C.A., Pulido, G.T., Lechuga, M.S.: Handing multiple objectives with particle swarm optimization. IEEE Trans. Evol. Comput. **8**(3), 256–279 (2004)
13. Coello, C.C.A., Cortes, N.C.: Solving multi-objective optimization problems using an artificial immune system. Genet. Program. Evolv. Mach. **6**(2), 163–190 (2005). https://doi.org/10.1007/s10710-005-6164-x
14. Luh, G.C., Chueh, C.H., Liu, W.: MOIA: multi-objective immune algorithm. Eng. Optim. **35** (2), 143–164 (2003)
15. Khan, N., Goldberg, D.E., Pelikan, M.: Multi-objective Bayesian optimization algorithm. In: Proceedings of the Genetic and Evolutionary Computation Conference, p. 684. Morgan Kaufmann, New York (2002)
16. Laumanns, M., Ocenasek, J.: Bayesian optimization algorithms for multi-objective optimization. In: Guervós, J.J.M., Adamidis, P., Beyer, H.-G., Schwefel, H.-P., Fernández-Villacañas, J.-L. (eds.) PPSN 2002. LNCS, vol. 2439, pp. 298–307. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45712-7_29
17. Cai, Z., Wang, Y.: A multi-objective optimization based evolutionary algorithm for constrained optimization. IEEE Trans. Evol. Comput. **10**(6), 658–675 (2006)
18. Li, X.: A non-dominated sorting particle swarm optimizer for multiobjective optimization. In: Cantú-Paz, E., et al. (eds.) GECCO 2003. LNCS, vol. 2723, pp. 37–48. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-45105-6_4
19. Jiao, L., Gong, M., Shang, R., Du, H., Lu, B.: Clonal Selection with Immune Dominance and Anergy Based Multiobjective Optimization. In: Coello Coello, Carlos A., Hernández Aguirre, A., Zitzler, E. (eds.) EMO 2005. LNCS, vol. 3410, pp. 474–489. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-31880-4_33
20. Gong, M.G., Jiao, L.C., Du, H.F., et al.: Multi-objective immune algorithm with non-dominated neighbor-based selection. Evol. Comput. **16**(2), 225–255 (2008)
21. Zhang, Q.F., Zhou, A.M., Jin, Y.: RM-MEDA: a regularity model based multi-objective estimation of distribution algorithm. IEEE Trans. Evol. Comput. **12**(1), 41–63 (2007)
22. Liu, J.: Research on Organizational Coevolutionary Algorithm and its Applications. Ph.D. thesis. Xidian University Xi'an (2004)
23. Tan, K.C., Yang, Y.J., Goh, C.K.: A distributed cooperative evolutionary algorithm for multi-objective optimization. IEEE Trans. Evol. Comput. **10**(5), 527–549 (2006)
24. Tan, Y., Zhu, Y.: Fireworks algorithm for optimization. In: Tan, Y., Shi, Y., Tan, K.C. (eds.) ICSI 2010. LNCS, vol. 6145, pp. 355–364. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13495-1_44
25. Zheng, Y.J., Song, Q., Chen, S.Y.: Multi-objective fireworks optimization for variable-rate fertilization in oil crop production. Appl. Soft Comput. **13**(11), 4253–4263 (2013)
26. Xie, C., Xu, L., Xia, X., Wei, B., et al.: Multi-objective fireworks optimization algorithm using elite opposition-based learning. Acta Electronica Sinica **44**(5), 1180–1188 (2016)