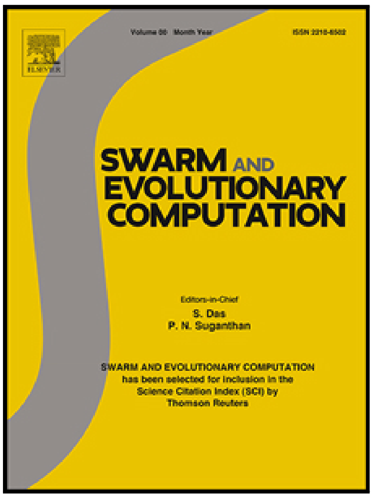

Volume 00 Month Year
ISSN 2210-6502
ELSEVIER
SWARM AND EVOLUTIONARY COMPUTATION
Editors-in-Chief
S. Das
P. N. Suganthan
SWARM AND EVOLUTIONARY COMPUTATION
has been selected for inclusion in the
Science Citation Index (SCI) by
Thomson Reuters

A Clustering-Based Differential Evolution Algorithm for Solving Multimodal Multi-Objective Optimization Problems

Jing Liang , Kangjia Qiao , Caitong Yue , Kunjie Yu , Boyang Qu ,
Ruohao Xu , Zhimeng Li , Yi Hu

PII: S2210-6502(20)30441-7
DOI: https://doi.org/10.1016/j.swevo.2020.100788
Reference: SWEVO 100788

To appear in: *Swarm and Evolutionary Computation*

Received date: 14 October 2019
Revised date: 12 August 2020
Accepted date: 27 September 2020

Please cite this article as: Jing Liang , Kangjia Qiao , Caitong Yue , Kunjie Yu , Boyang Qu , Ruohao Xu , Zhimeng Li , Yi Hu , A Clustering-Based Differential Evolution Algorithm for Solving Multimodal Multi-Objective Optimization Problems, *Swarm and Evolutionary Computation* (2020), doi: https://doi.org/10.1016/j.swevo.2020.100788

A Clustering-Based Differential Evolution Algorithm for Solving Multimodal Multi-Objective Optimization Problems

Jing Liang[a], Kangjia Qiao[a], Caitong Yue[a], Kunjie Yu[a], Boyang Qu [b,*], Ruohao Xu[a], Zhimeng Li[a], Yi Hu[a]

[a]*School of Electrical Engineering, Zhengzhou University, Zhengzhou 450001, China*

[b]*School of Electronic and Information Engineering, Zhongyuan University of Technology, Zhengzhou 450007, China*

[*]Corresponding author: Boyang Qu

Email addresses: liangjing@zzu.edu.cn (Jing Liang), qiaokangjia@yeah.net (Kangjia Qiao), zzuyuecaitong@163.com (Caitong Yue), yukunjie@zzu.edu.cn (Kunjie Yu), qby1984@hotmail.com (Boyang Qu), zzuxrh@163.com (Ruohao Xu), mengzhili@gs.zzu.edu.cn (Zhimeng Li), eehuyi@163.com (Yi Hu).

Abstract—Multimodal Multi-objective Optimization Problems (MMOPs) refer to the problems that have multiple Pareto-optimal solution sets in decision space corresponding to the same or similar Pareto-optimal front in objective space. These problems require the optimization algorithm to locate multiple Pareto Sets (PSs). This paper proposes a differential evolution algorithm based on the clustering technique and an elite selection mechanism to solve MMOPs. In this algorithm, a Clustering-based Special Crowding Distance (CSCD) method is designed to calculate the comprehensive crowding degree in decision and objective spaces. Subsequently, a distance-based elite selection mechanism (DBESM) is introduced to determine the learning exemplars of various individuals. New individuals are generated around the exemplars to obtain a well-distributed population in both decision and objective spaces. To test the performance of the proposed algorithm, extensive experiments on the suit of CEC'2019 benchmark functions have been conducted. The results indicate that the proposed method has superior performance compared with other commonly used algorithms.

Keywords—Multimodal multi-objective optimization problem, differential evolution, clustering, elite selection mechanism.

Author Statement

Jing Liang: Conceptualization, Investigation, Writing- Reviewing and Editing;
Kangjia Qiao: Software, Methodology, Writing- Original draft preparation;
Caitong Yue: Software, Visualization;

Kunjie Yu: Validation, Visualization;
Boyang Qu: Investigation, Methodology, Writing- Reviewing and Editing;
Ruohao Xu: Software, Validation;
Zhimeng Li: Methodology, Data curation;
Yi Hu: Validation, Visualization;

Declaration of Competing Interest

None.

1. Introduction

Multi-objective optimization aims to find a set of non-dominated solutions among multiple conflicting objectives [1-3]. The non-dominated solutions in the decision space are called the Pareto Set (PS) and its mapping vector in the objective space is called the Pareto Front (PF). When dealing with multi-objective optimization problems (MOPs), multi-objective evolutionary algorithms (MOEAs) are commonly used due to their ability of finding multiple solutions in one single run. However, most MOEAs only focus on finding PF in objective space, while the information in decision space is generally ignored. These algorithms perform poorly on handling multimodal multi-objective optimization problems (MMOPs) [4], which refer to the problems that have multiple PSs in decision space corresponding to the same or similar PF in objective space as shown in Figure 1. It is not easy to solve MMOPs as they require the optimization algorithm to find multiple PSs.

To solve MMOPs, various niching techniques [5-7] have been incorporated into MOEAs in the past few years [8-10]. Niching techniques were inspired by the way that organisms evolve in nature and they were originally introduced to solve single-objective multi-modal problems. These techniques aim to form stable niches to prevent the solutions from converging to a local area. Crowding [11, 12], fitness share [13, 14], clustering [15, 16], and clearing [17, 18] are commonly used niching techniques for multimodal problems and they can maintain the diversity of the population in decision space. However, the distributions of the population in both decision space and objective space need to be improved when solving MMOPs. To this end, a **M**ultimodal **M**ulti-**O**bjective **D**ifferential **E**volution algorithm using the **C**lustering-based **S**pecial **C**rowding **D**istance method and elite selection mechanism (MMODE_CSCD) is proposed in this paper. In this algorithm, the clustering-based special crowding distance (CSCD) is embedded into the non-dominated sorting scheme [1] to obtain well-distributed solutions in both decision space and objective space. The non-dominated sorting scheme first divides the population into multiple non-dominated levels, and all solutions corresponding to the same level are grouped into multiple classes by k-means [19] clustering algorithm in decision space. In this manner, the individuals in the same class come from the same PS. The crowding distances are calculated and assigned to each individual by using the neighborhood relationships within the same class. Moreover, a distance-based elite selection mechanism (DBESM) is designed to improve the distribution of the population on all PSs. The main contributions of this work are summarized as follows:

(1) A clustering-based special crowding distance method is proposed to seek the neighborhood relationship and measure the comprehensive crowding distance of each solution in both decision and objective spaces. The crowding distance is regarded as an indicator in the process of exemplar selection and environmental selection.

(2) A new distance-based elite selection mechanism which comprehensively considers the diversity and convergence of the population is introduced to determine the learning exemplar for each individual.

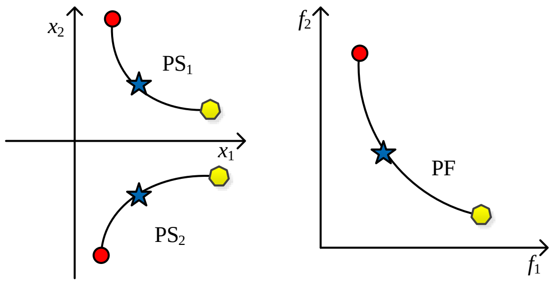


Figure 1: Multimodal multi-objective optimization problem.

The rest of this paper is arranged as follows. Section 2 reviews the basic DE and related works. The proposed algorithm is introduced in Section 3. Section 4 presents an empirical evaluation of MMODE_CSCD in comparison with several state-of-the-art algorithms. Detailed summary and future works are given in Section 5.

2. Background

2.1 Differential evolution

Differential evolution (DE) [20] is a popular population-based optimization algorithm because of its simple structure and high searching efficiency. Mutation, crossover, and selection are the three main operators in the evolution process of DE. To be specific, at the generation G = 0, an initial population consisting of NP individuals are generated randomly in the search domain, and each individual is called target individual and denoted by $\mathbf{x}_i^0 = (x_{i,1}^0, x_{i,2}^0, \cdots, x_{i,D}^0)$, i = 1, 2, …, NP, where D indicates the dimension of the problem.

At each generation G, the mutation operator is performed on each target individual $\mathbf{x}_i^G$ to generate the mutation vector $\mathbf{v}_i^G$. The basic mutation strategy DE/rand/1 is shown as follows:

$$\mathbf{v}_i^G = \mathbf{x}_{r1}^G + FF_i \cdot (\mathbf{x}_{r2}^G - \mathbf{x}_{r3}^G) \tag{1}$$

where r_1, r_2, and r_3 are mutually different integers ranging from 1 to NP, FF is the scaling factor.

After mutation, the crossover operator is conducted between $\mathbf{x}_i^G$ and $\mathbf{v}_i^G$ to generate one trial vector $\mathbf{u}_i^G$:

$$u_{i,j}^G = \begin{cases} v_{i,j}^G, & \text{if } rand_j \le Cr \text{ or } j = k \\ x_{i,j}^G, & \text{otherwise} \end{cases} \tag{2}$$

where $j = (1, 2, ..., D)$, and $rand_j \in [0,1]$. Cr indicates the crossover rate, and k is a random integer uniformly generated from 1 to D.

Finally, the selection operator is implemented between $\mathbf{x}_i^G$ and $\mathbf{u}_i^G$. The better one is selected to survive into the next generation:

$$\mathbf{x}_i^{G+1} = \begin{cases} \mathbf{u}_i^G, & \text{if } f(\mathbf{u}_i^G) \leqslant f(\mathbf{x}_i^G) \\ \mathbf{x}_i^G, & \text{otherwise} \end{cases} \tag{3}$$

where f is the objective function to be minimized.

2.2 Existing multimodal multi-objective optimization algorithms

Recently, many researchers have paid attention to the diversity in decision space in MOPs. Mike et al. [21] extended an existing CMA-EA niching framework to the multi-criterion domain to enhance decision space diversity. Deb et al. [22] proposed the Omni-optimizer algorithm, where crowding distance (CD) was adopted in decision space to maintain the decision space diversity. Generally, high decision space diversity means a good distribution in decision space. However, a well-distributed population in decision space does not ensure a good diversity in objective space. To solve this problem, some other techniques should be added. In [23], a Lebesgue contribution and neighborhood count schemes were adopted to maintain diversity in both decision and objective spaces. The disadvantage of this method is the computing time is very high, because the distances between the solution and each of its neighbors are needed to compute. Zhou et al. [24] proposed a novel MOEA based on a probabilistic model to estimate the distribution of PSs in decision space. This model helps to approximate the PS and the PF simultaneously. However, for the problem with a linear manifold PS, its performance is poor. Although the above research works paid attention to diversity in decision space and objective space, but the problems they solved were not termed as MMOPs.

Liang et al. [4] defined multimodal multi-objective optimization problems (MMOPs) and proposed a corresponding optimizer named DN_NSGAII. In this algorithm, the crowding method was used to create the mating pool to improve decision space diversity. Furthermore, the CD method was employed in decision space as an indicator to maintain multiple PSs. The experimental results show DN_NSGAII can save more Pareto-optimal solutions than various other MOEAs. However, the CD method is only used in decision space, which ignores the performance in the objective space. To overcome the shortcoming, Yue et al. [25] proposed the multiobjective PSO using ring topology and special crowding distance (MO_Ring_PSO_SCD). In this method, ring topology was adopted to form stable niching for maintaining a high population diversity. Furthermore, a new special CD (SCD) method is devised to reflect the crowding degrees in both decision space and objective space. Subsequently, Wang et al. [26] proposed a similar method named multi-objective competitive swarm optimization (CSO) algorithm based on the SCD method and ring topology (MO_Ring_CSO_SCD). It employed CSO to improve the performance of PSO. However, its performance on the problems with irregular PSs needs to be improved.

Since SCD provides an effective indicator for environmental selection, it is widely used in solving MMOPs. For example, Xu et al. [27] embedded a decision-variable preselection scheme into the DE algorithm to increase the diversity of solutions in both decision space and objective space. In addition, a new mutation-bound process was designed to reduce the number of individuals on the edge of the search space. Pal et al. [28] proposed a multimodal multi-objective DE algorithm based on the mating pool selection strategy and resource allocation scheme. In [29], the self-organizing map (SOM) network was combined with the SCD method to locate and maintain multiple PSs. The SOM network can obtain a reasonable neighbor structure to improve population diversity. In the same way, Hu et al. [30] introduced the SOM network and SCD method into the pigeon-inspired optimization to solve MMOPs. However, the high computational complexity of the SOM seriously affects the efficiency of the algorithms. To reduce the complexity of MMOPs, a novel zoning search was developed by dividing the decision space into multiple subspaces [31]. However, it may perform poorly on high-dimension problems. Zhang et al. [32] devised a cluster-based PSO (MMO-CLRPSO), which first employed the clustering niching method to divide the population into multiple subpopulations. Then ring topology was used to improve the information exchange among different subpopulations. Qu et al. [33] proposed a novel algorithm using the speciation niching technology and a self-organized method to improve population diversity and locate multiple PSs. However, the parameters in clustering and speciation niching methods are difficult to be determined. Thus the performance of these two algorithms on some specific types of problems is poor.

Additionally, some other techniques instead of the SCD method are also designed to find multiple PSs. Shang et al. [34] designed a novel DE algorithm based on reinforcement learning (RL) with fitness ranking (DE_RLFR). The designed reward function was used to guide the evolve of the population to find different PSs. The experimental results demonstrate that DE_RLFR has strong robustness. However, the reward function depends on the property of the problems. Liu et al. [35] employed the clustering niching technology and the clearing niching method to maintain diversity in objective space and decision space, respectively. This algorithm can get a large number of optimal solutions according to the proposed recombination strategy, even though the population size is small.

Although there are already several different MMO algorithms, they have some common disadvantages to be overcome. 1) Most of them use the SCD method, however, SCD cannot find reasonable neighbors in some special situations. 2) The distribution on different PSs should be improved. To overcome these problems, this study develops the MMODE_CSCD method, which has two main schemes: clustering-based special crowding distance (CSCD) and distance-based elite selection mechanism (DBESM). CSCD can find a correct neighbor relationship. DBESM can maintain population diversity and improve the uniformity of population distribution over all PSs.

3. The proposed method

In this section, the detailed descriptions of the CSCD and DBESM are explained, and then the main framework of MMODE_CSCD is presented.

3.1 The clustering-based special crowding distance

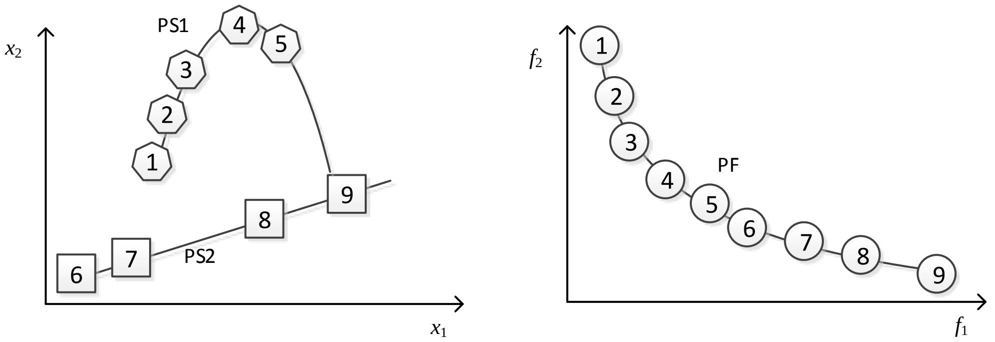
Special crowding distance (SCD) is a widely used method for MMOPs since it provides an indicator to sort the individuals in the same non-dominated level. In the following part, the SCD method and its defects are first introduced, and then how CSCD deals with this problem is explained.

Table 1: Nomenclature.

symbol	variable
$x_{i,j}$	the jth dimension of individual i
$\mathrm{CD}_{i,x}$	the decision space crowding distance of individual i
$\mathrm{CD}_{i,f}$	the objective space crowding distance of individual i
SCD_i	the comprehensive crowding distance of individual i in SCD method
CSCD_i	the comprehensive crowding distance of individual i in CSCD method
$\mathrm{CD}_{\mathrm{avg},x}$	the average decision space crowding distance of all individuals
$\mathrm{CD}_{\mathrm{avg},f}$	the average objective space crowding distance of all individuals

To describe the steps of the SCD method, a simple case of nine solutions is provided in Figure 2. Figure 2 (a) shows the distribution in decision space and Figure 2 (b) shows the distribution in objective space. In the decision space, these solutions belong to two PSs that are represented by PS1 and PS2, respectively. Solution 8 is regarded as one example to explain how to calculate SCD_i. First, these solutions are sorted according to each decision variable. Then two order lists (6, 7, 1, 2, 3, 4, 8, 5, 9) and (6, 7, 8, 9, 1, 2, 3, 5, 4) are obtained. According to these lists, the neighbors of solution 8 on x_1 are solutions 4 and 5, and the neighbors on x_2 are solutions 7 and 9. In addition, solutions 6 and 9 are the boundary solutions on x_1, and solutions 6 and 4 are the boundary solutions on x_2. Based on the above information, $\mathrm{CD}_{8,x}$ can be calculated by Eq. (4):

$$\mathrm{CD}_{8,x} = \left(\left|x_{4,1}-x_{5,1}\right|/\left|x_{6,1}-x_{9,1}\right|\right)+\left(\left|x_{7,2}-x_{9,2}\right|/\left|x_{6,2}-x_{4,2}\right|\right) \tag{4}$$



(a) Decision space (b) Objective space

Figure 2: The distribution of the one set of solutions in (a) decision space and (b) objective space.

In objective space, these solutions also need to be sorted on each objective dimension. Their orders on two objective variables f_1 and f_2 are the same, that is (1, 2, 3, 4, 5, 6, 7, 8, 9). Thus $\mathrm{CD}_{8,f}$ can be computed as follows:

$$\mathrm{CD}_{8,f} = \left(\left|x_{7,1}-x_{9.1}\right|/\left|x_{1,1}-x_{9,1}\right|\right)+\left(\left|x_{7,2}-x_{9,2}\right|/\left|x_{1,2}-x_{9,2}\right|\right) \tag{5}$$

Then, SCD_8 can be obtained using Eq. (6).

$$\mathrm{SCD}_8 = \begin{cases} \max(\mathrm{CD}_{8,x}, \mathrm{CD}_{8,f}), & \text{if } \mathrm{CD}_{8,x} > \mathrm{CD}_{avg,x} \text{ or } \mathrm{CD}_{8,f} > \mathrm{CD}_{avg,f} \\ \min(\mathrm{CD}_{8,x}, \mathrm{CD}_{8,f}), & \text{otherwise} \end{cases} \tag{6}$$

In the above situation, there are two kinds of defects in the SCD method. 1) The neighbors of one solution may be identified from other PSs. For example, solutions 4 and 5 are used as the neighbors of solution 8 on x_1, but these three solutions do not belong to the same PS. In fact, solutions 7 and 9 are the neighbors of solution 8 since all these three solutions belong to PS2. When solutions 7 and 9 are used as the neighbors of solution 8 on x_1, a larger $\mathrm{CD}_{8,x}$ will be obtained. It is consistent with the observation that the density at solution 8 is relatively low. 2) The area between one solution and its neighbor may not near the real PS. For example, the neighbors of solution 3 on x_2 are solutions 2 and 5 in the original SCD method. But, the area between solutions 3 and 4 is a promising area in which more solutions should be generated. Thus, the true neighbors of solution 3 are solutions 2 and 4.

Generally, without prior knowledge, it is difficult to know which individuals belong to the same PS. However, based on the above analysis, it can be known that an individual's neighbors should come from its surroundings. Therefore, the clustering algorithm can be used to divide the solutions in the same non-dominated level into multiple classes, and the individuals in the same class will come from the same local region. Based on this motivation, the clustering algorithm is combined with SCD to form a CSCD method. Among various clustering methods, the k-means algorithm is popular, thus it is employed in this study. Other clustering methods will be investigated in our future works to verify the impacts of different clustering methods on MMOPs.

In CSCD, an individual's neighbors can only be identified from the same class, thus the accuracy of the measured CD can be improved. The calculation way of CSCD is illustrated using the case in

Figure 2. Supposing that the nine solutions in Figure 2 (a) are divided into three classes, i.e., the first class (6, 7, 8, 9), the second class (1, 2, 3, 4), and the third class (5), the CSCD_8 can be computed based on Eqs. (7) - (9).

$$CD_{8,x} = \left(\left|x_{7,1} - x_{9,1}\right| / \left|x_{6,1} - x_{9,1}\right|\right) + \left(\left|x_{7,2} - x_{9,2}\right| / \left|x_{6,2} - x_{9,2}\right|\right) \tag{7}$$

$$CD_{8,f} = \left(\left|x_{7,1} - x_{9,1}\right| / \left|x_{1,1} - x_{9,1}\right|\right) + \left(\left|x_{7,2} - x_{9,2}\right| / \left|x_{1,2} - x_{9,2}\right|\right) \tag{8}$$

$$CSCD_8 = \begin{cases} \max(CD_{8,x}, CD_{8,f}), & \text{if } CD_{8,x} > CD_{avg,x} \text{ or } CD_{8,f} > CD_{avg,f} \\ \min(CD_{8,x}, CD_{8,f}), & \text{otherwise} \end{cases} \tag{9}$$

Since the CSCD method is implemented on each non-dominated level, which is obtained by the non-dominated sorting scheme, the total implementation procedure including non-dominated sorting scheme and CSCD method, is called Non-dominated_CSCD_sorting method whose pseudo-code is provided in **Algorithm 1**. In the non-dominated sorting scheme, the population is divided into multiple non-dominated levels ($\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_{num}$), where *num* is the number of non-dominated levels. After the non-dominated sorting, the first indicator, i.e., the non-dominated ranking of each individual, is obtained. Then, each $\mathbf{F}_j$ is grouped into multiple classes by the k-means algorithm. Within each class, the CSCD value of each solution is calculated according to Eqs. (7) - (9). CSCD is regarded as the second indicator. Then, the solutions with the same non-dominated rankings are sorted based on their CSCD values in descending order. Finally, the new population after sorting based on these two indicators is output.

Specially, it is necessary to explain how to set a *k* value for each $\mathbf{F}_j$. Since | $\mathbf{F}_j$ | (the number of solutions in $\mathbf{F}_j$, $j = (1, 2, \ldots, num)$) is changing in each generation, k_j should be changing accordingly. Therefore, a new parameter *n* is introduced. As shown in line 3 of **Algorithm** 1, k_j keeps a linear relationship with | $\mathbf{F}_j$ |. It is worth noting that the effect of different *n* values is investigated in Section 4.4.

```
Algorithm 1: Non-dominated_CSCD_sorting
Input: the population: $P$.
Output: a sorted population: $P^*$.
1. Carry out non-dominated sorting on $P$ to obtain multiple non-dominated levels which are
   defined $\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_{num}$ ($num$ is the number of levels), and $j = (1, 2, \ldots, num)$ indicate the
   non-dominated ranking of the individuals in $\mathbf{F}_j$;
2. For $j = 1 : num$
3.     Compute the value of $k_j$ used on $\mathbf{F}_j$: $k_j = \left\lceil \frac{|\mathbf{F}_j|}{n} \right\rceil$, where | $\mathbf{F}_j$ | indicates the number of
       individuals in $\mathbf{F}_j$ and $n$ is a positive integer;
4.     Divide the $\mathbf{F}_j$ into multiple classes by k-means algorithm;
5.     Compute the CSCD value of each individual by using the neighborhood relationships within
       the same class;
6. End For
7. Sort the population $P$ based on their non-dominated rankings and CSCD values to obtain a new
   population $P^*$;
```

Algorithm 2: DBESM

```
Input: the sorted population: $P^*$;
       the number of $\mathbf{F}$s: $num$;
       the number of solutions in each $\mathbf{F}$: $\{|\mathbf{F}_1|, |\mathbf{F}_2|, \ldots, |\mathbf{F}_{num}|\}$.
Output: the exemplar of each solution.
For $l = 1 : num$
  For $i = 1 : |\mathbf{F}_l|$
    Determine the target level $\mathbf{F}a$ by randomly selecting a number from the set $\{1, 2, \ldots, l\text{-}1\}$ (if $l$ = 1, $a$ is set as 1);
    Select the top 100*$p$% solutions with largest CSCD values in $\mathbf{F}_a$ as the candidate exemplars;
    Compute the distances between $x_i^l$ and all candidate exemplars as the distance indicator;
    Perform the roulette wheel selection to determine the final exemplar from the candidate exemplars based on the distance indicator;
  End For
End For
```

3.2 The distance-based elite selection mechanism

After performing the Non-dominated_CSCD_sorting, a sorted population with non-dominated ranking and CSCD value are obtained. In DBESM, the exemplar of each solution is determined by non-dominated ranking, CSCD, and a distance indicator in a sequential manner. The Pseudo-code of DBESM is illustrated in **Algorithm 2**.

DBESM contains three main steps. First, for each x_i^l (x_i^l represents the ith solution in lth level), the target level (denoted as $\mathbf{F}_a$) is determined by randomly selecting a number from the set {1, 2, …, l-1}. Each individual in $\mathbf{F}_a$ may become the exemplar of x_i^l. When $a < l$, the excellence of exemplar is guaranteed. Specially, if l = 1, a will be set as 1 since there is no better **F**. Second, the top 100*p% (p = 0.1 in this paper [36]) solutions with the largest CSCD values in $\mathbf{F}_a$ are regarded as the candidate exemplars. These candidate exemplars are used to provide some promising evolutionary directions. Third, the final exemplar is determined based on the distance indicator. The distance indicator refers to the distances between x_i^l and all candidate exemplars. Therefore, the candidate exemplar near x_i^l becomes the final exemplar with high probability. Then, the reason why the distance indicator is used is described as follows. If there are a few individuals belonging to the $PS_{x_i^l}$ (the PS that x_i^l locates on), x_i^l should choose the exemplar from $PS_{x_i^l}$ preferentially to generate offspring on $PS_{x_i^l}$. Thus, the population can distribute evenly on all PSs. The selected exemplar is used in the following mutation strategy.

A new mutation strategy DE/current-to-exemplar/1 is formed by modifying one popular mutation strategy DE/current-to-*p*best/1 strategy [36]. DE/current-to-*p*best/1 provides a better trade-off between diversity and convergence. However, it cannot be directly used to solve multi-objective optimization problems, because the "*p*best" individuals are not easy to be determined in a multi-objective situation. According to the property of MMOPs, DBESM can determine the learning exemplar of each individual. Meanwhile, the new mutation strategy keeps the advantage of DE/current-to-*p*best/1 in diversity and convergence. The DE/current-to-exemplar/1 strategy is formulated as follows:

$$\mathbf{v}_i^G = \mathbf{x}_i^G + FF_i \cdot \left(\mathbf{x}_{\text{exemplar}}^G - \mathbf{x}_i^G\right) + FF_i \cdot \left(\mathbf{x}_{r1}^G - \mathbf{x}_{r2}^G\right) \tag{10}$$

3.3 The framework of MMODE_CSCD

The framework of MMODE_CSCD is described in **Algorithm 3**. An initial population *P* with *NP* individuals is generated in the search space, and each individual is evaluated on all objectives. Next, Non-dominated_CSCD_sorting is carried out to obtain a sorted population P^*, the non-dominated ranking, and CSCD value of each individual. Then, DBESM is performed on P^* to determine the exemplar for each individual. Afterward, the mutation and crossover operators are executed to generate the offspring population *OP*. The temporary population *POP* is obtained by merging *P* and *OP*. Then, *POP* is sorted by Non-dominated_CSCD_sorting to obtain the population POP^*. Subsequently, a new population *P* is selected from POP^* through environmental selection. Figure 3 presents the process of the environmental selection method. Finally, if $G < G_{max}$, the loop will continue, otherwise, the solutions belong to $\mathbf{F}_1$ are output.

Algorithm 3: Framework of MMODE_CSCD

```
Input: population size: $NP$;
       maximum number of generations: $G_{max}$;
       scaling factor: $FF$;
       crossover rate: $Cr$.
Output: PF and PSs.
$G = 1$;
Generate the initial population $P$ with $NP$ individuals and evaluate all individuals on each objective;
While $G < G_{max}$
    Perform the Non-dominated_CSCD_sorting on $P$ to obtain a sorted population $P^*$; → Algorithm 1
    Implement the DBESM on $P^*$ to select an exemplar for each individual; → Algorithm 2
    Perform mutation based on Eq. (10) and crossover based on Eq. (2) on $P^*$ to obtain an offspring population $OP$;
    Execute the Non-dominated_CSCD_sorting on the temporary population $POP$ by merging $P$ and $OP$ to obtain a sorted population $POP^*$; → Algorithm 1
    Perform the environmental selection on $POP^*$ to obtain a new population $P$;
    $G = G + 1$;
End while
```

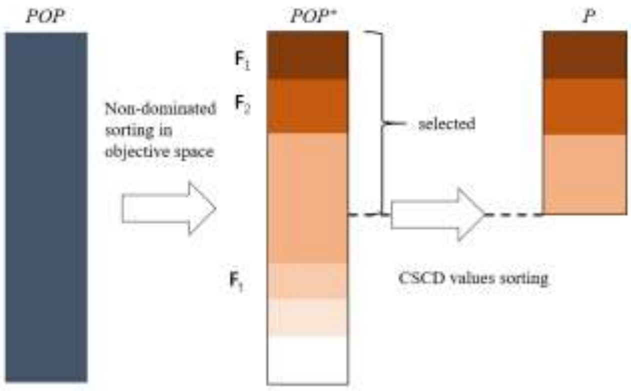


Figure 3: Environmental selection.

4. Experiments

4.1 Experimental setups

In this section, the effectiveness and the superiority of MMODE_CSCD are verified on the CEC’2019 multimodal multi-objective optimization benchmark suite [37]. The benchmark suite contains 22 test functions with different characteristics, thus they can verify the performance of the proposed method systematically. In addition, this benchmark suite provides four performance indicators: the reciprocal of Pareto Sets Proximity (1/PSP, rPSP [38]), the reciprocal of Hypervolume (1/HV, rHV [37]), Inverted Generational Distance (IGD [24]) in decision space (IGDx), and IGD in objective space (IGDf). rPSP and rHV are used instead of PSP and HV so that for all four indicators the smaller value means better performance. Among these four indicators, rPSP and IGDx are used to evaluate the distribution of the population in decision space, while rHV and IGDf are used to evaluate the distribution of the population in objective space. Specially, rPSP is a new indicator proposed recently in [38] to reflect not only the overlap rate between obtained PSs and the true PSs but also the diversity and convergence of the obtained solutions.

The compared algorithms include DN_NSGAII [4], MO_Ring_PSO_SCD [25], MMOPIO [30], MO_PSO_MM [29], Omni_optimizer [22], DE_RLFR [34], and TriMOEATAR [35]. The parameter settings of these compared algorithms are consistent with the original literature. For the two parameters of DE in MMODE_CSCD, *FF* = 0.8, *Cr* = 1. In addition, for the k-means algorithm in this paper, *n* is set as 10 to obtain the *k* value. Besides, for all algorithms, *NP* = 200, G_{max} = 100, and 21 independent runs are implemented. Furthermore, to observe the difference between MMODE_CSCD and the compared algorithms, the rank-sum test with α = 0.05 is adopted. When MMODE_CSCD performs significantly better or worse than the compared algorithm, + or – is marked. When there is no obvious difference between the two competitors, ≈ is marked.

Note that the proposed algorithm and other algorithms are all designed for finding global PSs, so only global PSs in the test functions are used to calculate the performance indicators.

4.2 Comparison results against other algorithms

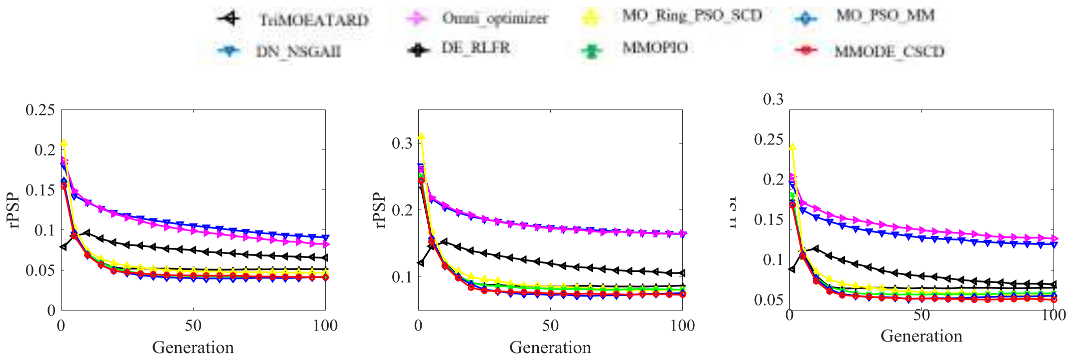
Tables 2-5 record the statistical comparison results regarding the mean value (mean) and standard deviation (std) of rPSP, rHV, IGDx, and IGDf obtained by all algorithms. IGDx is used to reflect the convergence between obtained PSs and true PSs. IGDf is used to reflect the convergence between obtained PFs and true PFs. Furthermore, rPSP can reflect both the convergence and overlap ratio between real PSs and obtained PSs. rHV can reflect the convergence and diversity of obtained PF. Although these four indicators have different roles, rPSP and rHV have more comprehensive characteristics than IGDx and IGDf. Therefore, the performance of all algorithms on rPSP and rHV is discussed. In addition, the Friedman test results [39] are given in Table 6, which provides the ranking on each indicator and comprehensive ranking on four indicators.

From Table 2, it can be obtained that MMODE_CSCD gets the smallest mean rPSP values on 11 functions, and the second smallest mean rPSP values on 6 functions. MO_PSO_MM ranks second and third on most of the functions. MMOPIO performs similarly with MO_PSO_MM since both of them adopt the SOM network to identify the neighbor structure. MO_Ring_PSO_SCD does not use the SOM neighbor relationship, thus its performance on most of the functions is worse than MO_PSO_MM and MMOPIO. TriMOEATAR ranks the first on 7 functions, however, it performs inferiorly on other functions. DE_RLFR only gets better results on Omni-test since the performance of DE_RLFR highly depends on the reward function. Omni_optimizer and DN_NSGAII perform poorly on all functions. This is because both of them use the CD method only in the decision space. Specially, for MMF14, MMF15, MMF14_a, and MMF15_a that have multiple PFs, the performance of MMODE_CSCD is not outstanding. The reason is that it does not attempt to find the neighbor relationship in objective space. Moreover, Table 6 indicates that MMODE_CSCD gets the first ranking on rPSP, followed by MO_PSO_MM, MMOPIO, MO_Ring_PSO_SCD, TriMOEATAR, DE_RLFR, Omni-optimizer, and DN_NSGAII. The above observations fully demonstrate the superiority of MMODE_CSCD in decision space.

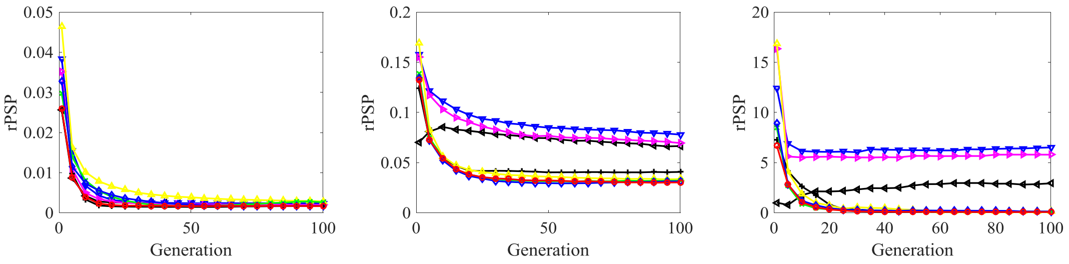
In Table 3, the rHV values obtained by all algorithms are recorded. MMODE_CSCD obtains the first ranking and the second ranking on 9 and 5 functions, respectively. However, its performance on the functions with multiple PFs is weak. Besides, TriMOEATAR obtains the best mean rHV values on 6 simple linear functions and 2 complex linear functions. MO_PSO_MM and MMOPIO also have similar performance in objective space. Omni_optimizer achieves the second and third rankings on most of the functions. In addition, the performance of DN_NSGAII, MO_Ring_PSO_SCD, and DE_RLFR are inferior. In fact, all algorithms consider the distribution of solution in objective space, thus their rHV values are close to each other. In summary, MMODE_CSCD still achieves the best performance in objective space.

Moreover, as shown in Table 6, the Friedman test results show MMODE_CSCD obtains the best result on each indicator. In terms of all four indicators, the best comprehensive result is still achieved by MMODE_CSCD.

Furthermore, the average rPSP values of MMODE_CSCD and seven comparative algorithms over 21 runs on some functions are plotted in Figure 4. MMODE_CSCD shows faster or similar convergence performance than the other seven algorithms, which demonstrates the superiority of the proposed algorithm.



(a) MMF2 (b) MMF5 (c) MMF6



(d) MMF12 (e) MMF1_z (f) SYM-PART_rotated

Figure 4: The average rPSP values obtained by MMODE_CSCD and seven compared algorithms on six functions.

Table 2: The rPSP values obtained by all algorithms.

	MMODE_CSCD	DN_NSGAII		MO_Ring_PSO_SCD		MMOPIO		MO_PSO_MM		Omni_optimizer		DE_RLFR[1]		TriMOEATAR	
	mean±std	mean±std		mean±std		mean±std		mean±std		mean±std		mean±std		mean±std	
MMF1	**0.0414**±0.0014	0.0882±0.0119	+	0.0467±0.0022	+	0.0419±0.0025	≈	0.0418±0.0025	≈	0.0880±0.0170	+	0.0529±0.0059	+	0.0648±0.0080	+
MMF2	**0.0102**±0.0018	0.1251±0.0776	+	0.0272±0.0108	+	0.0124±0.0036	+	0.0191±0.0059	+	0.1154±0.0670	+	0.0846±0.0505	+	0.0724±0.0457	+
MMF3	**0.0085**±0.0018	0.0827±0.0378	+	0.0188±0.0031	+	0.0118±0.0038	+	0.0137±0.0018	+	0.0925±0.0692	+	0.0566±0.0390	+	0.1100±0.0736	+
MMF4	**0.0223**±0.0011	0.1017±0.0365	+	0.0261±0.0019	+	0.0288±0.0042	+	0.0276±0.0022	+	0.1169±0.0341	+	0.0306±0.0036	+	0.0792±0.1544	+
MMF5	**0.0721**±0.0034	0.1604±0.0157	+	0.0800±0.0039	+	0.0847±0.0070	+	0.0751±0.0036	+	0.1632±0.0163	+	0.0868±0.0091	+	0.1028±0.0100	+
MMF6	**0.0625**±0.0022	0.1388±0.0180	+	0.0683±0.0040	+	0.0722±0.0045	+	0.0679±0.0030	+	0.1442±0.0293	+	0.0754±0.0045	+	0.0862±0.0140	+
MMF7	**0.0221**±0.0014	0.0461±0.0090	+	0.0257±0.0010	+	0.0346±0.0047	+	0.0321±0.0037	+	0.0387±0.0084	+	0.0392±0.0078	+	0.0546±0.0507	+
MMF8	0.0489±0.0031	0.3057±0.0970	+	0.0587±0.0032	+	0.0634±0.0132	+	**0.0480**±0.0033	≈	0.3230±0.1714	+	0.0743±0.0164	+	0.4676±0.1072	+
MMF9	0.0057±0.0003	0.0228±0.0095	+	0.0072±0.0004	+	0.0122±0.0025	+	0.0098±0.0017	+	0.0280±0.0160	+	0.0066±0.0012	≈	**0.0032**±0.0001	-
MMF10	0.0370±0.1063	0.1390±0.1457	+	0.0158±0.0047	-	0.0051±0.0022	-	**0.0019**±0.0003	≈	0.0607±0.0925	+	0.0656±0.1397	+	0.0029±0.0001	-
MMF11	0.0041±0.0003	0.0046±0.0002	+	0.0047±0.0001	+	0.0072±0.0017	+	0.0069±0.0018	+	0.0044±0.0002	+	0.0048±0.0034	+	**0.0037**±0.0001	-
MMF12	**0.0016**±0.0001	0.0044±0.0094	+	0.0030±0.0003	+	0.0022±0.0005	+	0.0017±0.0002	≈	0.0020±0.0001	+	0.0017±0.0004	≈	0.0023±0.0001	+
MMF13	**0.0277**±0.0008	0.0720±0.0130	+	0.0316±0.0011	+	0.0335±0.0022	+	0.0282±0.0013	≈	0.0710±0.0240	+	0.0317±0.0030	+	0.0533±0.0133	+
MMF14	0.0624±0.0016	0.1306±0.0141	+	0.0663±0.0023	+	0.0662±0.0027	+	0.0631±0.0023	≈	0.1154±0.0118	+	–	+	**0.0382**±0.0006	-
MMF15	0.0504±0.0019	0.0848±0.0092	+	0.0505±0.0018	≈	0.0489±0.0026	-	0.0475±0.0018	-	0.0710±0.0059	+	–	+	**0.0385**±0.0006	-
MMF1_z	**0.0288**±0.0009	0.0722±0.0162	+	0.0334±0.0016	+	0.0319±0.0025	+	0.0308±0.0018	+	0.0689±0.0196	+	0.0422±0.0049	+	0.0636±0.0146	+
MMF1_e	0.3617±0.1759	1.6845±1.0459	+	0.4114±0.0978	+	0.6117±0.3614	+	**0.3043**±0.1008	≈	2.3871±1.8144	+	6.4334±4.6594	+	4.8973±2.5963	+
MMF14_a	0.0741±0.0026	0.1452±0.0114	+	0.0747±0.0019	≈	0.0736±0.0025	≈	0.0726±0.0024	≈	0.1442±0.0161	+	–	+	**0.0718**±0.0016	-
MMF15_a	0.0591±0.0029	0.1095±0.0162	+	0.0550±0.0023	-	0.0548±0.0023	-	0.0535±0.0023	-	0.0915±0.0115	+	–	+	**0.0476**±0.0013	-
SYM-PART simple	0.0539±0.0039	5.0409±2.8973	+	0.1107±0.0156	+	0.0708±0.0105	+	0.0727±0.0093	+	5.4645±2.2086	+	0.0707±0.0155	+	**0.0292**±0.0165	-
SYM-PART rotated	**0.1075**±0.2512	7.0084±9.0814	+	0.1309±0.0131	+	0.1215±0.2058	+	0.1257±0.0104	+	6.9498±4.3783	+	0.1684±0.2523	+	1.9208±1.3825	+
Omni-test	0.5636±0.1177	1.7879±0.4169	+	0.3675±0.0820	-	0.6689±0.1403	+	0.3133±0.0788	-	1.9829±0.5164	+	**0.1502**±0.0597	-	0.6523±0.3016	≈
+			22		17		17		11		22		19		13
≈			0		2		2		8		0		2		1
–			0		3		3		5		0		1		8

Table 3: The rHV values obtained by all algorithms.

	MMODE_CSCD	DN_NSGAII		MO_Ring_PSO_SCD		MMOPIO		MO_PSO_MM		Omni_optimizer		DE_RLFR		TriMOEATAR	
	mean±std	mean±std		mean±std		mean±std		mean±std		mean±std		mean±std		mean±std	
MMF1	1.1455±0.0003	1.1490±0.0016	+	1.1476±0.0005	+	1.1481±0.0009	+	1.1473±0.0007	+	1.1473±0.0007	+	1.1477±0.0007	+	**0.9924**±0.4970	-
MMF2	**1.1497**±0.0009	1.1655±0.0209	+	1.1694±0.0045	+	1.1504±0.0013	≈	1.1602±0.0022	+	1.1596±0.0190	≈	1.2002±0.0229	+	1.1731±0.0098	+
MMF3	1.1488±0.0006	1.1574±0.0140	+	1.1617±0.0030	+	1.1495±0.0017	≈	1.1560±0.0011	+	1.1640±0.0301	≈	1.1793±0.0183	+	**1.0572**±0.3842	-
MMF4	1.8529±0.0009	1.8582±0.0017	+	1.8595±0.0022	+	1.8607±0.0039	+	1.8659±0.0049	+	1.8548±0.0005	+	1.8638±0.0094	+	**1.5042**±1.1165	-

[1] DE_RLFR can only address the bi-objective problems.

MMF5	1.1454±0.0003	1.1481±0.0011	+	1.1473±0.0003	+	1.1481±0.0016	+	1.1470±0.0007	+	1.1467±0.0010	+	1.1484±0.0030	+	**0.4526**±2.2123	-
MMF6	1.1457±0.0006	1.1482±0.0010	+	1.1477±0.0007	+	1.1474±0.0006	+	1.1473±0.0007	+	1.1465±0.0005	+	1.1489±0.0041	+	**1.0296**±0.3743	-
MMF7	**1.1454**±0.0002	1.1501±0.0024	+	1.1481±0.0005	+	1.1515±0.0016	+	1.1515±0.0011	+	1.1470±0.0004	+	1.1482±0.0008	+	1.1743±0.0639	+
MMF8	2.3747±0.0018	2.3805±0.0032	+	2.3917±0.0098	+	2.3799±0.0030	+	2.3792±0.0023	+	2.3744±0.0008	≈	2.4169±0.1205	+	**2.0725**±0.9758	-
MMF9	**0.1032**±0.0000	0.1033±0.0000	+	0.1034±0.0000	+	0.1034±0.0001	+	0.1035±0.0001	+	0.1033±0.0000	+	0.1034±0.0001	+	0.1047±0.0002	+
MMF10	0.0781±0.0018	0.0810±0.0034	+	0.0788±0.0003	+	0.0779±0.0004	-	**0.0778**±0.0001	-	0.0792±0.0024	+	0.0790±0.0030	+	0.0787±0.0000	-
MMF11	**0.0689**±0.0000	0.0689±0.0000	+	0.0690±0.0000	+	0.0690±0.0000	+	0.0690±0.0000	+	0.0689±0.0000	+	0.0691±0.0002	+	0.0696±0.0001	+
MMF12	**0.6355**±0.0000	0.6468±0.0402	+	0.6375±0.0009	+	0.6358±0.0002	+	0.6385±0.0002	+	0.6356±0.0000	+	0.6395±0.0052	+	0.6361±0.0000	+
MMF13	**0.0542**±0.0000	0.0543±0.0000	+	0.0544±0.0000	+	0.0543±0.0000	+	0.0543±0.0000	+	0.0543±0.0001	+	0.0543±0.0000	+	0.0550±0.0000	+
MMF14	0.3581±0.0151	0.3350±0.0206	-	0.3624±0.0345	≈	0.3274±0.0179	-	0.3189±0.0280	-	0.3400±0.0109	-	–	+	**0.3153**±0.0156	-
MMF15	0.2443±0.0103	0.2399±0.0138	-	0.2448±0.0150	≈	0.2257±0.0103	-	0.2286±0.0079	-	0.2416±0.0130	-	–	+	**0.2217**±0.0078	-
MMF1_z	**1.1455**±0.0002	1.1479±0.0007	+	1.1475±0.0004	+	1.1471±0.0005	+	1.1469±0.0003	+	1.1467±0.0010	+	1.1480±0.0007	+	1.1485±0.0023	+
MMF1_e	1.1711±0.0109	**1.1206**±0.3944	≈	1.1647±0.0135	≈	1.1488±0.0018	-	1.1526±0.0017	-	1.1571±0.0085	-	1.1519±0.0074	-	1.1520±0.0024	-
MMF14_a	0.3600±0.0199	0.3175±0.0135	-	0.3296±0.0318	≈	**0.3079**±0.0134	-	0.3148±0.0144	-	0.3331±0.0104	-	–	+	0.3432±0.0209	≈
MMF15_a	0.2498±0.0096	0.2358±0.0157	-	0.2396±0.0106	-	**0.2228**±0.0068	-	0.2273±0.0092	-	0.2367±0.0124	-	–	+	0.2296±0.0173	-
SYM-PART simple	**0.0600**±0.0000	0.0601±0.0000	+	0.0602±0.0000	+	0.0601±0.0000	+	0.0601±0.0000	+	0.0601±0.0000	+	0.0601±0.0000	+	0.0602±0.0000	+
SYM-PART rotated	**0.0600**±0.0000	0.0601±0.0000	+	0.0603±0.0000	+	0.0601±0.0000	+	0.0603±0.0000	+	0.0601±0.0000	+	0.0601±0.0000	+	0.0601±0.0000	+
Omni-test	0.0190±0.0000	0.0189±0.0000	-	0.0190±0.0000	+	0.0190±0.0000	-	0.0190±0.0000	+	**0.0189**±0.0000	-	0.0190±0.0000	≈	0.0190±0.0000	+
+			16		17		13		16		13		20		10
≈			1		4		2		0		3		1		1
–			5		1		7		6		6		1		11

Table 4: The IGDx values obtained by all algorithms.

	MMODE_CSCD	DN_NSGAII		MO_Ring_PSO_SCD		MMOPIO		MO_PSO_MM		Omni_optimizer		DE_RLFR		TriMOEATAR	
	mean±std	mean±std		mean±std		mean±std		mean±std		mean±std		mean±std		mean±std	
MMF1	**0.0412**±0.0013	0.0870±0.0114	+	0.0463±0.0022	+	0.0416±0.0025	≈	0.0416±0.0024	≈	0.0855±0.0156	+	0.0524±0.0057	+	0.0637±0.0071	+
MMF2	**0.0102**±0.0018	0.1030±0.0526	+	0.0261±0.0106	+	0.0120±0.0034	≈	0.0185±0.0057	+	0.1025±0.0530	+	0.0717±0.0367	+	0.0656±0.0380	+
MMF3	**0.0085**±0.0018	0.0738±0.0318	+	0.0180±0.0030	+	0.0115±0.0036	+	0.0133±0.0018	+	0.0790±0.0459	+	0.0500±0.0318	+	0.0900±0.0398	+
MMF4	**0.0221**±0.0011	0.1013±0.0359	+	0.0258±0.0019	+	0.0286±0.0041	+	0.0275±0.0022	+	0.1162±0.0339	+	0.0302±0.0035	+	0.0642±0.1068	+
MMF5	**0.0718**±0.0034	0.1594±0.0155	+	0.0795±0.0037	+	0.0841±0.0070	+	0.0749±0.0036	+	0.1602±0.0153	+	0.0860±0.0087	+	0.1016±0.0098	+
MMF6	**0.0622**±0.0022	0.1372±0.0169	+	0.0678±0.0039	+	0.0717±0.0043	+	0.0676±0.0030	+	0.1415±0.0268	+	0.0746±0.0042	+	0.0847±0.0126	+
MMF7	**0.0220**±0.0013	0.0454±0.0085	+	0.0254±0.0010	+	0.0344±0.0046	+	0.0320±0.0036	+	0.0378±0.0079	+	0.0360±0.0056	+	0.0398±0.0223	+
MMF8	0.0484±0.0029	0.3006±0.0967	+	0.0582±0.0031	+	0.0626±0.0121	+	**0.0478**±0.0033	≈	0.3105±0.1594	+	0.0732±0.0157	+	0.4127±0.0819	+
MMF9	0.0057±0.0003	0.0228±0.0095	+	0.0072±0.0004	+	0.0122±0.0025	+	0.0098±0.0017	+	0.0280±0.0160	+	0.0066±0.0012	≈	**0.0032**±0.0001	-
MMF10	0.0368±0.1063	0.1383±0.1461	+	0.0154±0.0042	-	0.0051±0.0022	-	**0.0019**±0.0003	≈	0.0602±0.0925	+	0.0644±0.1388	+	0.0029±0.0001	-
MMF11	0.0041±0.0003	0.0046±0.0002	+	0.0047±0.0001	+	0.0072±0.0017	+	0.0069±0.0018	+	0.0044±0.0002	+	0.0047±0.0032	+	**0.0037**±0.0001	-
MMF12	**0.0016**±0.0001	0.0044±0.0094	+	0.0030±0.0003	+	0.0022±0.0005	+	0.0017±0.0002	≈	0.0020±0.0001	+	0.0017±0.0004	≈	0.0023±0.0001	+
MMF13	**0.0276**±0.0008	0.0719±0.0129	+	0.0314±0.0011	+	0.0333±0.0022	+	0.0279±0.0012	≈	0.0690±0.0173	+	0.0310±0.0027	+	0.0469±0.0096	+
MMF14	0.0624±0.0016	0.1306±0.0141	+	0.0663±0.0023	+	0.0662±0.0027	+	0.0631±0.0023	≈	0.1154±0.0118	+	–	+	**0.0382**±0.0006	-
MMF15	0.0504±0.0019	0.0848±0.0092	+	0.0505±0.0018	≈	0.0488±0.0026	-	0.0475±0.0018	-	0.0710±0.0059	+	–	+	**0.0385**±0.0006	-

MMF1_z	**0.0286**±0.0009	0.0709±0.0155	+	0.0332±0.0015	+	0.0317±0.0024	+	0.0306±0.0018	+	0.0669±0.0179	+	0.0418±0.0047	+	0.0625±0.0138	+
MMF1_e	0.3198±0.1320	1.1126±0.5107	+	0.3732±0.0725	+	0.5041±0.2247	+	**0.2845**±0.0683	≈	1.3087±0.6787	+	2.3082±0.7334	+	2.1337±0.6565	+
MMF14_a	0.0740±0.0026	0.1452±0.0114	+	0.0745±0.0019	≈	0.0734±0.0025	≈	0.0724±0.0024	≈	0.1441±0.0161	+	–	+	**0.0718**±0.0016	-
MMF15_a	0.0590±0.0029	0.1095±0.0162	+	0.0549±0.0022	-	0.0547±0.0023	-	0.0534±0.0023	-	0.0915±0.0115	+	–	+	**0.0475**±0.0013	-
SYM-PART simple	0.0538±0.0038	4.3945±1.4891	+	0.1105±0.0155	+	0.0707±0.0105	+	0.0726±0.0093	+	5.0544±1.3630	+	0.0705±0.0154	+	**0.0292**±0.0165	-
SYM-PART rotated	**0.0997**±0.2171	4.2530±1.7326	+	0.1297±0.0130	+	0.1205±0.2037	+	0.1246±0.0102	+	4.8056±1.8665	+	0.1581±0.2130	+	1.6371±1.1647	+
Omni-test	0.5560±0.1159	1.6491±0.2145	+	0.3633±0.0808	-	0.6623±0.1386	+	0.3104±0.0776	-	1.7329±0.2275	+	**0.1484**±0.0588	-	0.6085±0.2512	≈
+			22		17		16		11		22		19		13
≈			0		2		3		8		0		2		1
–			0		3		3		3		0		1		8

Table 5: The IGDf values obtained by all algorithms.

	MMODE_CSCD mean±std	DN_NSGAII mean±std		MO_Ring_PSO_SCD mean±std		MMOPIO mean±std		MO_PSO_MM mean±std		Omni_optimizer mean±std		DE_RLFR mean±std		TriMOEATAR mean±std	
MMF1	**0.0023**±0.0001	0.0041±0.0010	+	0.0034±0.0002	+	0.0037±0.0004	+	0.0034±0.0003	+	0.0033±0.0003	+	0.0036±0.0004	+	0.0043±0.0011	+
MMF2	0.0043±0.0004	0.0158±0.0176	+	0.0135±0.0024	+	**0.0041**±0.0004	-	0.0092±0.0012	+	0.0115±0.0140	+	0.0385±0.0203	+	0.0174±0.0079	+
MMF3	0.0039±0.0003	0.0087±0.0085	+	0.0098±0.0014	+	**0.0037**±0.0009	-	0.0071±0.0006	+	0.0165±0.0256	≈	0.0207±0.0107	+	0.0453±0.0824	+
MMF4	**0.0023**±0.0001	0.0033±0.0003	+	0.0034±0.0003	+	0.0040±0.0007	+	0.0048±0.0010	+	0.0027±0.0002	+	0.0037±0.0003	+	0.0188±0.0498	+
MMF5	**0.0023**±0.0000	0.0034±0.0002	+	0.0032±0.0001	+	0.0037±0.0007	+	0.0033±0.0003	+	0.0030±0.0002	+	0.0036±0.0002	+	0.0041±0.0013	+
MMF6	**0.0023**±0.0001	0.0034±0.0002	+	0.0033±0.0001	+	0.0034±0.0003	+	0.0033±0.0003	+	0.0029±0.0002	+	0.0036±0.0004	+	0.0035±0.0009	+
MMF7	**0.0024**±0.0001	0.0041±0.0005	+	0.0037±0.0002	+	0.0063±0.0009	+	0.0061±0.0008	+	0.0031±0.0002	+	0.0041±0.0005	+	0.0040±0.0014	+
MMF8	**0.0028**±0.0001	0.0037±0.0004	+	0.0039±0.0002	+	0.0034±0.0003	+	0.0031±0.0001	+	0.0030±0.0002	+	0.0040±0.0003	+	0.0058±0.0072	+
MMF9	**0.0104**±0.0006	0.0141±0.0018	+	0.0149±0.0018	+	0.0207±0.0056	+	0.0254±0.0061	+	0.0133±0.0013	+	0.0192±0.0035	+	0.0698±0.0070	+
MMF10	0.0399±0.0883	0.1432±0.1181	+	0.0630±0.0101	+	0.0223±0.0235	-	**0.0132**±0.0032	-	0.0707±0.0918	+	0.0606±0.1018	+	0.0378±0.0025	-
MMF11	**0.0114**±0.0010	0.0131±0.0013	+	0.0159±0.0016	+	0.0221±0.0048	+	0.0198±0.0034	+	0.0120±0.0009	≈	0.0275±0.0119	+	0.0740±0.0067	+
MMF12	**0.0021**±0.0001	0.0057±0.0113	+	0.0047±0.0003	+	0.0038±0.0007	+	0.0025±0.0001	+	0.0023±0.0001	+	0.0075±0.0045	+	0.0057±0.0002	+
MMF13	**0.0149**±0.0035	0.0174±0.0039	+	0.0242±0.0061	+	0.0231±0.0067	+	0.0177±0.0041	+	0.0164±0.0068	≈	0.0245±0.0096	+	0.1043±0.0082	+
MMF14	0.0914±0.0029	0.1390±0.0098	+	0.1013±0.0039	+	0.1020±0.0036	+	0.1010±0.0050	+	0.1216±0.0051	+		+	**0.0911**±0.0011	≈
MMF15	0.1004±0.0036	0.1744±0.0198	+	0.1053±0.0033	+	0.1093±0.0071	+	0.1060±0.0047	+	0.1400±0.0077	+		+	**0.0918**±0.0013	-
MMF1_z	**0.0023**±0.0001	0.0034±0.0003	+	0.0033±0.0001	+	0.0033±0.0002	+	0.0032±0.0001	+	0.0029±0.0003	+	0.0037±0.0004	+	0.0038±0.0005	+
MMF1_e	0.0089±0.0023	0.0150±0.0107	≈	0.0074±0.0007	-	**0.0035**±0.0003	-	0.0053±0.0005	-	0.0087±0.0072	-	0.0060±0.0049	-	0.0053±0.0007	-
MMF14_a	**0.0894**±0.0021	0.1514±0.0164	+	0.0971±0.0032	+	0.0954±0.0034	+	0.0966±0.0033	+	0.1308±0.0085	+		+	0.0934±0.0013	+
MMF15_a	0.1028±0.0048	0.1825±0.0214	+	0.1026±0.0043	≈	0.1088±0.0058	+	0.1055±0.0051	≈	0.1487±0.0137	+		+	**0.0913**±0.0026	-
SYM-PATR simple	0.0105±0.0012	0.0129±0.0014	+	0.0245±0.0030	+	0.0141±0.0019	+	0.0172±0.0024	+	0.0119±0.0012	+	0.0205±0.0035	+	0.0400±0.0042	+
SYM-PART rotated	**0.0098**±0.0011	0.0145±0.0013	+	0.0273±0.0037	+	0.0139±0.0018	+	0.0295±0.0035	+	0.0131±0.0016	+	0.0238±0.0057	+	0.0278±0.0033	+
Omni-test	0.0207±0.0024	0.0116±0.0005	-	0.0361±0.0036	+	0.0113±0.0014	-	0.0340±0.0036	+	**0.0102**±0.0005	-	0.0199±0.0051	≈	0.0216±0.0052	≈
+			20		20		17		19		17		20		16
≈			1		1		0		1		3		1		2
–			1		1		5		2		2		1		4

Table 6: The Friedman test results regarding the score and ranking of each algorithm on each performance indicator.

Algorithm	rPSP	rHV	IGDx	IGDf	Comprehensive result
	score(ranking)	score(ranking)	score(ranking)	score(ranking)	score(ranking)
MMODE_CSCD	2.05(1)	2.91(1)	2.05(1)	1.86(1)	1.00(1)
DN_NSGAII	7.00(8)	4.73(5)	7.00(8)	5.27(6)	6.75(6)
MO_Ring_PSO_SCD	3.95(4)	5.77(6)	4.05(4)	4.68(5)	4.75(4)
MMOPIO	3.91(3)	3.91(3)	3.91(3)	4.23(3)	3.00(2)
MO_PSO_MM	2.64(2)	4.36(4)	2.64(2)	4.32(4)	3.00(2)
Omni_optimizer	6.59(7)	3.41(2)	6.68(7)	3.36(2)	4.50(3)
DE_RLFR	5.41(6)	6.55(7)	5.27(6)	6.46(8)	6.75(6)
TriMOEATAR	4.45(5)	4.36(4)	4.41(5)	5.82(7)	5.25(5)

Table 7: The rPSP values obtained by MMODE_CSCD and its two variants.

	MMODE_CSCD	MMODE_SCD		MMODE_CSCD_ESM	
	mean±std	mean±std		mean±std	
MMF1	0.0414±0.0014	**0.0396**±0.0014	-	0.0427±0.0024	+
MMF2	**0.0102**±0.0018	0.0476±0.0265	+	0.0492±0.0327	+
MMF3	**0.0085**±0.0018	0.0284±0.0153	+	0.0217±0.0169	+
MMF4	0.0223±0.0011	0.0298±0.0061	+	**0.0216**±0.6734	≈
MMF5	0.0721±0.0034	0.0737±0.0044	≈	**0.0721**±0.0031	≈
MMF6	**0.0625**±0.0022	0.0641±0.0035	≈	0.0628±0.0021	≈
MMF7	**0.0221**±0.0014	0.0225±0.0026	≈	0.0240±0.0027	+
MMF8	**0.0489**±0.0031	0.1033±0.0536	+	0.0605±0.0095	+
MMF9	**0.0057**±0.0003	0.0485±0.0195	+	0.0058±0.3347	≈
MMF10	0.0370±0.1063	**0.0157**±0.0337	-	0.1602±0.0760	+
MMF11	**0.0041**±0.0003	0.0042±0.4063	≈	0.0041±0.2070	≈
MMF12	**0.0016**±0.0001	0.0025±0.8863	+	0.0016±0.0808	≈
MMF13	**0.0277**±0.0008	0.0323±0.0046	+	0.0284±0.9553	+
MMF14	**0.0624**±0.0016	0.0861±0.0166	+	0.0634±0.0023	+
MMF15	0.0504±0.0019	0.0587±0.0048	+	**0.0426**±0.0012	-
MMF1_z	**0.0288**±0.0009	0.0318±0.0087	≈	0.0313±0.0028	+
MMF1_e	**0.3617**±0.1759	2.8791±3.5450	+	5.6545±3.7566	+
MMF14_a	**0.0741**±0.0026	0.0974±0.0157	+	0.0750±0.0037	≈
MMF15_a	0.0591±0.0029	0.0819±0.0131	+	**0.0554**±0.0015	-
SYM-PART simple	**0.0539**±0.0039	1.2379±1.2940	+	1.1269±1.4574	+
SYM-PART rotated	**0.1075**±0.2512	2.4794±4.2305	+	1.7071±2.2000	+
Omni-test	**0.5636**±0.1177	2.1266±0.7485	+	1.8531±1.8819	+
+			15		13
≈			5		7
–			2		2

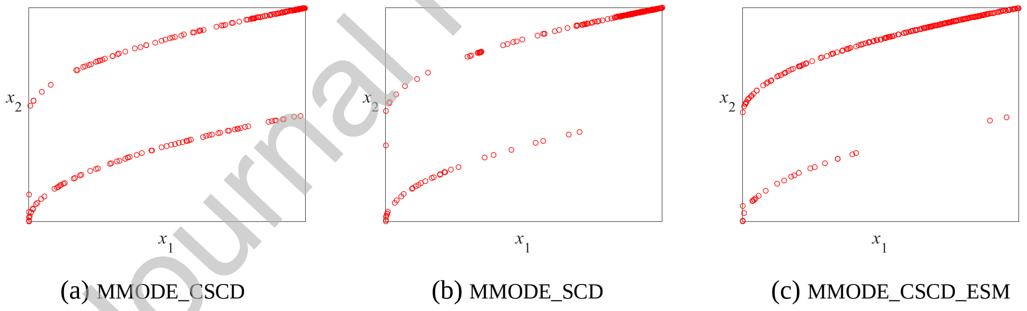
4.3 Verification of the proposed strategies

In this subsection, the effectiveness of the proposed CSCD and DBESM strategies are verified. Table 7 shows the rPSP values obtained by MMODE_CSCD and its two variants. MMODE_SCD indicates the SCD method instead of the CSCD method is adopted, and MMODE_CSCD_ESM represents the elite selection mechanism without considering the distance indicator.

As can be seen from Table 7, MMODE_CSCD outperforms MMODE_SCD on 15 functions. This is because CSCD helps MMODE_CSCD find the correct neighbor relationship. Moreover, MMODE_CSCD has better performance than MMODE_CSCD_ESM on 13 functions. The advantages of DBESM over ESM can be reflected by the functions whose PSs are far away from each other, such as SYM-PART simple. When the distance indicator is considered, the individual will prefer to choose the exemplar on the same PS. Thus the population can be distributed evenly on each PS as much as possible. In summary, the effectiveness of CSCD and DBESM are verified.

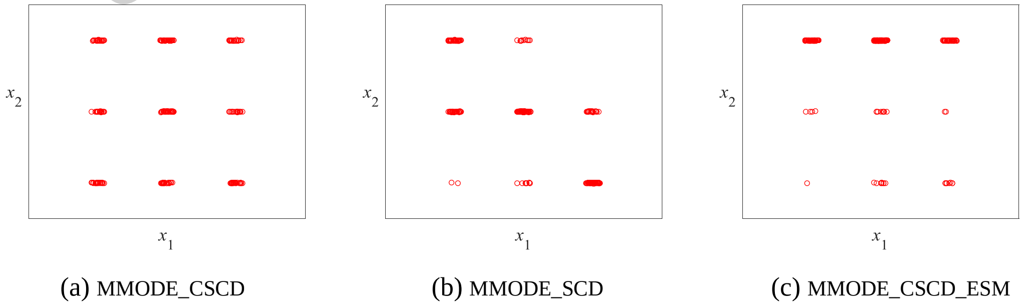
To observe the results of the above three algorithms more intuitively, the distribution of the Pareto-optimal solutions corresponding to the median rPSP values of 21 run times are plotted in Figures 5 and 6. Figure 5 shows the distribution of Pareto-optimal solutions on MMF2. MMF2 has two PSs with far distance, and these two PSs overlap on x_1. In Figure 5 (b), only one equivalent solution can be found. This is because the neighbors of one solution may come from another PS. Besides, in Figure 5 (c), when the distance indicator is not considered in the process of exemplar selection, the solutions on the lower PS may choose the exemplars that come from the upper PS. They will be misleaded to the upper PS. In contrast to Figures 5 (b-c), the distribution of solutions obtained by MMODE_CSCD in Figure 5 (a) is the best judging from convergence and uniformity.

Figure 6 shows the distribution of Pareto-optimal solutions on SYM-PART. MMODE_SCD loses one PS and has a smaller number of solutions on several other PSs. MMODE_CSCD_ESM finds all PSs. However, the number of solutions on nine PSs is not uniform. Only MMODE_CSCD has better convergence and overlap ratio between true PSs and obtained PSs.



(a) MMODE_CSCD (b) MMODE_SCD (c) MMODE_CSCD_ESM

Figure 5: The distribution of Pareto-optimal solutions obtained by MMODE_CSCD and its two variants on MMF2.



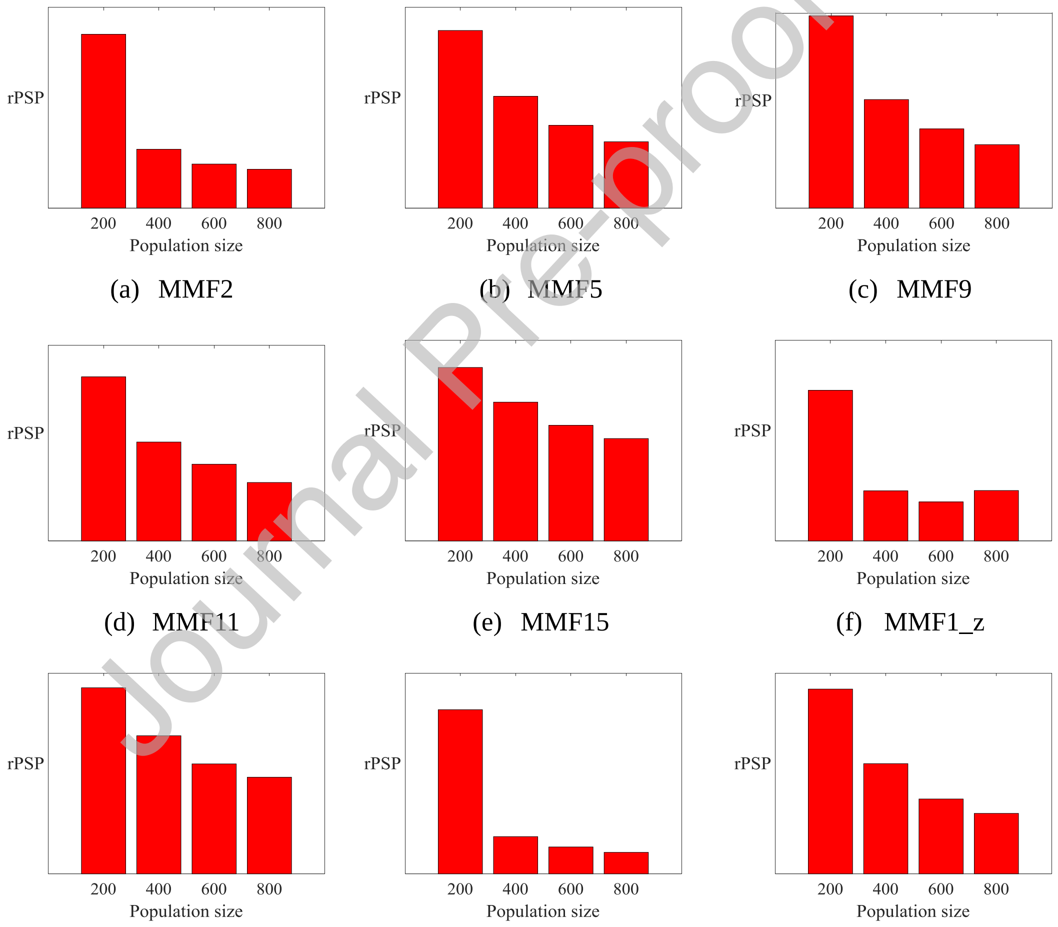
(a) MMODE_CSCD (b) MMODE_SCD (c) MMODE_CSCD_ESM

Figure 6: The distribution of Pareto-optimal solutions obtained by MMODE_CSCD and its two variants on on SYM-PART simple.

4.4 Parameter analysis

In this subsection, two key parameters in MMODE_CSCD are discussed. The first parameter is the population size, and the second one is n that is used for calculating the k value in the k-means algorithm.

First, the impact of the population size on the performance of MMODE_CSCD algorithm is discussed. Figures 7 (a) - (i) show the mean rPSP values of 21 runs obtained by MMODE_CSCD with different population sizes on nine functions. These nine functions have different complexities contributed by different shapes of PSs and different number of PSs. In Figure 7, the rPSP values decrease with the increase of the population size on all functions except for MMF9. This phenomenon proves the claim that a large population size can improve the performance of the multimodal multi-objective optimization algorithm. Specially, for MMF15, the descent degree of rPSP value is relatively uniform when population size changes from 200 to 800. While for the remaining eight functions, when the population size changes from 200 to 400, rPSP values decrease significantly. When the population size continues to increase, rPSP values decrease slowly. Hence, it can be concluded that population size is set to 400 is reasonable for MMODE_CSCD when considering the computational efficiency.



(a) MMF2 (b) MMF5 (c) MMF9

(d) MMF11 (e) MMF15 (f) MMF1_z

(g) MMF15_a (h) SYM-PART simple (i) Omni-test

Figure 7: The mean rPSP values obtained by MMODE_CSCD with different population sizes on nine functions.

Second, the performance of the proposed method with different n values is studied. As shown in line 3 of **Algorithm** 1, $k_j = \left\lceil \frac{|\mathbf{F}_j|}{n} \right\rceil$ indicates the number of clusters in $\mathbf{F}_j$, where $|\mathbf{F}_j|$ indicates the number of individuals in $\mathbf{F}_j$ and n is a fixed positive integer. It can be seen that parameter n indirectly determines the value of *k*. When *n* is large, the value of *k* will be small and vice versa. In this study, four different *n* values, i.e., 5, 10, 15, and 20, are studied. The experimental results provided in Table 8 show that when *n* = 5 or 10, the algorithm has similar results. However, when *n* = 15 or 20, the results of the algorithm are poor. Since one solution and its neighbors should come from the same local area, the inner distance of each class and *n* should be small. Meanwhile, if the number of solutions in the same class is too small, the CSCD value of boundary solutions will be affected. Thus, based on the rank-sum test results, *n* = 10 is recommended in this study.

Table 8: The rPSP values obtained by MMODE_CSCD with four different *n* values.

	n=10	*n*=5		*n*=15		*n*=20	
	mean±std	mean±std		mean±std		mean±std	
MMF1	0.0414±0.0014	0.0415±0.0013	≈	0.0418±0.0014	≈	0.0427±0.0022	+
MMF2	0.0102±0.0018	0.0236±0.0178	+	0.0323±0.0224	+	0.0331±0.0188	+
MMF3	0.0085±0.0018	0.0163±0.0093	+	0.0138±0.0053	+	0.0218±0.0165	+
MMF4	0.0223±0.0011	0.0219±0.0009	≈	0.0225±0.0009	≈	0.0233±0.0012	+
MMF5	0.0721±0.0034	0.0710±0.0024	≈	0.0718±0.0026	≈	0.0754±0.0058	+
MMF6	0.0625±0.0022	0.0619±0.0025	≈	0.0634±0.0026	≈	0.0637±0.0027	≈
MMF7	0.0221±0.0014	0.0219±0.0012	≈	0.0230±0.0021	≈	0.0227±0.0015	≈
MMF8	0.0489±0.0031	0.0507±0.0042	≈	0.0522±0.0041	+	0.0582±0.0094	+
MMF9	0.0057±0.0003	0.0061±0.0004	+	0.0065±0.0007	+	0.0067±0.0010	+
MMF10	0.0370±0.1063	0.1446±0.1030	+	0.0414±0.1074	≈	0.0129±0.0335	≈
MMF11	0.0041±0.0003	0.0042±0.0004	≈	0.0041±0.0003	≈	0.0039±0.0003	-
MMF12	0.0016±0.0001	0.0016±0.0001	≈	0.0017±0.0001	+	0.0017±0.0001	+
MMF13	0.0277±0.0008	0.0276±0.0009	≈	0.0288±0.0013	+	0.0284±0.0014	≈
MMF14	0.0624±0.0016	0.0614±0.0019	-	0.0646±0.0029	+	0.0660±0.0023	+
MMF15	0.0504±0.0019	0.0431±0.0020	-	0.0505±0.0023	≈	0.0519±0.0018	+
MMF1_z	0.0288±0.0009	0.0286±0.0013	≈	0.0299±0.0015	+	0.0320±0.0027	+
MMF1_e	0.3617±0.1759	0.4078±0.2133	≈	0.4831±0.3205	≈	1.3351±1.7656	+
MMF14_a	0.0741±0.0026	0.0748±0.0022	≈	0.0738±0.0018	≈	0.0737±0.0024	≈
MMF15_a	0.0591±0.0029	0.0546±0.0021	-	0.0604±0.0026	≈	0.0592±0.0027	≈
SYM-PART simple	0.0539±0.0039	0.0517±0.0023	≈	0.0611±0.0072	+	0.0663±0.0126	+
SYM-PART rotated	0.1075±0.2512	0.0553±0.0053	≈	0.1160±0.2184	+	0.4868±1.1791	+
Omni-test	0.5636±0.1177	0.5832±0.1251	≈	0.6820±0.1495	+	0.8794±0.2513	+
+			4		11		15
≈			15		11		6
–			3		0		1

5. Conclusions and future works

In this study, a new clustering-based differential evolution algorithm is proposed to address multimodal multi-objective optimization problems. The CSCD method adopts the k-means algorithm to overcome the shortcomings of SCD. The k-means algorithm is used to divide the individuals in the same non-dominated level into multiple classes. Thus one individual and its neighbors can be from the same PS. By doing this, more accurate CSCD values are obtained. To improve population diversity, the DBESM is developed to determine the exemplar for each individual. The proposed algorithm is compared with seven peer algorithms on the CEC’2019 multimodal multi-objective benchmark suite, and results demonstrate the superiority of MMODE_CSCD in both decision space and objective space.

In future work, different clustering algorithms will be introduced into MMODE_CSCD to verify their effects on CSCD. Furthermore, MMODE_CSCD will be applied to solve different practical problems. Moreover, a new maintain mechanism of local PSs should be studied to find local PSs.

Acknowledgment

This work is supported by the National Natural Science Foundation of China (61922072, 61976237, 61876169, 61806179, 61673404).

Reference:

[1] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, pp. 182-197, 2002.

[2] A. Trivedi, D. Srinivasan, K. Sanyal, and A. Ghosh, "A survey of multiobjective evolutionary algorithms based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 21, pp. 440-462, 2017.

[3] H. A. Abbass, R. Sarker, and C. Newton, "PDE: a Pareto-frontier differential evolution approach for multi-objective optimization problems," in *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546)*, 2001, pp. 971-978 vol. 2.

[4] J. J. Liang, C. T. Yue, and B. Y. Qu, "Multimodal multi-objective optimization: A preliminary study," in *2016 IEEE Congress on Evolutionary Computation (CEC)*, 2016, pp. 2454-2461.

[5] S. Biswas, S. Kundu, and S. Das, "Inducing niching behavior in differential evolution through local information sharing," *IEEE Transactions on Evolutionary Computation*, vol. 19, pp. 246-263, 2015.

[6] Y. Zhang, Y. Gong, H. Zhang, T. Gu, and J. Zhang, "Toward fast niching evolutionary algorithms: a locality sensitive hashing-based approach," *IEEE Transactions on Evolutionary Computation*, vol. 21, pp. 347-362, 2017.

[7] Y. Xu, "A niching particle swarm segmentation of infrared images," in *2010 Sixth International Conference on Natural Computation*, 2010, pp. 3739-3742.

[8] Y. Liu, H. Ishibuchi, Y. Nojima, N. Masuyama, and K. Shang, "A double-niched evolutionary algorithm and Its behavior on polygon-based problems," in *Parallel Problem Solving from Nature – PPSN XV*, Cham, 2018, pp. 262-273.

[9] R. Tanabe and H. Ishibuchi, "A niching indicator-based multi-modal many-objective optimizer," *Swarm and Evolutionary Computation*, vol. 49, pp. 134-146, 2019.

[10] B. Y. Qu, G. S. Li, Q. Q. Guo, L. Yan, X. Z. Chai, and Z. Q. Guo, "A niching multi-objective harmony search algorithm for multimodal multi-objective problems," in *2019 IEEE Congress on Evolutionary Computation (CEC)*, 2019, pp. 1267-1274.

[11] O. J. Mengshoel and D. E. Goldberg, "The crowding approach to niching in genetic algorithms," *Evolutionary Computation*, vol. 16, pp. 315-354, 2008.

[12] L. Qing, W. Gang, Y. Zaiyue, and W. Qiuping, "Crowding clustering genetic algorithm for multimodal function optimization," *Applied Soft Computing*, vol. 8, pp. 88-95, 2008.

[13] B. Sareni and L. Krahenbuhl, "Fitness sharing and niching methods revisited," *IEEE Transactions on Evolutionary Computation*, vol. 2, pp. 97-106, 1998.

[14] A. D. Cioppa, C. D. Stefano, and A. Marcelli, "On the role of population size and niche radius in fitness sharing," *IEEE Transactions on Evolutionary Computation,* vol. 8, pp. 580-592, 2004.

[15] B. Bošković and J. Brest, "Clustering and differential evolution for multimodal optimization," in *2017 IEEE Congress on Evolutionary Computation (CEC)*, 2017, pp. 698-705.

[16] Z. Wang, Z. Zhan, Y. Lin, W. Yu, H. Yuan, T. Gu*, et al.*, "Dual-strategy differential evolution with affinity propagation clustering for multimodal optimization problems," *IEEE Transactions on Evolutionary Computation,* vol. 22, pp. 894-908, 2018.

[17] A. Petrowski, "A clearing procedure as a niching method for genetic algorithms," in *Proceedings of IEEE International Conference on Evolutionary Computation*, 1996, pp. 798-803.

[18] G. Dick, "Automatic identification of the niche radius using spatially-structured clearing methods," in *IEEE Congress on Evolutionary Computation*, 2010, pp. 1-8.

[19] A. Passaro and A. Starita, "Particle swarm optimization for multimodal functions: a clustering approach," *Journal of Artificial Evolution and Applications,* vol. 2008, 2008.

[20] S. Das, S. S. Mullick, and P. N. Suganthan, "Recent advances in differential evolution – an updated survey," *Swarm and Evolutionary Computation,* vol. 27, pp. 1-30, 2016.

[21] O. M. Shir, M. Preuss, B. Naujoks, and M. Emmerich, "Enhancing decision space diversity in evolutionary multiobjective algorithms," in *International Conference on Evolutionary Multi-Criterion Optimization*, 2009, pp. 95-109.

[22] K. Deb and S. Tiwari, "Omni-optimizer: a procedure for single and multi-objective optimization," in *Evolutionary Multi-Criterion Optimization*, Berlin, Heidelberg, 2005, pp. 47-61.

[23] K. Chan and T. Ray, "An evolutionary algorithm to maintain diversity in the parametric and the objective space," in *International Conference on Computational Robotics and Autonomous Systems (CIRAS), Centre for Intelligent Control, National University of Singapore*, 2005.

[24] A. Zhou, Q. Zhang, and Y. Jin, "Approximating the set of Pareto-optimal solutions in both the decision and objective spaces by an estimation of distribution algorithm," *IEEE transactions on evolutionary computation,* vol. 13, pp. 1167-1189, 2009.

[25] C. Yue, B. Qu, and J. Liang, "A multiobjective particle swarm optimizer using ring topology for solving multimodal multiobjective problems," *IEEE Transactions on Evolutionary Computation,* vol. 22, pp. 805-817, 2018.

[26] Y. Wang, Z. Yang, Y. Guo, J. Zhu, and X. Zhu, "A novel multi-objective competitive swarm optimization algorithm for multi-modal multi objective problems," in *2019 IEEE Congress on Evolutionary Computation (CEC)*, 2019, pp. 271-278.

[27] J. Liang, W. Xu, C. Yue, K. Yu, H. Song, O. D. Crisalle*, et al.*, "Multimodal multiobjective optimization with differential evolution," *Swarm and Evolutionary Computation,* vol. 44, pp. 1028-1059, 2019.

[28] M. Pal and S. Bandyopadhyay, "Differential evolution for multi-modal multi-objective problems," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2019, pp. 1399-1406.

[29] J. Liang, Q. Guo, C. Yue, B. Qu, and K. Yu, "A self-organizing multi-objective particle

swarm optimization algorithm for multimodal multi-objective problems," in *Advances in Swarm Intelligence*, Cham, 2018, pp. 550-560.

[30] Y. Hu, J. Wang, J. Liang, K. Yu, H. Song, Q. Guo*, et al.*, "A self-organizing multimodal multi-objective pigeon-inspired optimization algorithm," *Science China Information Sciences,* vol. 62, p. 70206, 2019.

[31] Q. Fan and X. Yan, "Solving multimodal multiobjective problems through zoning search," *IEEE Transactions on Systems, Man, and Cybernetics: Systems,* pp. 1-12, 2019.

[32] W. Zhang, G. Li, W. Zhang, J. Liang, and G. G. Yen, "A cluster based PSO with leader updating mechanism and ring-topology for multimodal multi-objective optimization," *Swarm and Evolutionary Computation,* vol. 50, p. 100569, 2019.

[33] B. Qu, C. Li, J. Liang, L. Yan, K. Yu, and Y. Zhu, "A self-organized speciation based multi-objective particle swarm optimizer for multimodal multi-objective problems," *Applied Soft Computing,* vol. 86, p. 105886, 2020.

[34] Z. Li, L. Shi, C. Yue, Z. Shang, and B. Qu, "Differential evolution based on reinforcement learning with fitness ranking for solving multimodal multiobjective problems," *Swarm and Evolutionary Computation,* vol. 49, pp. 234-244, 2019.

[35] Y. Liu, G. G. Yen, and D. Gong, "A multimodal multiobjective evolutionary algorithm using two-archive and recombination strategies," *IEEE Transactions on Evolutionary Computation,* vol. 23, pp. 660-674, 2019.

[36] J. Zhang and A. C. Sanderson, "JADE: adaptive differential evolution with optional external archive," *IEEE Transactions on Evolutionary Computation,* vol. 13, pp. 945-958, 2009.

[37] J. Liang, B. Qu, D. Gong, and C. Yue, "Problem definitions and evaluation criteria for the cec 2019 special session on multimodal multiobjective optimization," in *Computational Intelligence Laboratory, Zhengzhou University*, 2019.

[38] C. Yue, B. Qu, K. Yu, J. J. Liang, and X. Li, "A novel scalable test problem suite for multimodal multiobjective optimization," *Swarm and evolutionary computation,* vol. 48, pp. 62-71, 2019.

[39] J. Alcalá-Fdez, L. Sánchez, S. García, M. J. del Jesus, S. Ventura, J. M. Garrell*, et al.*, "KEEL: a software tool to assess evolutionary algorithms for data mining problems," *Soft Computing,* vol. 13, pp. 307-318, 2009.