Research Article

A Constrained Solution Update Strategy for Multiobjective Evolutionary Algorithm Based on Decomposition

Yuchao Su, Qiuzhen Lin , Jia Wang, Jianqiang Li , Jianyong Chen , and Zhong Ming

College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China

Correspondence should be addressed to Qiuzhen Lin; qiuzhlin@szu.edu.cn

Received 28 November 2018; Accepted 23 January 2019; Published 8 May 2019

Academic Editor: Alex Alexandridis

Copyright © 2019 Yuchao Su et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper proposes a constrained solution update strategy for multiobjective evolutionary algorithm based on decomposition, in which each agent aims to optimize one decomposed subproblem. Different from the existing approaches that assign one solution to each agent, our approach allocates the closest solutions to each agent and thus the number of solutions in an agent may be zero and no less than one. Regarding the agent with no solution, it will be assigned one solution in priority, once offspring are generated closest to its subproblem. To keep the same population size, the agent with the largest number of solutions will remove one solution showing the worst convergence. This improves diversity for one agent, while the convergence of other agents is not lowered. On the agent with no less than one solution, offspring assigned to this agent are only allowed to update its original solutions. Thus, the convergence of this agent is enhanced, while the diversity of other agents will not be affected. After a period of evolution, our approach may gradually reach a stable status for solution assignment; i.e., each agent is only assigned with one solution. When compared to six competitive multiobjective evolutionary algorithms with different population selection or update strategies, the experiments validated the advantages of our approach on tackling two sets of test problems.

1. Introduction

In real-world applications, it is often needed to handle multiobjective optimization problems (MOPs) [1], such as recommendation systems [2, 3], privacy computing [4], and resource assignment [5–7]. Due to the conflicts among different objectives, the results of MOPs will output a set of Pareto solutions (PS) and their mapping in the objective space is called Pareto front (PF) [8–10]. These MOPs may be characterized with complicated features [11–13], which cannot be well solved by traditional mathematical methods. Instead, multiobjective evolutionary algorithms (MOEAs) can effectively obtain a set of solutions in one single run, which have shown a very promising performance in tackling different kinds of MOPs [14–16] and become very popular during the recent decades.

In the design of MOEAs, evolution and selection are their two important mechanisms [17–19]. The first one modifies individuals in order to approach the true PF, while the second one selects the most promising individuals to constitute the new population for next generation. Based on the selection

mechanisms, most MOEAs can be classified into three types, i.e., Pareto-based MOEAs [20–23], indicator-based MOEAs [24–29], and decomposition-based MOEAs [30–36]. Compared to the selection operators used in Pareto-based and indicator-based MOEAs, decomposition-based MOEAs are able to provide more flexibility to balance convergence and diversity [37], which has been found to provide a better performance when tackling some complicated MOPs, as reported in [38]. In this sort of MOEAs, the target MOP is decomposed into a set of subproblems, which are solved simultaneously using a set of cooperative agents. Each agent aims to optimize one subproblem in MOEA/D [39]. Due to the simplicity and effectiveness of MOEA/D, this framework has triggered a considerable amount of research, aiming to improve different components of MOEA/D, such as the adjustment and generation of weight vectors [40–44], dynamic resource allocation [45–47], enhanced evolutionary operators [48–50], and improved population selection or update mechanisms [51–57].

Especially, regarding the population selection or update mechanisms for decomposition-based MOEAs, the offspring

in MOEA/D [39] are allowed to update any solution in population. However, this method may significantly lower the diversity when a very good solution may replace most of the others in several generations. In MOEA/D-DE [58], its solution update approach is controlled by two preset probabilities δ and n_r , which obtains a better balance of convergence and diversity. The offspring is only allowed to update the parent solutions from the neighborhood with a probability δ and from the entire population with a probability $(1-\delta)$. Moreover, an offspring can only replace at most n_r parent solutions. This strategy was mostly used in the following design of decomposition-based MOEAs [49, 54]. Different from the decomposition approach in MOEA/D-DE, MOEA/D-M2M [38] separates the search space into multiple search subspace, which simplifies the solving of MOPs in each subspace and the solution update is constrained by including the equal number of solutions in each subspace. Thus, MOEA/D-M2M was shown to be very effective for complicated MOPs that strongly emphasize diversity (i.e., MOP problems [38]). To further find a better match of solutions and subproblems, a stable matching model was proposed in MOEA/D-STM [59], which associates the solutions to subproblems according to their respective preferences. In this way, MOEA/D-STM can maintain a good convergence speed and population diversity. Similarly, an improved interrelationship model was designed in MOEA/D-IR [37] to associate the solutions to subproblems based on their mutual-preferences, which is an essentially diversity first and convergence second strategy. Moreover, two improved versions [53] for MOEA/D-STM were proposed to embed the concept of the incomplete preference lists in the stable matching model, which further strengthens the diversity. In [51], an adaptive replacement neighborhood size was proposed to assign an offspring to its most appropriate subproblems, obtaining a better balance of convergence and diversity. In MOEA/D-ACD [54], an adaptive constrained decomposition approach was presented, in which the update regions of decomposition approach are constrained to maintain the diversity. Moreover, to further enhance the performance in MOPs with more than three objectives, decomposition approach and Pareto domination were simultaneously used in MOEA/DD [44], decomposition-based-sorting and angle-based-selection approaches were proposed in MOEA/D-SAS [57], and the diversity was preferred in solution update by selecting certain closest subproblems for an offspring in [60].

On the other hand, another kind of population selection or update mechanisms in MOEA/D aims to improve their used decomposition functions. In MOEA/D [39], three traditional decomposition functions, i.e., the weighted sum (WS) approach, the Tchebycheff (TCH) approach, and the penalty-based boundary intersection (PBI) approach, were employed. In [61, 62], a local PBI and WS were, respectively, designed to constrain the update regions of decomposition approaches, which avoid the diversity loss. In [63, 64], an adaptive Pareto front scalarizing (PaS) and penalty-based boundary intersection (PaP) decomposition approaches were, respectively, introduced to match the true PFs with various shapes. Two decomposition approaches were presented in MOEA/AD [65] and DECAL [66] to deal with the complicated PF. In

MOEA/AD, two coevolved populations were, respectively, updated by the two decomposition functions to fit different PF shapes, while two novel decomposition functions were, respectively, used to accelerate the convergence speed and enhance the population diversity in DECAL. Recently, MOEA/D-LTD [67] was proposed to trace the PF shape, in which the learning module predicts the PF shape and the decomposition function is adaptively adjusted to fit its PF shape.

Most of the above MOEAs all abide one basic principle that each agent should be assigned with one solution in order to find the optimal value for its subproblem. However, this kind of solution assignment may not be effective and efficient in decomposition-based MOEAs, as the solution assigned to the agent may be far away from its subproblem. In such case, it cannot truly reflect the diversity of each agent and cannot provide the correct neighboring information in evolution, which may slow down the convergence as decomposition-based MOEAs are designed as an essentially collaborative evolutionary framework. Therefore, a constrained solution update (CSU) strategy is designed in this paper for decomposition-based MOEAs to alleviate the above problem. The solutions are only assigned to the agent that handles the closest subproblem. This way, the correct neighboring information can be provided to guide the evolution and it is straightforward to show the diversity of each agent. In this case, the number of solutions in each agent may be zero or no less than one. To maintain the diversity of each agent, the offspring assigned to one agent are only allowed to renew its original solutions. When the agent has no solution, it will be assigned one solution in priority, once offspring are generated closest to its subproblem. To keep the same population size, the agent with the largest number of solutions will remove one solution showing the worst convergence. Thus, the diversity of one agent is enhanced, while the convergence of other agents is not affected. After a period of evolution, a stable status for solution assignment is anticipated so that each agent only has one solution. When compared to the existing population selection or update strategies for decomposition-based MOEAs, our experiments validate the superiority of the proposed approach when tackling two sets of complicated test MOPs.

The main contributions of this paper are clarified below.

- (1) Each agent may be assigned with no solution, or no less than one solution, which is different from the existing approaches that only assign one solution to each agent. This approach can truly reflect the diversity on the agents and provide the correct neighboring information in evolution.
- (2) A CSU strategy is designed for each agent in order to maintain diversity for all the agents without affecting their convergence. The agent with no solution will be assigned first, while the agent with the largest number of solutions will remove one solution showing the worst convergence. By this way, a stable status for solution assignment may be reached, so that each agent only has one solution, which ensures diversity in decomposition-based MOEAs.

- (3) When solution assignment is under an unstable status such that at least one agent is still not assigned any solution, the mating parents are randomly selected from the best solutions from all the agents, as the neighboring agent may have no solution. This random selection of mating parents helps to enhance the exploration ability in our algorithm.

The rest of this paper is organized as follows. Section 2 provides the related background, such as MOPs and the used decomposition function in this paper. Section 3 introduces the details of the proposed algorithm MOEA/D-CSU. The experimental results and discussions are provided in Section 4, while the conclusions and some future research directions are given in Section 5.

2. Related Background

2.1. Multiobjective Optimization Problems. Multiobjective optimization problems often need to optimize several conflicting objectives, which can be modeled by

$$\text{Minimize } F(x) = \{f_1(x), f_2(x), \dots, f_m(x)\}, \quad (1)$$

where $x = (x_1, x_2, \dots, x_n) \in \Omega$ is an n dimensional decision vector in the decision space Ω and m is the number of objectives. The target of MOP in (1) is to minimize all the objectives simultaneously.

2.2. The Decomposition Function. In this paper, the modified Tchebycheff method [55] is used for decomposing the MOP in (1), which is defined by

$$g^{Tch}(x | w, z^*) = \max_{1 \leq i \leq m} \left(\frac{|f_i(x) - z_i^*|}{w_i} \right), \quad (2)$$

where $w = (w_1, w_2, \dots, w_m)$ is a preset weight vector with $w_i \geq 0$ for each $i \in [1, m]$ and $\sum_{i=1}^m w_i = 1$, while $z^* = \{z_1^*, z_2^*, \dots, z_m^*\}$ is the ideal point by setting $z_i^* = \{\min f_i(x) | x \in \Omega\}$ for each $i \in [1, m]$. When using N uniformly distributed weight vectors in (2), the MOP in (1) is decomposed into a set of N subproblems, which can be solved by a set of N collaborative agents. The population selection or update strategies designed in decomposition-based MOEAs will reasonably allocate the solutions to the agents [39]. Different from the existing approaches [39, 58] that assign one solution to each agent, the agent in our approach is only allocated by the solutions that are closest to its subproblem, resulting in the fact that the number of solutions in each agent may be zero or no less than one. To show (2) more visually, a case of updating solution is depicted in Figure 1, where s_1 is a solution in current population while s_2 and s_3 are two offspring. For this case, s_3 can update the subproblem but s_2 cannot do this, because the yellow region is the improvement domain of s_1 by the weight vector and (2), and a solution like s_3 falling into the region can update the subproblem. Actually, (2) decides the profile of the region [54].

FIGURE 1: Update the subproblem by (2).

3. Our Algorithm: MOEA-CSU

Let $w^1, w^2, \dots, w^N$ be N weight vectors and A_i denote the agent, which aims to optimize the subproblem in (2) with the weight vector w^i ($i \in [1, N]$). In this paper, we classify the status of solution assignment into two kinds, i.e., a stable status (each A_i is assigned only one solution) and an unstable status (at least one A_i is not assigned any solution) ($i \in [1, N]$). Generally, an initial population often starts from the unstable status, while the purpose of our CSU strategy is to reach the stable status, which properly maintains the diversity of each agent.

3.1. Our CSU Strategy. Let P and O , respectively, denote the parent population and offspring population. At each generation, the solution set from P assigned to agent A_i is denoted by Λ_i^p ($i \in [1, N]$), while the solution set from O assigned to agent A_i is denoted by Λ_i^o ($i \in [1, N]$). In this paper, Λ_i^p and Λ_i^o can be obtained using the closest vector angles to the weight vector w^i of agent A_i , as follows:

$$\Lambda_i^p = \{x \in P \mid \langle F(x) - z^*, w^i \rangle \leq \langle F(x) - z^*, w^j \rangle\} \quad (3)$$

for $\forall j \in [1, N],$

$$\Lambda_i^o = \{x \in O \mid \langle F(x) - z^*, w^i \rangle \leq \langle F(x) - z^*, w^j \rangle\} \quad (4)$$

for $\forall j \in [1, N],$

where $z^* = (z_1^*, z_2^*, \dots, z_m^*)$ (m is the number of objectives) is an utopian objective vector that is approximated by the minimal objective values from the current parent and offspring populations, i.e., $z_i^* = \min f_i(x), x \in P \cup O$ for each $i \in [1, m]$,

```

(1) Get  $\Lambda_i^p$  and  $\Lambda_i^o$  respectively from  $\mathbf{P}$  and  $\mathbf{O}$  with Eqs. (3)-(4)
(2) for  $i=1$  to  $N$ 
(3)   if  $|\Lambda_i^o|!=0$ 
(4)     if  $|\Lambda_i^p|==0$ 
(5)       find one solution  $x$  with the minimal value in Eq. (2) from  $\Lambda_i^o$ 
(6)       add  $x$  into  $\Lambda_i^p$ 
(7)       find one agent  $\Lambda_k^p$  with the largest number of solutions
(8)       remove one solution with the worst value in Eq. (2) from  $\Lambda_k^p$ 
(9)     else
(10)      let  $\mathbf{U} = \Lambda_i^o \cup \Lambda_i^p$  and set  $\Lambda_i^p$  as an empty set
(11)      sort the solutions in  $\mathbf{U}$  ascendingly using the aggregated values in Eq. (2)
(12)      select the first  $|\Lambda_i^p|$  solutions from  $\mathbf{U}$  to compose a new  $\Lambda_i^p$ 
(13)    end if
(14)  end if
(15) end for
(16) collect all the  $\Lambda_i^p$  to compose a new  $\mathbf{P}$ 
(17) if each  $\Lambda_i^p$  is not empty
(18)    $status = \mathbf{True}$  // solution assignment is under the stable status
(19) end if
(20) return  $[\mathbf{P}, status]$ 

```

ALGORITHM 1: CSU($\mathbf{P}, \mathbf{O}, N$), constrained solution update.

and $\langle \mathbf{F}(\mathbf{x}) - \mathbf{z}^*, \mathbf{w}^i \rangle$ indicates the acute angle of two vectors $\mathbf{F}(\mathbf{x}) - \mathbf{z}^*$ and $\mathbf{w}^i$, as defined by

$$\langle \mathbf{F}(\mathbf{x}) - \mathbf{z}^*, \mathbf{w}^i \rangle = \arccos \left| \frac{\sum_{k=1}^m (f_k(\mathbf{x}) - z_k^*) \cdot w_k^i}{\sqrt{\sum_{k=1}^m (f_k(\mathbf{x}) - z_k^*)^2} \cdot \sqrt{\sum_{k=1}^m (w_k^i)^2}} \right|. \quad (5)$$

The design principle of our method is simple and effective. When Λ_i^o is empty, Λ_i^p will not be updated. Otherwise, the offspring assigned to each agent is only allowed to renew its original solutions; i.e., the solutions in Λ_i^o can only renew Λ_i^p , which will speed up the convergence for A_i , while the diversity of other agents is not affected, as the solutions in Λ_i^o are not allowed to update the solutions for other agents. In more detail, two cases for Λ_i^p are considered when Λ_i^o is not empty, i.e., $|\Lambda_i^p| = 0$ and $|\Lambda_i^p| > 0$, where $|\Lambda_i^p|$ indicates the size of Λ_i^p .

In the case with $|\Lambda_i^p| = 0$, as the agent A_i is not assigned with any solution before, one solution in Λ_i^o with the best aggregated value using (2) is assigned to A_i . To keep the same population size, the agent A_k ($k \in [1, N]$) with the largest number of solutions is found and then one solution in Λ_i^p having the worst aggregated value in (2) is removed. Please note that if more than one agent has the same largest number of solutions, one of them is randomly selected to remove one worst solution. This way, the agent A_i is assigned one solution to optimize its subproblem, which enhances its diversity, while the convergence for other agents (e.g., A_k) is not affected.

In other case with $|\Lambda_i^p| > 0$, the solutions in Λ_i^p and Λ_i^o are combined into $\mathbf{U}$, and they are sorted using the aggregated values in (2) with an ascending order. The first $|\Lambda_i^p|$ solutions

are selected from $\mathbf{U}$ to compose a new Λ_i^p , which keeps the same number of solutions for the agent A_i . By this way, the convergence for the agent A_i is enhanced, while the diversity for other agents is not affected, as the offspring in Λ_i^o are not allowed to update them.

With the above operations, the number of solutions assigned to each agent will be gradually reduced to only one, once there exists a solution generated around its subproblem. Finally, this approach may reach the stable status. Note that the stable status may be unreachable when solving MOPs with complicated PFs, e.g., disconnected and degenerated PFs. To further clarify the CSU strategy, its pseudocode is given in Algorithm 1. Please note that Algorithm 1 will return the updated population $\mathbf{P}$ and the status of solutions assignment.

3.2. The Used Recombination Operator. In this paper, the evolutionary operator [68] in MOEA/D-M2M is used, which is, respectively, defined in (6) and (7), as follows:

$$y = x + (z - x) r^1 (1 - (r^2)^\eta), \quad (6)$$

$$y = y + r^3 (1 - (r^2)^\eta) (u - l), \quad (7)$$

where x and z are the decision variables from two parents and y is that of an offspring, while u and l are, respectively, the lower and upper bounds for that decision variable. The crossover operator is defined in (6), where r^1 and r^2 are two random real numbers respectively generated from $(-1, 1)$ and $(0, 1)$, and η is an index that is set to $-(1-g/G_{\max})^{0.7}$ ($G_{\max}$ and g are, respectively, the maximum number of generation and the current generation). The mutation operator is defined in (7), where r^3 is a random real number produced from


```

(1) generate  $N$  weight vectors  $\{\mathbf{w}^1, \mathbf{w}^2, \dots, \mathbf{w}^N\}$ , initialize the values of  $g$ ,  $status$ , an
    offspring population  $\mathbf{O}$ , and generate an initial population  $\mathbf{P} = \{\mathbf{p}^1, \mathbf{p}^2, \dots, \mathbf{p}^N\}$ 
(2) get  $\Lambda_i^p$  from  $\mathbf{P}$  with (3)
(3) while  $g < G_{\max}$ 
(4)   for  $i=1$  to  $N$ 
(5)     if  $status == 0$ 
(6)       collect the best solution of each  $\Lambda_i^p$  to form the mating pool
(7)     else
(8)       set the neighbors of subproblem  $i$  as the mating pool
(9)     end if
(10)    generate  $\mathbf{o}^i$  with  $\mathbf{p}^i$  and the parents from the mating pool
(11)    evaluate the objectives in  $\mathbf{o}^i$  and update  $\mathbf{z}^*$  in (2)
(12)    add  $\mathbf{o}^i$  into the offspring population  $\mathbf{O}$ 
(13)  end for
(14)   $[\mathbf{P}, status] = \text{CSU}(\mathbf{P}, \mathbf{O}, N)$ 
(15)  set  $g = g + 1$  and initialize  $\mathbf{O}$  as an empty set
(16) end while
(17) return  $\mathbf{P}$ 

```

ALGORITHM 2: MOEA/D-CSU.

(-0.25, 0.25). When y is out of the parameter boundary, a repair operation will be executed in (8) and (9), as follows:

$$y = u - r^4(u - y) \quad \text{if } y > u, \quad (8)$$

$$y = l + r^4(y - l) \quad \text{if } y < l, \quad (9)$$

where r^4 is a random real number generated in (-0.5, 0.5).

3.3. MOEA/D-CSU. In this section, our CSU strategy is embedded into a general framework of decomposition-based MOEAs, named MOEA/D-CSU. Its pseudocode is provided in Algorithm 2. In line (1), N weight vectors $\mathbf{w}^1, \mathbf{w}^2, \dots, \mathbf{w}^N$ are generated and then the value of generation counter g is initialized to 1, the value of $status$ is initialized to *False* (indicating the unstable status of solution assignment), the offspring population $\mathbf{O}$ is initialized as an empty set, and an initial population $\mathbf{P} = \{\mathbf{p}^1, \mathbf{p}^2, \dots, \mathbf{p}^N\}$ is generated randomly in line (1). In line (2), Λ_i^p for each agent is obtained from $\mathbf{P}$ by (5). If g is smaller than the preset maximum number of generations $G_{\max}$, the following evolution and selection procedures in lines (4)-(15) are run. For each subproblem i in line (4), it is checked whether the $status$ of the solutions assignment is stable in line (5). If it is not, we collect the best solution in each Λ_i^p to form the mating pool. Otherwise, we set the neighbors of subproblem i as the mating pool in line (8) based on the Euclidean distance between the used weight vectors. Here, the neighbor size T in line (8) is dynamically adjusted according to the number of generations, by using

$$T = \left\lceil N \times \frac{(G_{\max} - g)}{G_{\max}^{1.25}} \right\rceil + 1. \quad (10)$$

After that, an offspring $\mathbf{o}^i$ is generated using the recombination operators defined in (6)-(9) based on $\mathbf{p}^i$ and the mating parents in line (10) and is further evaluated to get the objective

values in line (11), which is used to update the approximately ideal point $\mathbf{z}^*$ in (2). In line (12), this offspring $\mathbf{o}^i$ is added into the offspring population $\mathbf{O}$. After all the offspring are collected into $\mathbf{O}$, the CSU strategy (Algorithm 1) is run in line (14) with the inputs $\mathbf{P}, \mathbf{O}, N$ to get a new population $\mathbf{P}$. In line (15), the value of g is increased by 1 and the offspring population $\mathbf{O}$ is reset to an empty set. The above evolutionary process will be terminated when g reaches $G_{\max}$, and the final population $\mathbf{P}$ is reported.

4. Experimental Results

4.1. Benchmark Problems and Parameters Settings. In this study, two complicated test suites (MOP [38] and IMB [69]) were used to assess the performance of MOEA/D-CSU, including *MOP1-MOP7* and *IMB1-IMB10*. They have complicated mathematical features on the PS shapes. Please note that *MOP1-MOP5*, *IMB1-IMB3*, and *IMB7-IMB9* have two optimization objectives, while *MOP6-MOP7*, *IMB4-IMB6*, and *IMB10* include three optimization objectives. The number of decision variables is set to 10 for all the test problems. Regarding the biobjective and three-objective test problems, the population sizes were, respectively, set to 100 and 300 as suggested in [38], while the maximum numbers of function evaluations were, respectively, set to 3×10^5 and 9×10^5 . The performance of MOEA/D-CSU is compared to six competitive MOEA/Ds with different population selection or update strategies, i.e., MOEA/D-M2M [38], MOEA/D-STM [59], MOEA/D-AGR [51], MOEA/D-IR [37], MOEA/D-DE [58], and MOEA/D-ACD [54]. Please note that MOEA/D-M2M, MOEA/D-AGR, and MOEA/D-CSU are run in Matlab, while the rest algorithms are realized in jMetal [70]. The parameters in all the compared algorithms were set as recommended in their original references. The crossover mutation probability in our algorithm was set to 1.0 and $1/n$ to run (6) and (7), respectively, as suggested in [38].

TABLE 1: IGD comparison of results of MOEA/D-CSU and six competitors on all the *MOP* and *IMB* test problems.

		MOEA/D-M2M	MOEA/D-STM	MOEA/D-AGR	MOEA/D-IR	MOEA/D-DE	MOEA/D-ACD	MOEA/D-CSU
<i>MOP1</i>	Mean (Std)	2.26E-2 – (1.52E-3)	3.46E-1 – (2.27E-2)	2.69E-2 – (3.53E-3)	2.63E-2 – (2.72E-3)	3.62E-1 – (7.47E-3)	2.72E-2 – (2.33E-3)	1.66E-2 (4.38E-4)
<i>MOP2</i>	Mean (Std)	7.40E-3 – (5.09E-4)	2.95E-1 – (7.83E-2)	6.79E-2 – (7.69E-2)	6.02E-2 – (6.46E-2)	2.77E-1 – (7.19E-2)	3.13E-2 – (6.67E-2)	6.87E-3 (2.29E-3)
<i>MOP3</i>	Mean (Std)	1.04E-2 – (2.20E-3)	1.55E-1 – (3.43E-2)	3.85E-2 – (5.83E-2)	1.43E-2 – (1.91E-2)	1.23E-1 – (4.82E-2)	1.36E-2 – (2.06E-2)	8.14E-3 (3.76E-3)
<i>MOP4</i>	Mean (Std)	4.57E-3 = (3.39E-4)	3.00E-1 – (2.99E-2)	3.52E-2 – (3.23E-2)	1.07E-1 – (8.46E-2)	2.76E-1 – (3.24E-2)	5.48E-2 – (5.74E-2)	4.47E-3 (1.22E-4)
<i>MOP5</i>	Mean (Std)	2.00E-2 – (6.91E-4)	3.12E-1 – (2.78E-2)	2.31E-2 – (3.04E-3)	2.16E-2 – (2.41E-3)	3.16E-1 – (8.02E-3)	2.39E-2 – (2.31E-3)	1.51E-2 (3.57E-4)
<i>MOP6</i>	Mean (Std)	4.96E-2 – (1.58E-3)	2.90E-1 – (2.24E-2)	4.90E-2 – (2.36E-3)	4.84E-2 – (2.80E-3)	2.90E-1 – (2.56E-2)	5.02E-2 – (2.39E-3)	3.30E-2 (2.80E-4)
<i>MOP7</i>	Mean (Std)	7.93E-2 – (4.69E-3)	3.51E-1 – (8.93E-8)	1.77E-1 – (3.16E-2)	1.92E-1 – (1.86E-2)	3.38E-1 – (2.27E-2)	2.33E-1 – (2.47E-2)	4.65E-2 (3.35E-4)
<i>IMB1</i>	Mean (Std)	1.04E-2 – (6.11E-4)	1.05E-1 – (8.78E-2)	1.23E-2 – (1.33E-3)	1.18E-2 – (1.96E-3)	1.76E-1 – (7.14E-2)	1.28E-2 – (9.52E-4)	7.53E-3 (1.36E-4)
<i>IMB2</i>	Mean (Std)	1.19E-2 – (5.46E-4)	1.62E-1 – (1.16E-2)	4.70E-2 – (1.79E-2)	6.64E-2 – (2.46E-2)	1.67E-1 – (1.11E-2)	5.85E-2 – (1.84E-2)	8.92E-3 (2.16E-4)
<i>IMB3</i>	Mean (Std)	1.78E-2 – (7.66E-4)	2.82E-1 – (3.55E-2)	2.56E-2 – (3.48E-3)	2.37E-2 – (2.73E-3)	2.84E-1 – (1.42E-2)	2.26E-2 – (1.88E-3)	1.19E-2 (3.16E-4)
<i>IMB4</i>	Mean (Std)	4.05E-2 – (1.59E-3)	1.33E-1 – (6.92E-3)	2.53E-2 + (5.04E-4)	2.39E-2 + (2.57E-4)	1.37E-1 – (6.97E-3)	2.82E-2 = (5.45E-4)	2.79E-2 (2.19E-4)
<i>IMB5</i>	Mean (Std)	5.67E-2 – (9.92E-3)	8.40E-2 – (2.47E-5)	8.39E-2 – (1.37E-5)	7.90E-2 – (4.61E-3)	9.12E-2 – (5.74E-5)	7.41E-2 – (7.50E-3)	3.28E-2 (1.21E-4)
<i>IMB6</i>	Mean (Std)	3.85E-2 – (5.24E-3)	4.62E-2 – (1.24E-4)	4.61E-2 – (2.12E-4)	4.70E-2 – (1.45E-4)	5.22E-2 – (6.28E-4)	2.50E-2 – (3.10E-4)	2.37E-2 (8.71E-5)
<i>IMB7</i>	Mean (Std)	1.08E-2 – (5.89E-4)	2.98E-2 – (2.28E-4)	2.49E-2 – (8.30E-3)	2.76E-2 – (2.76E-2)	2.99E-2 – (1.95E-4)	2.85E-2 – (6.95E-3)	7.92E-3 (3.05E-4)
<i>IMB8</i>	Mean (Std)	1.24E-2 – (5.30E-4)	3.45E-2 – (9.77E-4)	2.61E-2 – (1.06E-2)	3.34E-2 – (5.35E-3)	3.38E-2 – (3.58E-3)	3.52E-2 – (4.19E-3)	9.40E-3 (3.81E-4)
<i>IMB9</i>	Mean (Std)	1.35E-2 – (6.62E-4)	3.84E-2 – 2.43E-4	3.25E-2 – (9.16E-3)	3.79E-2 – (3.42E-3)	3.83E-2 – (2.29E-4)	3.96E-2 – (1.20E-3)	1.18E-2 (6.18E-4)
<i>IMB10</i>	Mean (Std)	4.79E-2 – (8.68E-4)	2.82E-2 + (1.80E-3)	3.29E-2 + (9.61E-4)	2.95E-2 + (8.55E-4)	3.63E-2 – (1.95E-3)	3.17E-2 + (7.63E-4)	3.57E-2 (3.02E-4)
<i>Better/Worse/Similar</i>		16/0/1	16/1/0	15/2/0	15/2/0	17/0/0	15/1/1	

4.2. Performance Measures. In this paper, in order to provide a comprehensive assessment on the performance of all the competitors, two widely used performance indicators, i.e., inverted generational distance (IGD) [71] and Hypervolume (HV) [71], were adopted to measure the convergence and the diversity of the final solution set. A lower value of IGD and a larger value of HV indicate a better performance to approach the true PF and to spread solutions uniformly along the true PF. When computing the IGD indicator, no less than 500 sampling points from the true PF were used. For the HV calculation, the reference points were set to 1.1 times the upper bound of the PF, i.e., (1.1, 1.1) for biobjective problems and to (1.1, 1.1, 1.1) for three-objective problems, as suggested in [71].

All the algorithms were run 30 times, and the mean results and standard deviations were collected for comparison. In order to have a statistically sound conclusion, Wilcoxon's rank sum test with a 5% significance level was conducted to

compare the significance of statistical difference between the results obtained by MOEA/D-CSU and other competitors.

4.3. Performance Comparisons with Six Competitive MOEA/Ds. Table 1 gives all the mean IGD results and standard deviations on *MOP* and *IMB* test problems, where the best mean result for each problem is highlighted in **boldface**. The last row “*Better/Worse/Similar*” in Table 1 summarizes the numbers of test problems in which MOEA/D-CSU, respectively, performed better than, worse than, and similarly to its competitors.

From Table 1, it is observed that MOEA/D-CSU performed best on most of the *MOP* and *IMB* test problems. As these problems were designed with complicated mathematical features that require more diversity in the population, MOEA/Ds only emphasizing the convergence will get easily trapped into local PFs. That is the reason why MOEA/D-STM and MOEA/D-DE had a poor performance obtaining

TABLE 2: HV comparison of results of MOEA/D-CSU and six competitors on all the *MOP* and *IMB* test problems.

		MOEA/D-M2M	MOEA/D-STM	MOEA/D-AGR	MOEA/D-IR	MOEA/D-DE	MOEA/D-ACD	MOEA/D-CSU
<i>MOP1</i>	Mean (Std)	6.95E-1 – (1.78E-3)	2.63E-1 – (3.61E-2)	6.92E-1 – (4.23E-3)	6.93E-1 – (2.83E-3)	2.38E-1 – (1.37E-2)	6.92E-1 – (2.44E-3)	7.03E-1 5.13E-4
<i>MOP2</i>	Mean (Std)	4.39E-1 = (3.41E-4)	2.02E-1 – (4.86E-2)	3.79E-1 – (8.12E-2)	3.86E-1 – (6.67E-2)	2.06E-1 – (4.05E-2)	4.12E-1 – (6.60E-2)	4.39E-1 3.60E-3
<i>MOP3</i>	Mean (Std)	3.40E-1 – (3.21E-3)	2.15E-1 – (2.90E-2)	3.11E-1 – (5.75E-2)	3.32E-1 – (2.64E-2)	2.41E-1 – (3.98E-2)	3.33E-1 – (2.68E-2)	3.41E-1 4.00E-3
<i>MOP4</i>	Mean (Std)	5.95E-1 = (8.24E-4)	2.90E-1 – (1.62E-2)	5.69E-1 – (3.57E-2)	4.74E-1 – (9.65E-2)	3.01E-1 – (2.31E-2)	5.36E-1 – (6.66E-2)	5.95E-1 2.48E-4
<i>MOP5</i>	Mean (Std)	6.96E-1 – (1.13E-3)	4.04E-1 – (2.08E-2)	6.94E-1 – (4.87E-3)	6.96E-1 – (2.74E-3)	4.00E-1 – (1.03E-17)	6.94E-1 – (2.72E-3)	7.03E-1 4.65E-4
<i>MOP6</i>	Mean (Std)	8.22E-1 – (1.74E-3)	6.34E-1 – (2.48E-2)	8.24E-1 – (3.62E-3)	8.28E-1 – (2.42E-3)	6.34E-1 – (2.97E-2)	8.28E-1 – (2.21E-3)	8.40E-1 2.71E-4
<i>MOP7</i>	Mean (Std)	5.41E-1 – (4.43E-3)	4.07E-1 – (1.89E-7)	4.97E-1 – (3.39E-2)	4.94E-1 – (4.91E-3)	4.12E-1 – (1.02E-2)	4.94E-1 – (1.45E-2)	5.45E-1 5.26E-4
<i>IMB1</i>	Mean (Std)	7.11E-1 – (7.41E-4)	6.15E-1 – (8.87E-2)	7.09E-1 – (1.56E-3)	7.10E-1 – (2.04E-3)	5.43E-1 – (6.95E-2)	7.09E-1 – (1.00E-3)	7.15E-1 1.56E-4
<i>IMB2</i>	Mean (Std)	5.70E-1 – (7.03E-4)	3.91E-1 – (9.14E-3)	5.25E-1 – (2.33E-2)	5.01E-1 – (3.02E-2)	3.90E-1 – (7.18E-3)	5.10E-1 – (2.34E-2)	5.74E-1 2.90E-4
<i>IMB3</i>	Mean (Std)	3.29E-1 – (7.63E-4)	1.33E-1 – (2.20E-2)	3.24E-1 – (1.97E-3)	3.23E-1 – (2.62E-3)	1.28E-1 – (7.18E-3)	3.24E-1 – (1.69E-3)	3.35E-1 3.90E-4
<i>IMB4</i>	Mean (Std)	8.23E-1 – (2.44E-3)	7.71E-1 – (3.03E-3)	8.49E-1 + (7.80E-4)	8.53E-1 + (4.54E-4)	7.64E-1 – (3.10E-3)	8.46E-1 = (9.63E-4)	8.46E-1 2.45E-4
<i>IMB5</i>	Mean (Std)	5.48E-1 – (6.76E-3)	5.45E-1 – (2.31E-5)	5.45E-1 – (2.34E-5)	5.43E-1 – (7.91E-4)	5.32E-1 – (2.43E-4)	5.42E-1 – (8.46E-4)	5.71E-1 2.71E-4
<i>IMB6</i>	Mean (Std)	8.31E-1 – (3.07E-3)	8.41E-1 – (3.53E-5)	8.40E-1 – (4.28E-5)	8.40E-1 – (1.44E-4)	8.32E-1 – (1.25E-3)	8.52E-1 – (3.58E-4)	8.53E-1 1.11E-4
<i>IMB7</i>	Mean (Std)	7.09E-1 – (8.19E-4)	6.95E-1 – (5.16E-4)	6.99E-1 – (6.91E-3)	6.97E-1 – (5.58E-3)	6.95E-1 – (4.67E-4)	6.96E-1 – (5.87E-3)	7.14E-1 5.14E-4
<i>IMB8</i>	Mean (Std)	5.67E-1 – (8.33E-4)	5.45E-1 – (1.14E-3)	5.55E-1 – (1.18E-2)	5.46E-1 – (6.04E-3)	5.46E-1 – (3.94E-3)	5.44E-1 – (4.77E-3)	5.73E-1 5.41E-4
<i>IMB9</i>	Mean (Std)	3.31E-1 – (1.14E-3)	3.13E-1 – (8.88E-4)	3.18E-1 – (6.82E-3)	3.14E-1 – (2.49E-3)	3.14E-1 – (7.83E-4)	3.11E-1 – (2.02E-3)	3.34E-1 9.06E-4
<i>IMB10</i>	Mean (Std)	8.17E-1 – (2.19E-3)	8.48E-1 + (1.35E-3)	8.39E-1 + (1.23E-3)	8.46E-1 + (1.01E-3)	8.34E-1 – (3.05E-3)	8.43E-1 + (9.98E-4)	8.36E-1 3.35E-4
<i>Better/Worse/Similar</i>		15/0/2	16/1/0	15/2/0	15/2/0	17/0/0	15/1/1	

IGD results mostly under an accuracy of 10^{-1} . Other competitors, e.g., MOEA/D-M2M, MOEA/D-AGR, MOEA/D-ACD, and MOEA/D-IR were designed to put more emphasis on diversity, and they performed much better, obtaining IGD results mostly with an accuracy of 10^{-2} , which is still not so close to the true PFs. Since the proposed CSU strategy was used in MOEA/D-CSU, it strongly emphasizes diversity but impacts the convergence less. MOEA/D-CSU properly converged to the true PFs, obtaining IGD results under an accuracy of 10^{-3} for half of test problems adopted. On *MOP1* to *MOP7*, MOEA/D-CSU gets the all the best results. Particularly, some results are under an accuracy of 10^{-3} , while the competitors cannot converge to the PF well. To *IMB* test problems, the performance of MOEA/D-CSU is superior except for the results on *IMB4* and *IMB10*. On *IMB4*, MOEA/D-CSU is worse than MOEA/D-ARG and MOEA/D-IR, similar to MOEA/D-ACD, and better than the rest algorithms. For *IMB10*, MOEA/D-STM gets the best

result and MOEA/D-DE has a pretty good performance. It indicates that the convergence is important on *IMB10*. To summarize the experimental results on Table 1, MOEA/D-CSU is superior to the competitors on most of test problems. Seeing the last row “*Better/Worse/Similar*”, when compared to six competitive MOEA/D variants, MOEA/D-CSU can perform better on at least 15 cases and worse on at most 2 cases, which indicates our outstanding performance to balance convergence and diversity for these test problems adopted. Moreover, the HV results provided in Table 2 also confirm the advantages of MOEA/D-CSU, as MOEA/D-CSU performs best on most of the cases.

To visually show our performance, the best nondominated solution sets obtained by MOEA/D-CSU from 30 runs were plotted in Figure 2, where the circles indicate the solutions, while the lines and grids mean the true PFs on the biobjective and three-objective test problems, respectively. On the test problems with continuous PFs (i.e., *MOP1-MOP3*,

FIGURE 2: The nondominated solution sets on *MOP1-MOP7* and *IMB1-IMB10*.

MOP5-MOP7, and *IMB1-IMB10*), MOEA/D-CSU can reach the stable status and find all the optimal values for the agents. Even for *MOP4* which has a disconnected PF, MOEA/D-CSU could properly approach all the segments of the true PF. From these plots, it is reasonable to conclude that our proposed CSU strategy is very effective in tackling complicated test problems, such as *MOP* and *IMB*.

5. Conclusions and Future Work

In this paper, an enhanced decomposition-based MOEA with a CSU strategy was presented. The agent in our approach aims to optimize the subproblem, which is only allocated with the solutions that are closest to its subproblem. Thus, the number of solution in each agent may be zero or no less

than one, which helps to reflect the true diversity among the agents and to provide the correct neighboring information in evolution. To ensure diversity, the offspring in each agent are only allowed to update its original solutions. In the case that the agent has no solution, one solution will be assigned in priority once there are offspring generated closest to its subproblem. Another agent with the largest number of solutions will remove one solution showing the worst convergence. Therefore, for each agent, this approach may enhance its diversity or convergence, but will not deteriorate either of them. After assessing its performance on two complicated test suites (*MOP* and *IMB*), the experimental results confirmed the superiority of MOEA/D-CSU over six competitive MOEA/Ds with other population selection or update strategies.

In our future work, the performance of this CSU strategy will be further studied to improve the way in which it reaches the stable status. One possible path is to embed an adaptive adjustment strategy for generating weight vectors in MOEA/D-CSU, which can cooperate with the CSU strategy to attain real-diversity when dealing with disconnected or incomplete PFs. The application of MOEA/D-CSU in some real-world problems will also be our future research direction.

Data Availability

The source code and source data can be provided by contacting with the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by Shenzhen Technology Plan under Grant JCYJ20170817102218122, the National Natural Science Foundation of China under Grants 61876110, 61836005, and 61402291, the Joint Funds of the National Natural Science Foundation of China under Key Program Grant U1713212, and the Natural Science Foundation of Guangdong Province under Grant 2017A030313338. Also, this work was supported by the National Engineering Laboratory for Big Data System Computing Technology.

References

- [1] K. Miettinen, *Nonlinear Multiobjective Optimization*, Kluwer Academic Publishers, Norwell, Mass, USA, 1999.
- [2] Q. Lin, X. Wang, B. Hu et al., "Multiobjective personalized recommendation algorithm using extreme point guided evolutionary computation," *Complexity*, vol. 2018, Article ID 1716352, 18 pages, 2018.
- [3] X. Li, D. Zhou, Q. Pan, Y. Tang, and J. Huang, "Weapon-target assignment problem by multiobjective evolutionary algorithm based on decomposition," *Complexity*, vol. 2018, Article ID 8623051, 19 pages, 2018.
- [4] M. Eskandari Nasab, I. Maleksaedi, M. Mohammadi, and N. Ghadimi, "A new multiobjective allocator of capacitor banks and distributed generations using a new investigated differential evolution," *Complexity*, vol. 19, no. 5, pp. 40–54, 2014.
- [5] Z. Gao, X. Cui, Y. Duan, Z. Jun, and Z. Peng, "Using MOPSO for optimizing randomized response schemes in privacy computing," *Mathematical Problems in Engineering*, vol. 2018, Article ID 7846547, 16 pages, 2018.
- [6] X. Li, J. Lai, and R. Tang, "A hybrid constraints handling strategy for multiconstrained multiobjective optimization problem of microgrid economical/environmental dispatch," *Complexity*, vol. 2017, Article ID 6249432, 12 pages, 2017.
- [7] K. Deb, *Multiobjective Optimization Using Evolutionary Algorithms*, Wiley, New York, NY, USA, 2001.
- [8] S. Huband, L. Barone, L. While, and P. Hingston, "A scalable multi-objective test problem toolkit," *Lecture Notes in Computer Science*, vol. 3410, pp. 280–295, Springer, Berlin, Germany, 2005.
- [9] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multiobjective optimization," in *Evolutionary Multiobjective Optimization*, Advanced Information and Knowledge Processing Series, pp. 105–145, Springer, Berlin, Germany, 2005.
- [10] Q. Zhang, A. Zhou, S. Zhao, P. Suganthan, W. Liu, and S. Tiwari, "Multiobjective optimization test instances for the CEC 2009 special session and competition," Tech. Rep. CES-887, University of Essex and Nanyang Technological University, Essex, U.K./Singapore, 2008.
- [11] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: empirical results," *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, 2000.
- [12] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "Test problems for large-scale multiobjective and many-objective optimization," *IEEE Transactions on Cybernetics*, vol. 47, no. 12, pp. 4108–4121, 2017.
- [13] R. Cheng, M. Li, Y. Tian et al., "A benchmark test suite for evolutionary many-objective optimization," *Complex and Intelligent Systems*, vol. 3, no. 1, pp. 67–81, 2017.
- [14] S. Yang, S. Jiang, and Y. Jiang, "Improving the multiobjective evolutionary algorithm based on decomposition with new penalty schemes," *Soft Computing*, vol. 21, no. 16, pp. 4677–4691, 2017.
- [15] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multiobjective optimization," *Evolutionary Multiobjective Optimization*, pp. 105–145, 2005.
- [16] S. Huband, L. Barone, L. While, and P. Hingston, "A scalable multi-objective test problem toolkit," in *Lecture Notes in Computer Science*, vol. 3410, pp. 280–295, Springer, Berlin, Germany, 2005.
- [17] M. Elarbi, S. Bechikh, A. Gupta, L. Ben Said, and Y.-S. Ong, "A new decomposition-based nsga-ii for many-objective optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 48, no. 7, pp. 1191–1210, 2018.
- [18] Q. Lin, J. Chen, Z.-H. Zhan et al., "A hybrid evolutionary immune algorithm for multiobjective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 5, pp. 711–729, 2016.
- [19] Y. Y. Tan, Y. C. Jiao, H. Li, and X. K. Wang, "A modification to MOEA/D-DE for multiobjective optimization problems with complicated Pareto sets," *Information Sciences*, vol. 213, pp. 14–38, 2012.
- [20] K. Li, K. Deb, Q. Zhang, and Q. Zhang, "Efficient non-domination level update method for steady-state evolutionary multiobjective optimization," *IEEE Transactions on Cybernetics*, vol. 47, no. 9, pp. 2838–2849, 2017.
- [21] J. Knowles and D. Corne, "The pareto archived evolution strategy: a new baseline algorithm for multiobjective optimisation," in *Proceedings of the 1999 Congress on Evolutionary Computation-CEC '99*, vol. 1, pp. 98–105, WA, USA, July 1999.
- [22] J. Bader and E. Zitzler, "HypE: an algorithm for fast hypervolume-based many-objective optimization," *Evolutionary Computation*, vol. 19, no. 1, pp. 45–76, 2011.
- [23] S. Rostami, F. Neri, and M. Epitropakis, "Progressive preference articulation for decision making in multi-objective optimisation problems," *Integrated Computer-Aided Engineering*, vol. 24, no. 4, pp. 315–335, 2017.
- [24] S. Rostami and F. Neri, "Covariance matrix adaptation pareto archived evolution strategy with hypervolume-sorted adaptive grid algorithm," *Integrated Computer-Aided Engineering*, vol. 23, no. 4, pp. 313–329, 2016.

- [25] S. Rostami and F. Neri, "A fast hypervolume driven selection mechanism for many-objective optimisation problems," *Swarm and Evolutionary Computation*, vol. 34, pp. 50–67, 2017.
- [26] E. Zitzler and S. Künzli, "Indicator-based selection in multiobjective search," in *Parallel Problem Solving from Nature—PPSN VIII*, vol. 3242 of *Lecture Notes in Computer Science*, pp. 832–842, Springer, Berlin, Germany, 2004.
- [27] D. Brockhoff, T. Wagner, and H. Trautmann, "On the properties of the R2 indicator," in *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation*, pp. 465–472, ACM, Philadelphia, Pa, USA, July 2012.
- [28] K. Bringmann and T. Friedrich, "An efficient algorithm for computing hypervolume contributions," *Evolutionary Computation*, vol. 18, no. 3, pp. 383–402, 2010.
- [29] S. Jiang, J. Zhang, Y.-S. Ong, A. N. Zhang, and P. S. Tan, "A simple and fast hypervolume indicator-based multiobjective evolutionary algorithm," *IEEE Transactions on Cybernetics*, vol. 45, no. 10, pp. 2202–2213, 2015.
- [30] Z. Wang, Q. Zhang, and H. Li, "Balancing convergence and diversity by using two different reproduction operators in MOEA/D: some preliminary work," in *Proceedings of the 2015 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 2849–2854, Kowloon, Hong Kong, October 2015.
- [31] F. Gu and Y.-M. Cheung, "Self-organizing map-based weight design for decomposition-based many-objective evolutionary algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 2, pp. 211–225, 2018.
- [32] L. Ke, Q. Zhang, and R. Battiti, "MOEA/D-ACO: a multiobjective evolutionary algorithm using decomposition and AntColony," *IEEE Transactions on Cybernetics*, vol. 43, no. 6, pp. 1845–1859, 2013.
- [33] S. Jiang and S. Yang, "An improved multiobjective optimization evolutionary algorithm based on decomposition for complex pareto fronts," *IEEE Transactions on Cybernetics*, vol. 46, no. 2, pp. 421–437, 2015.
- [34] H. Sato, "Inverted PBI in MOEA/D and its impact on the search performance on multi and many-objective optimization," in *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*, pp. 645–652, Vancouver, Canada, July 2014.
- [35] Y. Su, J. Wang, L. Ma, X. Wang, Q. Lin, and J. Chen, "A novel many-objective optimization algorithm based on the hybrid angle-encouragement decomposition," in *Lecture Notes in Computer Science*, vol. 10956, pp. 47–53, Springer International Publishing, Cham, Switzerland, 2018.
- [36] H. Li, Q. Zhang, and J. Deng, "Biased multiobjective optimization and decomposition algorithm," *IEEE Transactions on Cybernetics*, vol. 47, no. 1, pp. 52–66, 2017.
- [37] K. Li, S. Kwong, Q. Zhang, and K. Deb, "Interrelationship-based selection for decomposition multiobjective optimization," *IEEE Transactions on Cybernetics*, vol. 45, no. 10, pp. 2076–2088, 2015.
- [38] H.-L. Liu, F. Gu, and Q. Zhang, "Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 3, pp. 450–455, 2014.
- [39] Q. Zhang and H. Li, "MOEA/D: a multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.
- [40] Y. T. Qi, X. L. Ma, F. Liu, L. C. Jiao, J. Y. Sun, and J. S. Wu, "MOEA/D with adaptive weight adjustment," *Evolutionary Computation*, vol. 22, no. 2, pp. 231–264, 2014.
- [41] H.-L. Liu, L. Chen, Q. Zhang, and K. Deb, "Adaptively allocating search effort in challenging many-objective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 3, pp. 433–448, 2018.
- [42] X. Cai, Z. Mei, and Z. Fan, "A decomposition-based many-objective evolutionary algorithm with two types of adjustments for direction vectors," *IEEE Transactions on Cybernetics*, vol. 48, no. 8, pp. 2335–2348, 2018.
- [43] M. Asafuddoula, H. K. Singh, and T. Ray, "An enhanced decomposition-based evolutionary algorithm with adaptive reference vectors," *IEEE Transactions on Cybernetics*, vol. 48, no. 8, pp. 2321–2334, 2018.
- [44] K. Li, K. Deb, Q. Zhang, and S. Kwong, "An evolutionary many-objective optimization algorithm based on dominance and decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 5, pp. 694–716, 2015.
- [45] Q. Lin, G. Jin, Y. Ma et al., "A diversity-enhanced resource allocation strategy for decomposition-based multiobjective evolutionary algorithm," *IEEE Transactions on Cybernetics*, vol. 48, no. 8, pp. 2388–2501, 2018.
- [46] A. Zhou and Q. Zhang, "Are all the subproblems equally important? Resource allocation in decomposition-based multiobjective evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 1, pp. 52–64, 2016.
- [47] Q. Zhang, W. Liu, and H. Li, "The performance of a new version of MOEA/D on CEC09 unconstrained MOP test instances," in *Proceedings of the 2009 IEEE Congress on Evolutionary Computation*, pp. 203–208, Trondheim, Norway, May 2009.
- [48] Q. Lin, Z. Liu, Q. Yan et al., "Adaptive composite operator selection and parameter control for multiobjective evolutionary algorithm," *Information Sciences*, vol. 339, pp. 332–352, 2016.
- [49] K. Li, A. Fialho, S. Kwong, and Q. Zhang, "Adaptive operator selection with bandits for a multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 1, pp. 114–130, 2014.
- [50] Q. Lin, C. Tang, Y. Ma, Z. Du, J. Li, and J. Chen, "A novel adaptive control strategy for decomposition-based multiobjective algorithm," *Computers & Operations Research*, vol. 78, pp. 94–107, 2017.
- [51] Z. Wang, Q. Zhang, A. Zhou, M. Gong, and L. Jiao, "Adaptive replacement strategies for MOEA/D," *IEEE Transactions on Cybernetics*, vol. 46, no. 2, pp. 474–486, 2016.
- [52] R. Wang, J. Xiong, H. Ishibuchi, G. Wu, and T. Zhang, "On the effect of reference point in MOEA/D for multi-objective optimization," *Applied Soft Computing*, vol. 58, pp. 25–34, 2017.
- [53] M. Wu, K. Li, S. Kwong, Y. Zhou, and Q. Zhang, "Matching-based selection with incomplete lists for decomposition multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 5, pp. 714–730, 2017.
- [54] L. Wang, Q. Zhang, A. Zhou, M. Gong, and L. Jiao, "Constrained subproblems in a decomposition-based multiobjective evolutionary algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 3, pp. 475–480, 2016.
- [55] X. Ma, Q. Zhang, G. Tian, J. Yang, and Z. Zhu, "On tchebycheff decomposition approaches for multiobjective evolutionary optimization," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 2, pp. 226–244, 2018.
- [56] L. Cai, S. Qu, and G. Cheng, "Two-archive method for aggregation-based many-objective optimization," *Information Sciences*, vol. 422, pp. 305–317, 2018.

- [57] X. Cai, Z. Yang, Z. Fan, and Q. Zhang, "Decomposition-based-sorting and angle-based-selection for evolutionary multiobjective and many-objective optimization," *IEEE Transactions on Cybernetics*, vol. 47, no. 9, pp. 2824–2837, 2017.
- [58] H. Li and Q. Zhang, "Multiobjective optimization problems with complicated pareto sets, MOEA/D and NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 284–302, 2009.
- [59] K. Li, Q. Zhang, S. Kwong, M. Li, and R. Wang, "Stable matching-based selection in evolutionary multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 6, pp. 909–923, 2014.
- [60] Y. Yuan, H. Xu, B. Wang, B. Zhang, and X. Yao, "Balancing convergence and diversity in decomposition-based many-objective optimizers," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 2, pp. 180–198, 2016.
- [61] R. Wang, H. Ishibuchi, Y. Zhang, X. Zheng, and T. Zhang, "On the effect of localized PBI method in MOEA/D for multiobjective optimization," in *Proceedings of the 2016 IEEE Symposium Series on Computational Intelligence*, pp. 645–652, Athens, Greece, 2016.
- [62] R. Wang, Z. Zhou, H. Ishibuchi, T. Liao, and T. Zhang, "Localized weighted sum method for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 3–18, 2018.
- [63] R. Wang, Q. Zhang, and T. Zhang, "Decomposition-based algorithms using pareto adaptive scalarizing methods," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 6, pp. 821–837, 2016.
- [64] M. Ming, R. Wang, Y. Zha, and T. Zhang, "Pareto adaptive penalty-based boundary intersection method for multi-objective optimization," *Information Sciences*, vol. 414, pp. 158–174, 2017.
- [65] M. Wu, K. Li, S. Kwong, and Q. Zhang, "Evolutionary many-objective optimization based on adversarial decomposition," *IEEE Transactions on Cybernetics*, pp. 1–12, 2018.
- [66] Y. Zhang, Y. Gong, T. Gu et al., "DECAL: decomposition-based coevolutionary algorithm for many-objective optimization," *IEEE Transactions on Cybernetics*, vol. 49, no. 1, pp. 27–41, 2019.
- [67] M. Wu, K. Li, S. Kwong, Q. Zhang, and J. Zhang, "Learning to decompose: a paradigm for decomposition-based multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, p. 1, 2018.
- [68] H.-L. Liu and X. Q. Li, "The multiobjective evolutionary algorithm based on determined weight and sub-regional search," in *Proceedings of the 2009 IEEE Congress on Evolutionary Computation*, pp. 1928–1934, IEEE, Trondheim, Norway, May 2009.
- [69] H. Liu, L. Chen, K. Deb, and E. D. Goodman, "Investigating the effect of imbalance between convergence and diversity in evolutionary multi-objective algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 3, pp. 408–425, 2017.
- [70] J. J. Durillo, A. J. Nebro, and E. Alba, "The jmetal framework for multi-objective optimization: design and architecture," in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1–8, Barcelona, Spain, 2010.
- [71] K. Li, R. Wang, T. Zhang, and H. Ishibuchi, "Evolutionary many-objective optimization: a comparative study of the state-of-the-art," *IEEE Access*, vol. 6, pp. 26194–26214, 2018.

Submit your manuscripts at
www.hindawi.com