Information Sciences 511 (2020) 18–35

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

A Multiobjective multifactorial optimization algorithm based on decomposition and dynamic resource allocation strategy

Shuangshuang Yao[a], Zhiming Dong[b], Xianpeng Wang[c,*], Lei Ren[d]

[a] *College of Information Science and Engineering, Northeastern University, Shenyang, 110819, China*
[b] *Key Laboratory of Data Analytics and Optimization for Smart Industry, Northeastern University, Ministry of Education, Northeastern University, Shenyang, 110819, China*
[c] *Liaoning Engineering Laboratory of operation Analytics and Optimization for Smart Industry, Liaoning Key Laboratory of Manufacturing System and Logistics, Institute of Industrial & Systems Engineering, Northeastern University, Shenyang, 110819, China*
[d] *School of Automation and Electrical Engineering, Beihang University, 100191, China*

A R T I C L E I N F O

Article history:
Received 8 April 2019
Revised 18 September 2019
Accepted 23 September 2019
Available online 24 September 2019

Keywords:
Multiobjective multifactorial optimization
Evolutionary multitasking
Decomposition
Dynamic resource allocating

A B S T R A C T

Multiobjective multifactorial optimization (MO-MFO), *i.e.*, multiple multiobjective tasks are simultaneously optimized by a single population, has received considerable attention in recent years. Traditional algorithms for the MO-MFO usually allocate equal computing resources to each task, however, this may not be reasonable due to the fact that different tasks usually have different degrees of difficulty. Motivated by the idea that the limited computing resources should be adaptively allocated to different tasks according to their difficulties, this paper proposes an algorithm for the MO-MFO based on decomposition and dynamic resource allocation strategy (denoted as MFEA/D-DRA). In the MFEA/D-DRA, each multiobjective optimization task is firstly decomposed into a series of single-objective subproblems. Thereafter, a single population is used to evolve all the single-objective subproblems. In the process of evolution, subproblems with fast evolution rate will have the opportunity to get more rewards, *i.e.*, computing resources. The evolution rate is measured by a utility function and updated periodically. Moreover, different multiobjective optimization tasks can communicate with each other according to a random mating probability. Finally, a set of evenly distributed approximate Pareto optimal solutions is obtained for each multiobjective optimization task. The statistical analysis of experimental results illustrates the superiority of the proposed MFEA/D-DRA algorithm on a variety of benchmark MO-MFO problems.

1. Introduction

Multitasking optimization refers to that multiple tasks are optimized simultaneously [20]. The key motivation behind multitasking optimization is that if the general characteristics of the optimal solutions or the function landscapes of the optimization tasks are related with each other, the search pace of each task can be accelerated by information transfer between different tasks. In particular, from the perspective of engineering design practice, the potential of automated transfer,

* Corresponding author.
E-mail addresses: yss_neu@163.com (S. Yao), dongzm@stumail.neu.edu.cn (Z. Dong), wangxianpeng@ise.neu.edu.cn, neu_wxp@163.com (X. Wang), renlei@buaa.edu.cn (L. Ren).

https://doi.org/10.1016/j.ins.2019.09.058
0020-0255/© 2019 Elsevier Inc. All rights reserved.

as well as the reuse and migration of knowledge, is considered invaluable [21]. Different from the multiobejctive optimization that simultaneously optimizes multiple conflicting single-objective problems to achieve a series of trade-off solutions, the target of multitasking optimization is to use the potential information between different tasks to accelerate convergence speed, and finally to find the approximate optimal solutions for each task.

The research on multitasking optimization has been very diverse since it was proposed, such as single-objective multitasking optimization [3,17,18,45], multiobjective multitasking optimization [10,21,32], expensive multitasking optimization [14], manytasking optimization [9,26], and the applications of multitasking optimization in bi-level programming [19], capacitated vehicle routing problem [47], multi-step chaotic time series prediction [7], multiple sparse reconstruction [23], neural networks [6], modular extremal learning machine [38], and so forth.

Multiobjective multifactorial (multitasking) optimization (MO-MFO) problem means that each task is a multiobjective optimization problem (MOP). A naive way to deal with MO-MFO problem is to solve each MOP separately. In recent decades, evolutionary algorithms (EAs) have achieved remarkable results in solving MOPs. These algorithms can be roughly divided into the following four categories: dominance-based algorithms [12,27,37,39,40], indication-based algorithms [2,4,48], decomposition-based algorithms [22,25,44,49] and learning-based algorithms [36,42]. Although these population-based algorithms show superior performance when solving a single MOP, they cannot be used to solve multiple distinct MOPs at the same time, let alone explore and use the knowledge between different MOPs, to accelerate the problem solving.

For solving the MO-MFO problem, in addition to exploring and utilizing the useful information between different tasks to accelerate optimization, it is also important to reasonably allocate computing resources between different tasks because different tasks may often have different degrees of difficulty. In this paper, we propose a multiobjective multifactorial evolutionary algorithm based on decomposition and dynamic resource allocation strategy (MFEA/D-DRA). The main contributions of this paper are as follows.

- A decomposition strategy that converts the MO-MFO problem into a number of single-objective optimization subproblems is proposed, *i.e.*, each MOP task is decomposed into a series of single-objective optimization subproblems, and a single population with multiple factors distributed in unified space is used to optimize all these single-objective subproblems simultaneously.
- To efficiently solve these single-objective optimization problems, a dynamic resource allocation strategy with multifactorial environment is proposed. The computational resources are allocated according to the evolution rate of these single-objective subproblems in each generation. That is, a single-objective problem with fast evolution rate should obtain more rewards, *i.e.*, computational resources, and such rewards are updated periodically. In each generation, the subproblems with more rewards will have higher opportunity to generate better solutions that can update the population.
- The proposed algorithm is applied to solve a multiobjective multifactorial operation optimization problem of continuous annealing process based on data analytics, and the computational results show its superiority.

The following parts of this paper is organized as follows. Section 2 first reviews the related work including the basic concepts of multiobjective and MO-MFO, the multiobjective multifactorial evolutionary algorithm (MO-MFEA), the decomposition method of MOP and the dynamic resource allocation strategy, and then describes the motivation of our proposed algorithm. Section 3 presents the proposed MFEA/D-DRA algorithm in detail. Experimental studies and the parameter analysis of the proposed algorithm are carried out in Section 4. Finally, the conclusions and future work of this paper are discussed in Section 5.

2. Related work

2.1. Multiobjective optimization

Without loss of generality, the MOP with minimizing all objectives can be mathematically expressed as follow [30].

$$\begin{aligned} &minimize\ \mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_M(\mathbf{x})) \\ &s.t. \mathbf{x} \in \mathbf{\Omega} \end{aligned} \tag{1}$$

where $\mathbf{\Omega} \subset \mathbb{R}^D$ represents the decision space, $\mathbf{x} = (x_1, x_2, \dots, x_D) \in \mathbf{\Omega}$ is a D-dimensional decision variable, and $\mathbf{F} : \mathbf{\Omega} \to \mathbb{R}^M$ consists of M conflicting objective functions.

The target of multiobjective optimization is to find a representative Pareo optimal front to help decision-makers understand the conflict between different objectives and then make optimal tradeoffs. Some basic definitions of multiobjective optimization are as follows.

Definition 2.1. Dominance : Supposing that $\mathbf{x_1}, \mathbf{x_2} \in \mathbb{R}$, if $\forall\ i \in \{1, 2, \dots, M\}$, $f_i(\mathbf{x_1}) \le f_i(\mathbf{x_2})$, and $\exists\ j \in \{1, 2, \dots, M\}$ satisfying $f_j(\mathbf{x_1}) < f_j(\mathbf{x_2})$, then it is said that $\mathbf{x_1}$ dominates $\mathbf{x_2}$, denoted as $\mathbf{x_1} \succ \mathbf{x_2}$.

Definition 2.2. Pareto optimal solution : For a solution $\mathbf{x^*}$, if there does not exist any other solution $\mathbf{x} \in \mathbf{\Omega}$ such that $\mathbf{x} \succ \mathbf{x^*}$, then $\mathbf{x^*}$ is called the Pareto optimal solution.

Definition 2.3. Pareto optimal set : The set of all Pareto optimal solutions for an MOP is called the Pareto optimal set (PS).

Definition 2.4. Pareto front : The set of mappings of the PS in the objective space, $\{\mathbf{F}(\mathbf{x}) \in \mathbb{R}^M | \mathbf{x} \in PS\}$, is called the Pareto front (PF).

2.2. Multiobjective multifactorial optimization

In general, a multiobjective multifactorial optimization problem with K distinct tasks and minimization of objectives can be mathematically represented by (2) [21].

$$\begin{aligned} &\{\mathbf{x_1}, \mathbf{x_2}, \dots, \mathbf{x_K}\} = \arg\min\{\mathbf{F}_1(\mathbf{x}), \mathbf{F}_2(\mathbf{x}), \dots, \mathbf{F}_K(\mathbf{x})\} \\ &s.t. \quad \mathbf{x}_i \in \mathbf{X}_i, \quad i \in \{1, 2, \dots, K\} \end{aligned} \tag{2}$$

where $\mathbf{x}_i$ is the Pareto optimal set of task i, $\mathbf{X}_i \subset \mathbb{R}^{D_i}$ is the decision space, D_i represents the number of the decision variables of the ith task, and $\mathbf{F}_i : \mathbf{X}_i \to \mathbb{R}^{M_i}$ consists of M_i conflicting objective functions, and $i \in \{1, 2, \dots, K\}$.

An important feature of multifactorial optimization is to optimize multiple tasks by a single population, and each task can be regarded as a factor affecting the evolution process of the population. Therefore, these tasks act together as a multifactorial environment that affects the evolution of the population. For an individual p_i in population $\mathcal{P}$, some definitions are shown below (the jth task is denoted as $\mathcal{T}_j$).

Definition 2.5. Factorial Rank : the factorial rank r^i_j of individual p_i in $\mathcal{T}_j$ is the index of the sorted preference in decreasing order of $\mathcal{T}_j$.

Definition 2.6. Skill Factor : the skill factor τ_i refers to the task that is associated to the individual p_i. If p_i is evaluated for all tasks, then the skill factor of p_i can be written as $\tau_i = \arg\min_{j \in \{1,2,\dots K\}}\{r^i_j\}$.

Definition 2.7. Scalar Fitness : the scalar fitness of individual p_i can be given by $\varphi_i = 1/r^i_{\tau_i}$.

In Definition 2.5, for a multiobjective problem, the preference ordering of the population is not as simple as the single-objective problem, which is just sorted by the objective function values. In order to make any two individuals p_1 and p_2 more explicitly comparable, two attributes, *i.e.*, NF (non-dominated sorting fronts) and CD (crowding distance) defined in [12], are used. If either of the following conditions is met, then the performance of p_2 is considered to be superior to that of p_1.

- $\mathrm{NF}_2 < \mathrm{NF}_1$
- $\mathrm{NF}_2 = \mathrm{NF}_1$ and $\mathrm{CD}_2 > \mathrm{CD}_1$

2.3. Multiobjective multifactorial evolutionary algorithm

Based on the NSGA-II [12] and borrowing strategies from the field of memetic computing [8], Gupta et al. proposed a multiobjective multifactorial evolutionary algorithm (MO-MFEA) [21], whose framework is presented in Algorithm 1. In MO-MFEA, three attributes including factorial rank, skill factor and scalar fitness are associated to each individual of the whole population. More precisely, the skill factor determines which task is mated for a certain individual, the factorial

Algorithm 1: Framework of MO-MFEA.

```
1  Generate the initial population $\mathcal{P}_0$ with $N$ individuals from a unified space $\mathbf{Y}$ ;
2  for each $p_i$ in $\mathcal{P}_0$ do
3      $\tau_i \leftarrow$ Set the skill factor of $p_i$ to be $\tau_i$ ;
4      Evaluate $p_i$ for $\mathcal{T}^{\tau_i}_{mop}$ ;
5  According to NS and CD, compute the scalar fitness $\varphi$ of each individual in $\mathcal{P}_0$ ;
6  $t$ = 0 ;
7  while the stopping criterion is not met do
8      $\mathcal{P}'_t \leftarrow$ Selection($\mathcal{P}_t$) ~~ // select parents ;
9      $\mathcal{O}_t \leftarrow$ Reproduction($\mathcal{P}'_t$) ~~ // generate offspring via Associate Mating ;
10     for each $o_i$ in $\mathcal{O}_t$ do
11         $\tau_i \leftarrow$ Determine skill factor of $o_i$ via Vertical Cultural Transmission ;
12         Evaluate $o_i$ for $\mathcal{T}^{\tau_i}_{mop}$ ;
13     $\mathcal{R}_t = \mathcal{P}_t \cup \mathcal{O}_t$ ;
14     Update scalar fitness of each individual in $\mathcal{R}_t$ ;
15     $\mathcal{P}_{t+1} \leftarrow$ Select $N$ fittest individuals via the sorted scalar fitness ;
16     $t$ = $t$+1 ;
```

rank indicates the preference relationship of the individuals with the same skill factor, and the scalar fitness quantifies the skill factor and the factorial rank into a single attribute. Incidentally, the scalar fitness can also be regarded as a metric of selection pressure to population evolution. It should be noted that, all individuals are encoded from the unified space $\mathbf{Y} = \{y_1, y_2, \ldots, y_{D_{multitask}}\}$, where $D_{multitask} = \max\{D_1, D_2, \ldots, D_K\}$ and y_i is randomly generated between [0,1], which is also called the random-key [21]. For any individual evaluated by a task, taking $\mathcal{T}^k$ as an example, the following formula is needed to map this individual to the decision space of $\mathcal{T}^k$ before evaluation.

$$x_j = L_j + (U_j - L_j) * y_j \tag{3}$$

where x_j represents the jth variable, U_j and L_j represent the upper and lower bounds of the jth variable of the box constraint, respectively, and $j \in \{1, 2, \ldots, D_k\}$.

In Assortative Mating phase of Algorithm 1, if the parents possess the same skill factor, *i.e.*, the meaning of each position on the chromosome is consistent, then the offspring is directly generated by the genetic algorithm (GA). On the contrary, a random mating probability (*rmp*) is used to determine whether the offspring is produced by GA or simply according to the mutation operator of the parents' own chromosomes. The random mating probability has an important effect on the knowledge exchange between different tasks. The larger *rmp* means the higher frequency of the knowledge exchange between different tasks, and vice versa. When the random mating probability is not 0, the parents with different skill factor attributes are likely to produce offspring via GA (crossover and mutation operators), and the setting of skill factors of offspring is realized by Vertical Cultural Transmission, where the skill factors of the offspring are set to be consistent with their parents in a random way.

2.4. MOEA/D with dynamic resource allocating strategy

The decomposition-based multiobjective evolutionary algorithm (MOEA/D) combines the decomposition strategy of the traditional mathematical programming with the evolutionary algorithm to form a new evolutionary algorithm framework for MOPs [43]. It decomposes an MOP into a series of single-objective subproblems by an aggregation function (also called scalarizing function) and a set of weight vectors which are uniformly distributed in the first quadrant. The weight vectors and the aggregation function are two important components of MOEA/D. The Das & Dennis method is a commonly used method for generating weight vectors [11]. Due to the promising performance and relatively stable feature of Chebyshev method for MOPs, it is the most popular aggregation function used in research, and the mathematical expression is as follows.

$$g^{tch}(\mathbf{x}|\lambda, z^*) = min\ max\ \ w_i|f_i(x) - z_i^*| \tag{4}$$

where z^* is the ideal point, $\lambda = (w_1, w_2, \ldots, w_M)$ is the weight vector, satisfying $w_i \geq 0$, and $\sum_i^M w_i = 1,\ i \in \{1, 2, \ldots, M\}$.

However, recent studies have shown that, for MOEA/D, the subproblems are not equally important (or difficulty levels are diverse) [46]. Zhang et al. [44] proposed a dynamic resource allocation strategy for MOEA/D, in which a utility function was used to determine the importance of the subproblem, and this function was updated periodically. Furthermore, the subproblems with large utility fitness were involved in population evolution by the 10-tournament strategy in each generation. Zhou et al. [46] made a further work on [44] and proposed a generalized resource allocation strategy (GRA). In MOEA/D-GRA, a probability of improvement (PoI) vector was introduced, and each subproblem was associated with the elements in this vector one-to-one. In each generation, several subproblems were selected according to the PoI vector for optimization (investment), so that the dynamic allocation of computing resources was realized. Mashwani et al. [28] proposed a hybrid crossover operator for MOEA/D algorithm with dynamic resource allocation. In [28], the crossover operators which generated the competitive offsprings were rewarded, and then dynamic usage of diverse crossover operators was realized, based on which the advantages of different crossover operators at different stages were fully utilized. Similarly, Mashwani et al. proposed a multiobjective evolutionary algorithm based on multimethods (NSGA-II & MOEA/D) with dynamic resource allocation strategy [29]. The idea of dynamic resource allocation has been also used in the single-objective multifactorial optimization. Gong et al. [18] proposed an evolutionary multitasking with dynamic resource allocation strategy based on [46] so as to efficiently utilize the limited computing resources, and the main idea was that difficult task should be allocated more computing resources.

3. Proposed algorithm

3.1. Framework

This section describes our proposed algorithm in detail. The pseudo-code of MFEA/D-DRA is shown in Algorithm 2 , and the algorithm framework is mainly composed of the following four parts.

3.1.1. Initialization

In the initialization section, each multiobjective optimization task is decomposed into a series of single-objective subproblems, as shown in lines 1 to 4 of Algorithm 2. Here, for the task $\mathcal{T}_{mop}^k$, $\mathbf{\Lambda}_k = \{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_{N_k}\}$ refers to a series of weight vectors which are given by Das & Dennis method [11], where K is the number of tasks and $N_k = N/K$. $\mathcal{P}_k = \{p_1^k, p_2^k, \ldots, p_{N_k}^k\}$

Algorithm 2: MFEA/D-DRA.

```
Input: a multiobjective multifactorial optimization problem $\mathcal{T}_{mop} = \{\mathcal{T}^1_{mop}, \mathcal{T}^2_{mop}, \ldots, \mathcal{T}^K_{mop}\}$; population size $N$; stopping
       criterion $FEs$; neighborhood size $T$; replacement size $n_r$; neighborhood selection probability $\delta$; random mating
       probability $rmp$;
Output: final population $\mathcal{P}$;
1  $\mathcal{P} \leftarrow \emptyset$ ;
2  for each $\mathcal{T}^k_{mop}$ in $\mathcal{T}_{mop}$ do
3      $(\mathbf{\Lambda}_k, B_k, \mathcal{P}_k, \mathbf{z}^*_k)$ = INIT($\mathcal{T}^k_{mop}$, $N/K$, $T$);
4      $\mathcal{P} = \mathcal{P} \cup \mathcal{P}_k$ ;
5  $\mathcal{P}_o = \mathcal{P}$,  $\mathcal{Q} \leftarrow \emptyset$,  $g = 1$ ;
6  while the stopping criterion is not met do
7      $\mathcal{S} \leftarrow$ Use 10-tournament strategy to select $N/5$ individuals based on the scalar fitness ;
8      for each $s$ in $\mathcal{S}$ do
9          $k = \tau_s$ ;
10         $id \leftarrow$ the subproblem index of $s$ in $\mathcal{T}_k$ ;
11         if $rand < \delta$ then
12             $\mathcal{Q} \leftarrow \{x_i | x_i \in B_k(id)\}$ // $B_k(id)$ represents the $id$th subproblem's neighborhood in task $k$ ;
13         else
14             $\mathcal{Q} \leftarrow \{x_i | x_i \in \mathcal{P}_k\}$ ;
15         $\hat{x} \leftarrow$ The DE-rand/1/bin operator is used to generate a child $\hat{x}$ whose parents are randomly selected from $\mathcal{Q}$ ;
16         if $rand < rmp$ then
17             $k = randomInt(1, K)$ && $k \neq \tau_s$  // another task $\mathcal{T}^k_{mop}$ will be selected ;
18             $\mathcal{Q} \leftarrow \{x_i | x_i \in \mathcal{P}_k\}$  // the subpopulation $\mathcal{P}_k$ is taken as the object to be updated. ;
19         Set skill factor of individual $\hat{x}$ to be $k$ ;
20         Evaluate individual $\hat{x}$ for $\mathcal{T}^k_{mop}$ ;
21         Update ideal point $z^*_k$ ;
22         $time$=0 ;
23         for each $x$ in $\mathcal{Q}$ do
24             $id_x \leftarrow$ the subproblem index of $x$ in $\mathcal{T}^k_{mop}$ ;
25             if $g^{tch}(\hat{x}|\mathbf{\Lambda}^{id_x}_k, z^*_k) < g^{tch}(x|\mathbf{\Lambda}^{id_x}_k, z^*_k)$ then
26                 $x = \hat{x}$ ;
27                 $time$++ ;
28             if $time \geq n_r$ then
29                 break ;
30     $g = g + 1$;
31     if $mod(g,30)=0$ then
32         for $i$ = $1{:}N$ do
33             $\Delta^i \leftarrow$ calculate relative decrease value according to (5) ;
34             $\varphi_i \leftarrow$ update Scalar Fitness according to (6) ;
35         $\mathcal{P}_o = \mathcal{P}$ ;
36 return $\mathcal{P}$;
```

represents a subpopulation and each individual's skill factor is initialized to k. It should be noted that each individual p^k_i is randomly sampled from the unified space $\mathbf{Y}$. $\mathbf{z}^*_k$ is the ideal point for $\mathcal{T}^k_{mop}$. The initialization of these elements is shown in Algorithm 3 . $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_K\}$ is a united population.

3.1.2. Reproduction

In each generation, $N/5$ subproblems are selected based on scalar fitness via the 10-tournament selection strategy [31] for optimization, as shown in line 7 of Algorithm 2. Furthermore, three different individuals are randomly selected as parents, and the neighborhood selection probability δ determines whether the parents are selected from the neighborhood of individual s or the whole subpopulation $\mathcal{P}_k$, as shown in lines 11–14, where *rand* represents a random number generated between 0 and 1. Then, a new individual $\hat{x}$ is generated by the DE-rand/1/bin operator [35] and polynomial mutation operator [13].

Algorithm 3: INIT.

```
Input: task $k$ denoted by $\mathcal{T}_{mop}^k$; population size $N_k$; neighborhood size $T$;
Output: weight vectors $\mathbf{\Lambda}_k$; neighborhood structure $B_k$; population $\mathcal{P}_k$; ideal point $\mathbf{z}_k^*$;
1 $\mathbf{\Lambda}_k = \{\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_{N_k}\} \leftarrow$ a set of weight vectors are generated by Das & Dennis method [11] ;
2 $B_k = \{B_1^k, B_2^k, \dots, B_{N_k}^k\} \leftarrow$ where $B_i^k = \{i_1, i_2, \dots, i_T\}$ is the neighborhood of the subproblem $i$ which calculated by the Euclidean distance between the weight vectors $\mathbf{\Lambda}_k$ ;
3 $\mathcal{P}_k = \{p_1^k, p_2^k, \dots, p_{N_k}^k\} \leftarrow$ randomly initialized $N_k$ individuals ;
4 set each individual's skill factor $\tau_i = k$, set each individual's scalar fitness $\varphi_i = 1.0$, and evaluate each individual for $\mathcal{T}_{mop}^k$ ;
5 $\mathbf{z}_k^* = \{z_1, z_2, \dots, z_m\} \leftarrow$ initialize the ideal point ;
6 return $(\mathbf{\Lambda}_k, B_k, \mathcal{P}_k, \mathbf{z}_k^*)$;
```

3.1.3. Update of population

Random mating probability *rmp* plays an important role in the communication between different tasks. If $rand < rmp$ condition is satisfied, then another task will be randomly selected, *i.e.*, knowledge will be transferred between different tasks, as shown in lines 16–18. Lines 23–29 describe the process of population updating. Replacement size n_r limits the number of subproblems to be updated, thus multiple individuals replaced by a single solution are prevented and the diversity of the population can be maintained.

3.1.4. Update of scalar fitness

In [21], the scalar fitness is determined by the factorial rank and the scalar factor. The factorial rank is calculated by non-dominated sorting and crowding distance. In this paper, a new scalar fitness update method is used. Inspired by Zhang et al. [44], the utility function is used to update the individual's scalar fitness, which is updated periodically, as shown in lines 31–35 of Algorithm 2. In line 33, the relative decrease Δ^i is calculated by formula (5), and formula (6) is used to update the scalar fitness φ_i.

$$\Delta^i = \frac{g^{tch}(x_i^{old}|\lambda_i, z^*) - g^{tch}(x_i^{new}|\lambda_i, z^*)}{g^{tch}(x_i^{old}|\lambda_i, z^*)} \tag{5}$$

$$\varphi_i = \begin{cases} 1, & \Delta^i > 0.001 \\ (0.95 + 0.05\frac{\Delta^i}{0.001})\varphi_i, & \text{otherwise} \end{cases} \tag{6}$$

3.2. Notes of MFEA/D-DRA

Different from the traditional decomposition based multiobjective evolutionary algorithm, MFEA/D-DRA needs to optimize multiple distinct multiobjective tasks and obtain the approximate Pareto front of each task. Therefore, each multiobjective task needs to be given a set of weight vectors based on which it can be decomposed into a series of single-objective subproblems, as shown in line 1 of Algorithm 3. In the process of population evolution, the existence of random matching probability makes it possible to exchange information between different tasks, as shown in lines 16–18 of Algorithm 2. It should be noted that MFEA/D-DRA updates individuals with different parental skill factors through the offspring generated by parents of the same skill factor to achieve information exchange between different tasks (a simple statement is that the offspring generated by the parents associated with task A are used to update task B). Scalar fitness can be interpreted as the degree of competition of the individual in the population. A larger scalar fitness value indicates that the individual is more competitive and more worthy of being invested (giving more computing resources), as shown in line 7 of Algorithm 2. The scalar fitness values of all individuals in the population are updated dynamically, as shown in lines 31–35 of Algorithm 2, to achieve a reasonable allocation of computing resources.

Fig. 1 clearly shows the process that an MO-MFO problem (each task is a bi-objective problem) is decomposed into a series of single-objective subproblems. In this figure, the neighborhood size of each subproblem is set as 5. It can be seen that, if the red line represents the current subproblem, the shaded subproblems are its neighborhood. In the process of evolution, the random matching probability (*rmp*) determines whether communication occurs between different tasks. Taking subproblem sp_4 as an example, if the condition $rand < rmp$ is satisfied, the newly generated individual is not used to update $\mathcal{T}_{mop}^1$. Instead, it is used to update the individuals whose skill factors are equal to 2, *i.e.*, $\mathcal{T}_{mop}^2$ (due to the limitation of replacement size n_r, up to n_r individuals will be updated actually). Similarly, the knowledge of sp_{10} in $\mathcal{T}_{mop}^2$ will be also transferred to $\mathcal{T}_{mop}^1$ with a certain probability.

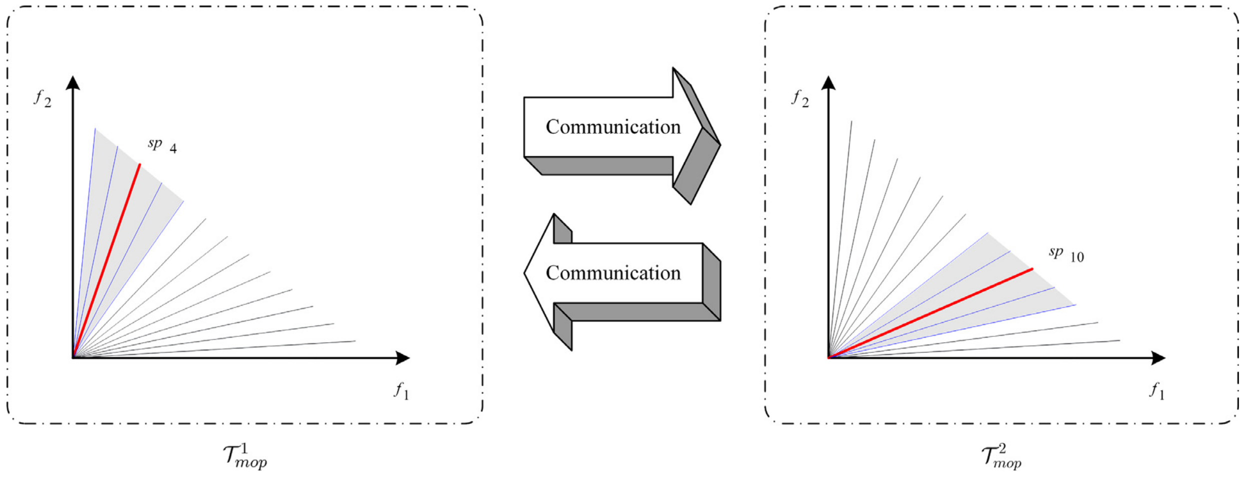


Fig. 1. The process of communication between two tasks.

3.3. Computational complexity

In each generation, Ns individuals (Algorithm 2, line 7, $Ns=|\mathcal{S}|$) are selected based on scalar fitness for the offspring generation. For a newly generated individual, the size of individual set $\mathcal{Q}$ that can be used for updating is related to the neighborhood selection probability δ, the number of tasks K, and the random matching probability *rmp* (Algorithm 2, lines 11–18). The worst case is that the individual set $\mathcal{Q} = \mathcal{P}_k$, ($|\mathcal{Q}|=N/K$, where N is population size), and all of the individuals within $\mathcal{Q}$ are traversed (lines 23–29 of the Algorithm 2). According to the above analysis, the computational complexity in each generation is $O(Ns \times N/K)$. In general, Ns is linearly dependent on N, here, $Ns=N/5$, so the final computational complexity of the proposed algorithm is $O(N^2/K)$.

3.4. Comparisons with MO-MFEA [21]

The MO-MFEA is an extension of the NSGA-II for solving MO-MFO problem, and in the special case, if only one task to be optimized, it takes the exact form of NSGA-II. The MO-MFEA algorithm evolves multiple tasks, each of which is an MOP, by a single population. In each generation, the offspring is generated via the Assortative Mating strategy, and the skill factor of each offspring is determined according to the Vertical Cultural Transmission. The factorial rank of each individual is determined by non-dominated sorting and crowding distance. Finally, the population for next generation is constructed by selecting individuals according to their scalar fitness. Therefore, MO-MFEA is a generational evolution model [15]. Different from MO-MFEA, the proposed MFEA/D-DRA converts each MOP task into a series of single-objective optimization subproblems, and a single population is used to evolve the union of single-objective subproblems of all MOP tasks. In each generation, subproblems with more rewards are selected for evolution. Each newly generated individual evolves its own subproblem in real time or updates other subproblem with a certain probability, which is a steady-state evolution model [15].

4. Experimental studies

The most state-of-the-art algorithms to be compared with our proposed algorithm are MO-MFEA [21], MOEA/D-DE [24], and MOEA/D-DRA [44]. The reason for choosing these algorithms is that MO-MFEA is a paradigm for solving MO-MFO problems, and that our proposed algorithm is based on the idea of MOEA/D-DE and dynamic resource allocation (so MOEA/D-DE and MOEA/D-DRA are also selected). All algorithms are run on a personal computer with Intel(R) Core(TM) i7-7700K CPU @ 4.20GHz and 16GB RAM. The codes of these algorithms are implemented in Java on the jMetal [33] platform.

4.1. Benchmark problems

Nine MO-MFO benchmark test problems are selected for experimental studies [41], each of which is a two-task problem. According to a transfer function $q(x)$, the relationship between two tasks of these benchmark problems can be divided into three categories, *i.e.*, Complete Intersection, Partial Intersection, and No Intersection. Moreover, the Spearmanâs rank correlation coefficient of the fitness landscape of $q(x)$ obtained by random sampling in the unified space is regarded as the similarity between two tasks, and these benchmark problems can also be divided into the following three categories, *i.e.*, High Similarity, Medium Similarity, Low Similarity. Therefore, the following nine benchmark problems can be obtained by combining the degree of intersection and similarity, *i.e.*, Complete Intersection with High/Medium/Low Similarity (CIHS, CIMS,

Table 1
Test problems.

Problem	Similarity	Variable Bounds	n	m	PF Properties
CIHS1	0.97	$[0,1]\times[-100,100]^{n-1}$	50	2	concave, unimodal, separable
CIHS2		$[0,1]\times[-100,100]^{n-1}$	50	2	concave, unimodal, separable
CIMS1	0.52	$[0,1]\times[-5,5]^{n-1}$	10	2	concave, multimodal, nonseparable
CIMS2		$[0,1]\times[-5,5]^{n-1}$	10	2	concave, unimodal, nonseparable
CILS1	0.07	$[0,1]\times[-2,2]^{n-1}$	50	2	concave, multimodal, separable
CILS2		$[0,1]\times[-1,1]^{n-1}$	50	2	convex, multimodal, nonseparable
PIHS1	0.99	$[0,1]\times[-5,5]^{n-1}$	50	2	convex, unimodal, separable
PIHS2		$[0,1]\times[-5,5]^{n-1}$	50	2	convex, multimodal, separable
PIMS1	0.55	$[0,1]^{n}$	50	2	concave, unimodal, nonseparable
PIMS2		$[0,1]^{n}$	50	2	concave, multimodal, nonseparable
PILS1	0.002	$[0,1]\times[-50,50]^{n-1}$	50	2	concave, multimodal, nonseparable
PILS2		$[0,1]\times[-100,100]^{n-1}$	50	2	concave, multimodal, nonseparable
NIHS1	0.94	$[0,1]\times[-80,80]^{n-1}$	50	2	concave, multimodal, nonseparable
NIHS2		$[0,1]\times[-80,80]^{n-1}$	50	2	convex, unimodal, separable
NIMS1	0.51	$[0,1]^2\times[-20,20]^{n-2}$	20	3	concave, multimodal, nonseparable
NIMS2		$[0,1]^2\times[-20,20]^{n-2}$	20	2	concave, unimodal, nonseparable
NILS1	0.001	$[0,1]^2\times[-50,50]^{n-2}$	25	3	concave, multimodal, nonseparable
NILS2		$[0,1]^2\times[-100,100]^{n-2}$	50	2	concave, multimodal, nonseparable

Table 2
Parameter setting.

	T	nr	δ	F	Cr
MOEA/D-DE [24]	20	2	0.9	0.5	1.0
MOEA/D-DRA [44]	20	2	0.9	0.5	1.0
MFEA/D-DRA	10	2	0.8	0.5	0.9

CILS), Partial Intersection with High/Medium/Low Similarity (PIHS, PIMS, PILS) and No Intersection with High/Medium/Low Similarity (NIHS, NIMS NILS). In addition, the PF properties of these MO-MFO tasks are also diverse, such as concave, convex, unimodal, multimodal, separable and non-separable. The detailed information of the test benchmark problems used in this experimental studies is shown in Table 1.

4.2. Parameter setting

For each MO-MFO problem, the common parameters of all algorithms are set as follows :

- *Maximum number of Approximate PS* : For each task, the maximum number of non-dominated solutions obtained by the algorithm is set to 105[1].
- *Stopping Criterion* : Maximum number of function evaluations ($FEs = 210{,}000$).
- *Independent Run Times* : With different random seeds, all tasks run 30 times independently.

It should be noted here that for a multifactorial problem, each function evaluation refers to the evaluation of a single task.

The parameters used in the generation of offspring in the MO-MFEA are set as follows. The random mating probability (rmp) of knowledge exchange between different tasks is set to 0.3, the crossover probability (p_c) for the simulated binary crossover (SBX) operator [1] is set to 0.9, the mutation probability (p_m) for the polynomial mutation operator [13] is set to $1/D$, where D is the dimension of the unified space, the distribution index for crossover (η_c) is set to 20, and the distribution index for mutation (η_m) is set to 20. For MOEA/D, MOEA/D-DRA and MFEA/D-DRA, the parameters of polynomial mutation operator are the same as MO-MFEA. In addition, differential evolution (DE) [35] operator is used in MOEA/D, MOEA/D-DRA and MFEA/D-DRA. Here, the setting of constant factor (F), crossover constant (C_r), neighborhood size (T), replacement size (n_r) and neighborhood selection probability (δ) is shown in Table 2. In MOEA/D-DRA, the update period of utility function is set as 30, and in MFEA/D-DRA, the update period of the scalar fitness is also set as 30. The random matching probability of MFEA/D-DRA is set as 0.1, which is obtained by the experimental analysis in Section 4.5.3.

[1] In the decomposition-based algorithm, in general, the population size is consistent with the number of weight vectors. When the number of objectives is greater than 2, the number of weight vectors generated by Das & Dennis method is not flexible [11]. Here, 105 wt vectors can satisfy the Das & Dennis method for both bi- and tri- objective problems, and for a flexible population size setting method, please refer to Zhang et al. [44].

Table 3
The HV metric Mean and Standard Deviation performance analysis of MFEA/D-DRA with 3 state-of-the-art algorithms.

	MO-MFEA	MOEA/D-DE	MOEA/D-DRA	MFEA/D-DRA
CIHS1	$1.933e-01_{5.0e-03}{}^{+}$	$1.661e-01_{1.2e-02}{}^{+}$	$1.719e-01_{9.1e-03}{}^{+}$	$\mathbf{2.092e-01_{5.9e-04}}$
CIHS2	$2.270e-01_{1.3e-02}{}^{+}$	$2.607e-01_{8.0e-03}{}^{+}$	$2.558e-01_{1.3e-02}{}^{+}$	$\mathbf{3.091e-01_{4.5e-03}}$
CIMS1	$1.416e-01_{1.5e-01}{}^{+}$	$2.727e-02_{8.6e-02}{}^{+}$	$1.051e-02_{5.8e-02}{}^{+}$	$\mathbf{3.287e-01_{1.5e-05}}$
CIMS2	$7.407e-02_{7.8e-02}{}^{+}$	$2.094e-01_{6.7e-04}{}^{+}$	$2.091e-01_{1.1e-03}{}^{+}$	$\mathbf{2.101e-01_{1.3e-04}}$
CILS1	$2.001e-01_{1.8e-03}{}^{+}$	$0.000e+00_{0.0e+00}{}^{+}$	$0.000e+00_{0.0e+00}{}^{+}$	$\mathbf{2.081e-01_{5.4e-04}}$
CILS2	$6.589e-01_{2.5e-04}{}^{+}$	$6.544e-01_{6.3e-04}{}^{+}$	$6.557e-01_{7.7e-04}{}^{+}$	$\mathbf{6.611e-01_{7.0e-05}}$
PIHS1	$\mathbf{6.444e-01_{4.3e-03}}{}^{-}$	$6.285e-01_{8.3e-03}{}^{=}$	$6.385e-01_{4.9e-03}{}^{-}$	$6.288e-01_{7.3e-03}$
PIHS2	$2.803e-02_{4.8e-02}{}^{+}$	$0.000e+00_{0.0e+00}{}^{+}$	$0.000e+00_{0.0e+00}{}^{+}$	$\mathbf{1.434e-01_{1.4e-01}}$
PIMS1	$1.215e-01_{2.6e-02}{}^{+}$	$9.209e-02_{2.9e-02}{}^{+}$	$1.275e-01_{2.8e-02}{}^{=}$	$\mathbf{1.381e-01_{2.7e-02}}$
PIMS2	$0.000e+00_{0.0e+00}{}^{=}$	$0.000e+00_{0.0e+00}{}^{=}$	$0.000e+00_{0.0e+00}{}^{=}$	$0.000e+00_{0.0e+00}$
PILS1	$2.033e-01_{4.6e-03}{}^{-}$	$\mathbf{2.048e-01_{4.5e-03}}{}^{-}$	$2.032e-01_{9.0e-03}{}^{-}$	$1.946e-01_{6.9e-03}$
PILS2	$3.946e-02_{1.7e-02}{}^{+}$	$1.953e-03_{6.1e-03}{}^{+}$	$1.015e-02_{2.2e-02}{}^{+}$	$\mathbf{1.610e-01_{1.5e-02}}$
NIHS1	$0.000e+00_{0.0e+00}{}^{=}$	$0.000e+00_{0.0e+00}{}^{=}$	$0.000e+00_{0.0e+00}{}^{=}$	$0.000e+00_{0.0e+00}$
NIHS2	$6.433e-01_{4.3e-03}{}^{+}$	$6.362e-01_{5.5e-03}{}^{+}$	$6.409e-01_{3.1e-03}{}^{+}$	$\mathbf{6.479e-01_{4.4e-03}}$
NIMS1	$0.000e+00_{0.0e+00}{}^{=}$	$9.726e-03_{5.3e-02}{}^{=}$	$\mathbf{1.107e-02_{5.0e-02}}{}^{=}$	$0.000e+00_{0.0e+00}$
NIMS2	$1.107e-01_{1.2e-01}{}^{+}$	$2.243e-01_{1.2e-01}{}^{+}$	$2.042e-01_{1.3e-01}{}^{+}$	$\mathbf{3.208e-01_{5.7e-03}}$
NILS1	$3.670e-01_{1.1e-02}{}^{+}$	$3.727e-01_{1.9e-02}{}^{+}$	$3.749e-01_{1.9e-02}{}^{+}$	$\mathbf{3.961e-01_{1.7e-02}}$
NILS2	$0.000e+00_{0.0e+00}{}^{=}$	$9.300e-03_{4.1e-02}{}^{=}$	$\mathbf{2.254e-02_{6.9e-02}}{}^{=}$	$1.000e-02_{5.5e-02}$
+/=/-	12/4/2	12/5/1	11/5/2	

4.3. Performance metric

The inverted generational distance (IGD) [34] and hypervolume (HV) [16] are used to measure the performance of the proposed algorithm. The smaller IGD metric means the approximate Pareto front obtained by the algorithm is closer to the true Pareto front and can cover the true Pareto front better at the same time, while the HV metric is vice versa.

Below, let PF^* represent the approximate Pareto front obtained by the algorithm, PF refer to the true Pareto front, p_i^* and p_j be the elements in PF^* and PF, respectively, then the calculation formula of the IGD metric can be written as Eq. (7):

$$\begin{aligned} &IGD(PF, PF^*) = \frac{1}{|PF|}\sqrt{\sum_{i=1}^{|PF|} d(p_i, PF^*)^2} \\ &d(p_i, PF^*) = \min_{j=1}^{|PF^*|} ||p_i - p_j^*|| \end{aligned} \tag{7}$$

where $d(p_i, PF^*)$ refers to the minimum Euclidean distance from point p_i to PF^*.

The mathematical expression of HV metric is as follows :

$$HV(S) = VOL\left(\bigcup_{\mathbf{x}\in S}[f_1(\mathbf{x}), z_1^{nad}] \times \ldots \times [f_m(\mathbf{x}), z_m^{nad}]\right) \tag{8}$$

where S is the solution set obtained by the algorithm.

In this experiment, the true Pareto front for all test problems are downloaded from the Internet[2], and when calculating HV metric, the true Pareto front and approximate Pareto front obtained by the algorithm are normalized.

The non-parameter Wilcoxon test with a significance level of 0.05 is used to analyze whether there are significant differences between different algorithms after multiple runs. The symbols '+', '=' and '-' are used to mark whether MFEA/D-DRA is significantly better than, similar to or worse than the rival algorithms, respectively.

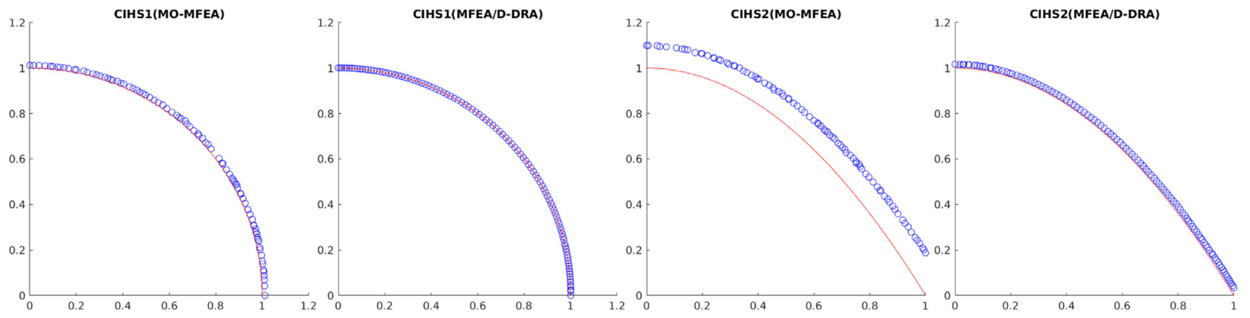
4.4. Performance analysis

The statistical analysis results of the Mean and Standard Deviation of HV metric and IGD metric are presented in Tables 3 and 4, respectively, which are obtained by 30 independent runs for each test problem by MO-MFEA, MOEA/D-DE, MOEA/D-DRA and MFEA/D-DRA. The best metric value is highlighted in gray shadow. From Table 3, in terms of HV metric, the Wilcoxon test shows that MFEA/D-DRA outperforms MO-MFEA for 11 out of 18 tasks, outperforms MOEA/D-DE for 12 tasks, and outperforms MOEA/D-DRA for 11 tasks. Furthermore, MFEA/D-DRA obtained the best HV mean metrics for 15 of the 18 tasks. MO-MFEA performs better than the compared algorithms on PIHS1; MOEA/D-DE performs better than the compared algorithms on PILS1; and MOEA/D-DRA performs better than the compared algorithms on NIMS1 and NILS2. From Table 4, in terms of IGD metric, the Wilcoxon test shows that MFEA/D-DRA outperforms MO-MFEA for 13 out of 18 tasks, outperforms MOEA/D-DE for 15 tasks, and outperforms MOEA/D-DRA for 14 tasks. Furthermore, MFEA/D-DRA obtained the best IGD mean metrics for 14 of the 18 tasks. MO-MFEA performs better than the compared algorithms on PIHS1, PIHS1 and PIMS2,

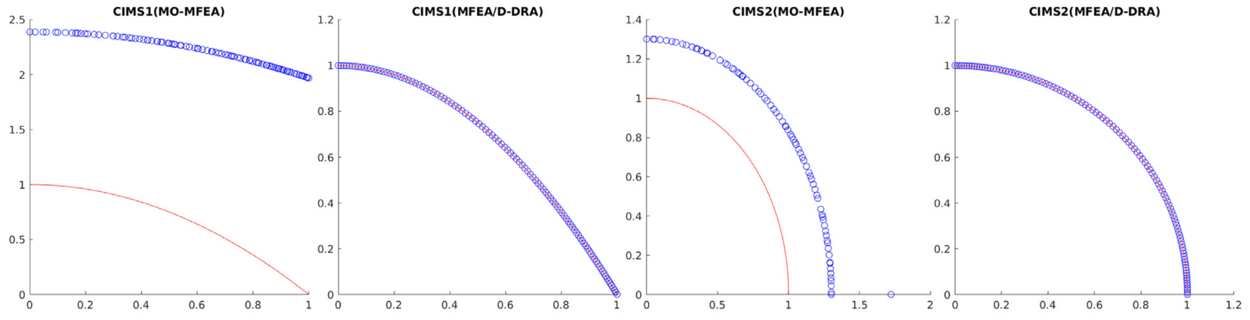
[2] http://www.ntu.edu.sg/home/asysong/.

Table 4
The IGD metric Mean and Standard Deviation performance analysis of MFEA/D-DRA with 3 state-of-the-art algorithms.

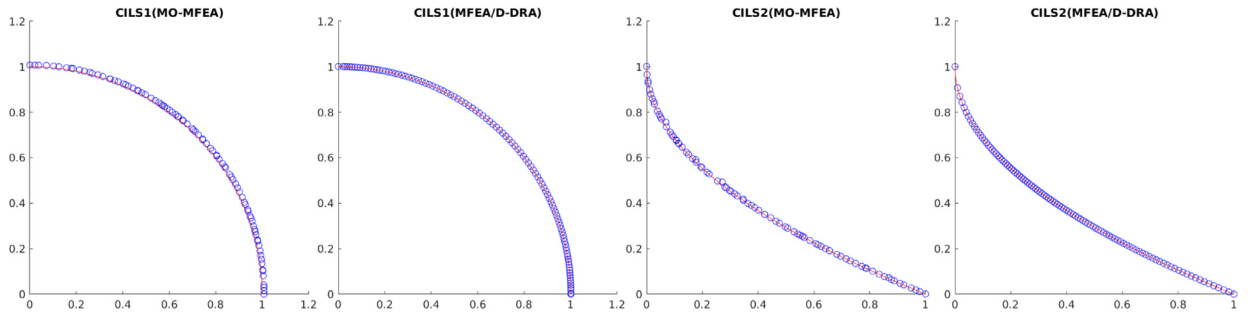
	MO-MFEA	MOEA/D-DE	MOEA/D-DRA	MFEA/D-DRA
CIHS1	$4.332e-04_{1.2e-04}{}^{+}$	$1.156e-03_{3.5e-04}{}^{+}$	$9.859e-04_{2.6e-04}{}^{+}$	$\mathbf{1.676e-04_{3.0e-06}}$
CIHS2	$2.772e-03_{4.3e-04}{}^{+}$	$1.749e-03_{2.4e-04}{}^{+}$	$1.916e-03_{3.9e-04}{}^{+}$	$\mathbf{4.773e-04_{1.1e-04}}$
CIMS1	$5.901e-02_{7.5e-02}{}^{+}$	$1.004e-01_{5.2e-02}{}^{+}$	$1.142e-01_{4.1e-02}{}^{+}$	$\mathbf{1.339e-04_{2.9e-08}}$
CIMS2	$1.085e-02_{1.2e-02}{}^{+}$	$1.667e-04_{2.7e-06}{}^{+}$	$1.688e-04_{7.1e-06}{}^{+}$	$\mathbf{1.649e-04_{2.0e-07}}$
CILS1	$2.951e-04_{3.2e-05}{}^{+}$	$8.947e-01_{5.0e-01}{}^{+}$	$1.038e+00_{6.2e-01}{}^{+}$	$\mathbf{1.720e-04_{3.5e-06}}$
CILS2	$1.984e-04_{6.8e-06}{}^{+}$	$2.313e-04_{1.2e-05}{}^{+}$	$2.091e-04_{1.2e-05}{}^{+}$	$\mathbf{1.560e-04_{3.5e-07}}$
PIHS1	$\mathbf{4.222e-04_{8.8e-05}}{}^{-}$	$7.939e-04_{1.9e-04}{}^{=}$	$5.644e-04_{1.1e-04}{}^{-}$	$7.818e-04_{1.7e-04}$
PIHS2	$\mathbf{2.616e-02_{7.4e-03}}{}^{=}$	$5.350e-01_{1.1e-01}{}^{+}$	$5.478e-01_{1.3e-01}{}^{+}$	$3.384e-02_{4.7e-02}$
PIMS1	$2.688e-03_{1.1e-03}{}^{+}$	$3.944e-03_{1.3e-03}{}^{+}$	$2.487e-03_{1.1e-03}{}^{=}$	$\mathbf{2.081e-03_{9.5e-04}}$
PIMS2	$\mathbf{1.295e+01_{4.2e+00}}{}^{-}$	$2.270e+01_{6.7e+00}{}^{+}$	$2.333e+01_{7.2e+00}{}^{+}$	$1.929e+01_{4.5e+00}$
PILS1	$2.568e-04_{7.9e-05}{}^{-}$	$\mathbf{2.172e-04_{8.4e-05}}{}^{-}$	$2.566e-04_{2.0e-04}{}^{-}$	$4.092e-04_{1.7e-04}$
PILS2	$6.900e-03_{1.3e-03}{}^{+}$	$5.020e-01_{2.4e-01}{}^{+}$	$4.868e-01_{2.7e-01}{}^{+}$	$\mathbf{1.310e-03_{4.6e-04}}$
NIHS1	$1.537e+00_{2.5e-02}{}^{+}$	$5.956e+00_{3.0e+00}{}^{+}$	$5.431e+00_{1.5e+00}{}^{+}$	$\mathbf{1.476e+00_{1.2e-02}}$
NIHS2	$4.447e-04_{8.9e-05}{}^{+}$	$6.164e-04_{1.2e-04}{}^{+}$	$5.121e-04_{6.9e-05}{}^{+}$	$\mathbf{3.735e-04_{9.4e-05}}$
NIMS1	$3.269e-01_{2.9e-01}{}^{+}$	$1.870e-01_{1.8e-01}{}^{+}$	$2.062e-01_{2.4e-01}{}^{+}$	$\mathbf{9.719e-02_{1.8e-02}}$
NIMS2	$3.673e-02_{6.9e-02}{}^{+}$	$7.667e-03_{1.5e-02}{}^{+}$	$9.856e-03_{1.7e-02}{}^{+}$	$\mathbf{2.297e-04_{1.1e-04}}$
NILS1	$8.187e-04_{5.5e-05}{}^{+}$	$6.635e-04_{6.6e-05}{}^{+}$	$6.508e-04_{5.6e-05}{}^{+}$	$\mathbf{5.892e-04_{3.7e-05}}$
NILS2	$6.428e-01_{4.0e-04}{}^{=}$	$5.606e-01_{2.1e-01}{}^{=}$	$\mathbf{5.421e-01_{2.3e-01}}{}^{=}$	$6.206e-01_{1.2e-01}$
+/=/-	13/2/3	15/2/1	14/2/2	



(a) CIHS

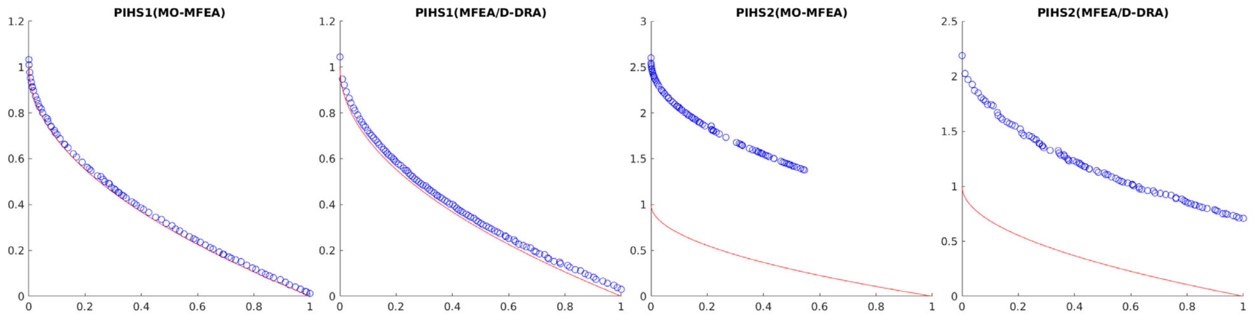


(b) CIMS

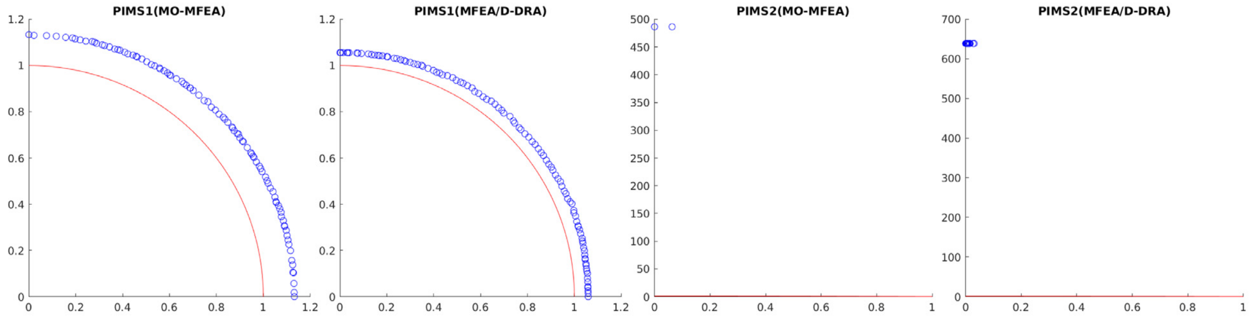


(c) CILS

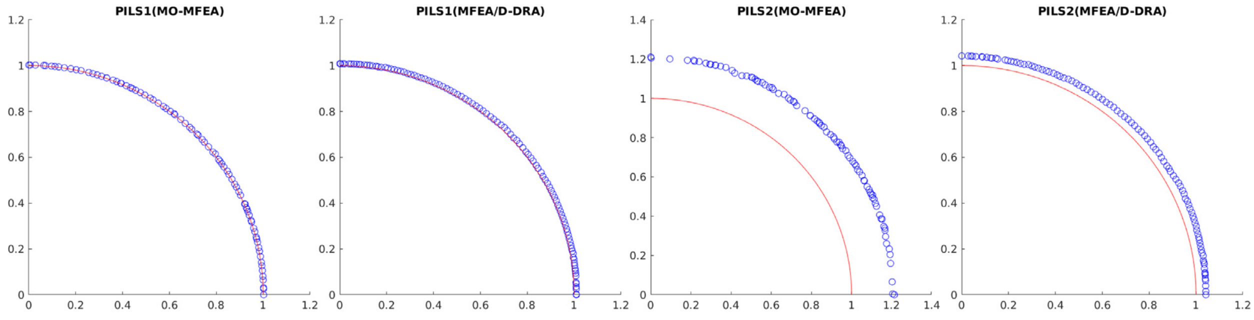
Fig. 2. Final solution set obtained by two MFEAs in 30 independently runs with the Median IGD metric value of CIHS, CIMS, and CILS.



(a) PIHS



(b) PIMS



(c) PILS

Fig. 3. Final solution set obtained by two MFEAs in 30 independently runs with the Median IGD metric value of PIHS, PIMS, and PILS.

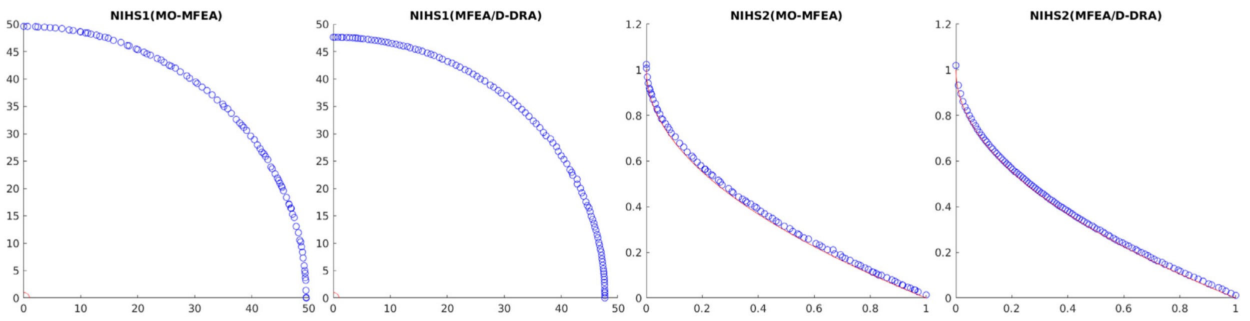
MOEA/D-DE performs better than the compared algorithms on PILS1, and MOEA/D-DRA performs better than the compared algorithms on NILS2.

In order to visually observe the distribution of the final solution set of our proposed algorithm and the difference between MO-MFEA (since both algorithms are based on multifactorial evolutionary strategy, only the comparison between these the two algorithms is performed here.), Fig. 2–4 show the results of Pareto front with the median IGD metric among 30 independent runs. It can be clearly seen that the MFEA/D-DRA algorithm can obtain a better Pareto front approximation for a variety of PF shape problems. In addition, Fig. 5 shows the process results with the median IGD metric among 30 independent runs. In general, for these 9 different types of MO-MFO benchmark problems, the MFEA/D-DRA algorithm is competitive with or even superior to MO-MFEA, both in terms of convergence speed and final evaluation metrics.

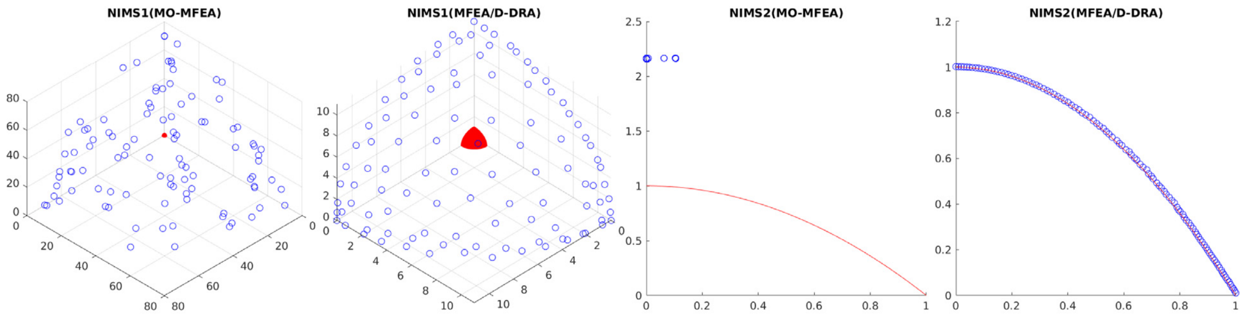
4.5. *More discussions*

4.5.1. *Efficiency analysis of dynamic resource allocation strategy*

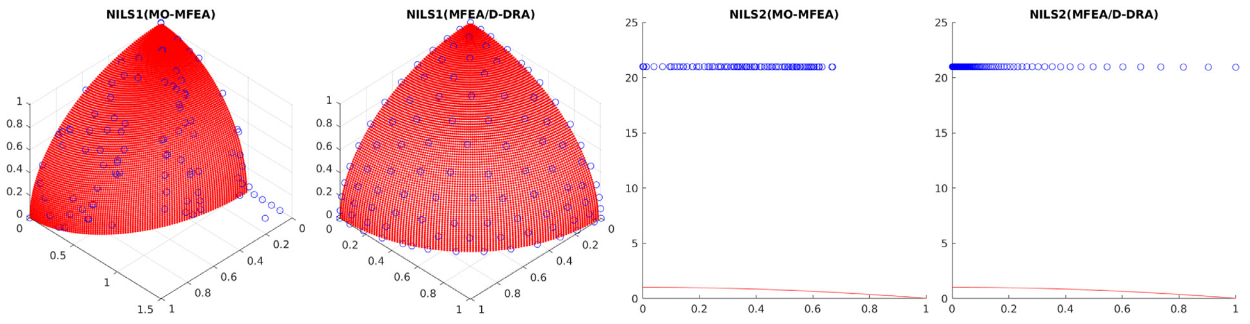
The above statistical analysis of experimental results illustrates the superiority of the proposed MFEA/D-DRA algorithm on a variety of benchmark MO-MFO problems. In this section, we further carried out experiment to check the performance of the algorithm without the dynamic resource allocation strategy. In the experiment, we change the pseudo-code line 7 in Algorithm 2 to $\mathcal{S} \leftarrow \mathcal{P}$, and delete lines 31 to 35. In this way, the whole algorithm becomes a multifactorial evolutionary algorithm based on decomposition strategy, denoted as MFEA/D. Based on the above test problems, parameter settings



(a) Fig. NIHS



(b) NIMS



(c) NILS

Fig. 4. Final solution set obtained by two MFEAs in 30 independently runs with the Median IGD metric value of NIHS, NIMS, and NILS.

and performance evaluation metrics, the statistical analysis of experimental results of the Mean and Standard Deviation of HV&IGD metrics are shown in Table 5. As can be seen from Table 5, regardless of the HV metric or the IGD metric, Wilcoxon test shows that MFEA/D outperforms MFEA/D-DRA for only 3 out of 18 tasks. Therefore, it can be concluded that the dynamic resource allocation strategy plays an important role in the proposed algorithm.

4.5.2. Computational resource allocation ratio analysis

In MFEA/D-DRA, limited computing resources are not only dynamically assigned to different tasks, but also to different single-objective optimization subproblems. Due to limited space, CIHS, CIMS and CILS are taken as examples, and the distribution of computing resources for each problem is shown in Fig. 6. The two sides of Fig. 6 show the visualization results of the number of evaluations allocated to each subproblem of different tasks after 30 independent runs of MFEA/D-DRA, and the middle figure shows the statistical results of average resource distribution occupancy between two tasks after 30 independent runs. As can be seen from Fig. 6, for CIHS1 *vs.* CIHS2, CIHS2 is allocated more computing resources, accounting for 64.04%. Specifically, in terms of subproblems, the subproblems in CIHS1 that are close to the coordinate axis are allocated more computing resources, combined with Fig. 1. In CIHS2, the reward difference for each subproblem is also very clear, and it seems that the subproblems with larger indexes (*i.e.*, close to coordinate axis 2 in Fig. 1) is more worth being invested. For CIMS1 *vs.* CIMS2, the difference in computing resources between the two tasks is not obvious. In CIMS1, subproblems far from the coordinate axis are allocated a relatively small amount of computing resources. But in CIMS2, the subproblems close the coordinate axis are allocated more computing resources and the fluctuations are obvious. For CILS1 *vs.* CILS2, CILS1

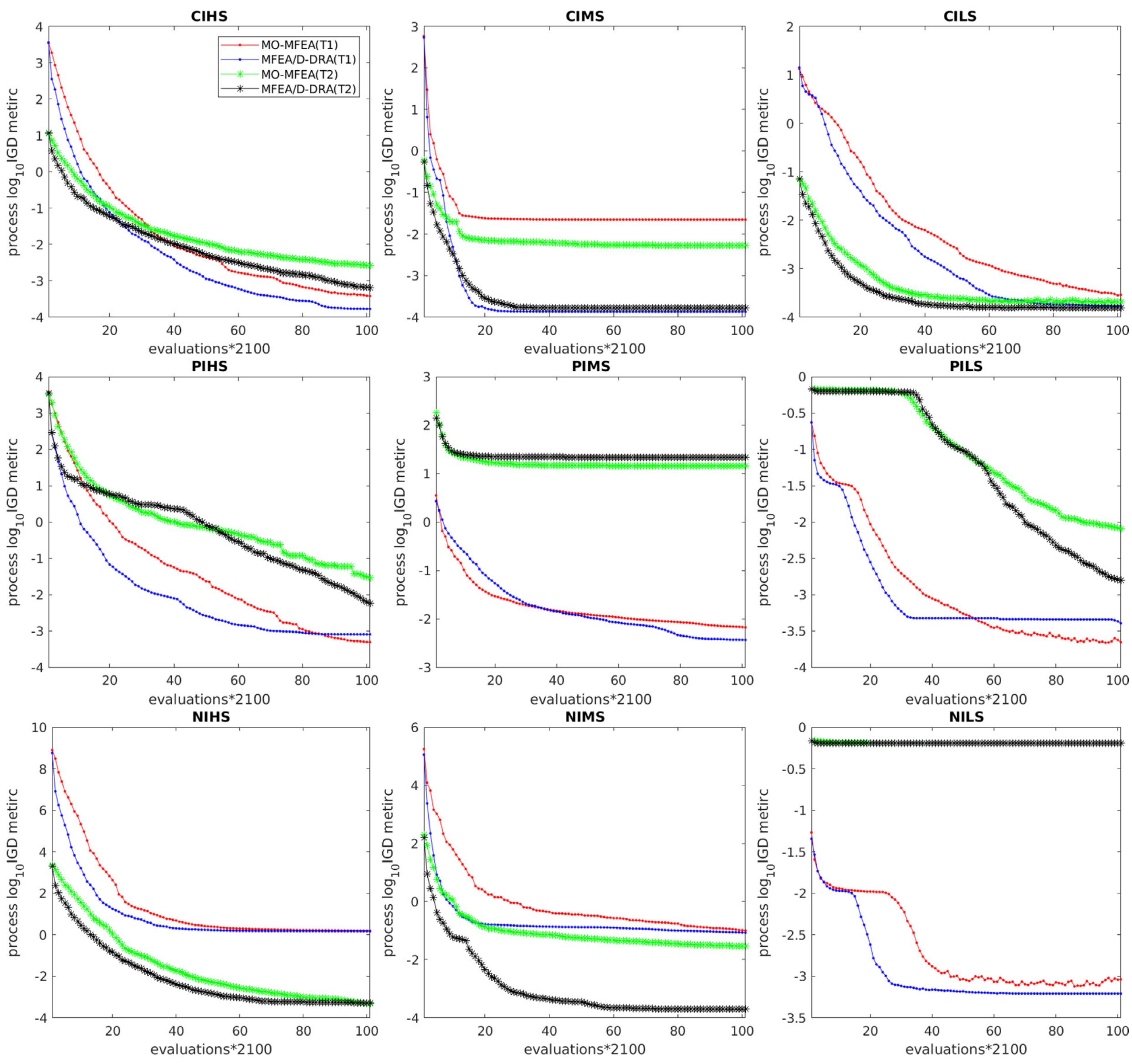


Fig. 5. Process results with the median IGD metric.

obtains the absolutely dominant computing resources, and the computing resources obtained by each subproblem fluctuated greatly. But in general, the allocation of computing resources for each subproblem of CILS1 is relatively uniform. In CILS2, the subproblems close to coordinate axis 1 (as shown in Fig. 1) are allocated a relatively small amount of computing resources.

Therefore, based on the above observations, it can be concluded that the DRA mechanism actually realizes the adaptive allocation of limited computing resources to both different tasks and different subproblems in each task according to their difficulty levels, which can focus the computing resources on more difficult tasks and subproblems and consequently can further improve the performance of MFEA/D.

4.5.3. Parameter sensitivity analysis

The effect of random matching probability (*rmp*) on the performance of our proposed algorithm is analyzed in this section. Intuitively, the smaller *rmp* means the less frequency of communication between different tasks. When $rmp = 0$, different tasks are separated and optimized independently. With the increase of *rmp*, the communication frequency between different tasks will increase correspondingly. In the extreme case, when $rmp = 1.0$, the offspring generated by parents with skill factor *k* will not update the individuals with skill factor *k*, *i.e.*, the optimization of each task is completely controlled by other tasks. Specifically, the effect analysis of different values of *rmp* in {0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0}

Table 5
The HV&IGD metrics Mean and Standard Deviation performance analysis of MFEA/D-DRA with MFEA/D.

Problem	HV		IGD	
	MFEA/D	MFEA/D-DRA	MFEA/D	MFEA/D-DRA
CIHS1	$2.085e-01_{7.5e-04}^{=}$	$\mathbf{2.089e-01_{1.1e-03}}$	$1.704e-04_{5.0e-06}^{=}$	$\mathbf{1.697e-04_{1.0e-05}}$
CIHS2	$2.971e-01_{5.3e-03}^{+}$	$\mathbf{3.075e-01_{4.6e-03}}$	$7.542e-04_{1.3e-04}^{+}$	$\mathbf{5.176e-04_{1.1e-04}}$
CIMS1	$3.287e-01_{1.2e-05}^{=}$	$\mathbf{3.287e-01_{2.4e-05}}$	$1.339e-04_{1.5e-08}^{=}$	$\mathbf{1.339e-04_{3.8e-08}}$
CIMS2	$2.100e-01_{8.0e-05}^{=}$	$\mathbf{2.100e-01_{2.1e-04}}$	$\mathbf{1.649e-04_{1.2e-07}}^{=}$	$1.649e-04_{4.2e-07}$
CILS1	$2.034e-01_{1.4e-03}^{+}$	$\mathbf{2.080e-01_{7.1e-04}}$	$2.274e-04_{2.1e-05}^{+}$	$\mathbf{1.729e-04_{5.1e-06}}$
CILS2	$6.606e-01_{1.1e-04}^{+}$	$\mathbf{6.611e-01_{8.0e-05}}$	$1.575e-04_{4.6e-07}^{+}$	$\mathbf{1.561e-04_{3.6e-07}}$
PIHS1	$\mathbf{6.419e-01_{5.2e-03}}^{-}$	$6.286e-01_{9.3e-03}$	$\mathbf{4.896e-04_{1.2e-04}}^{-}$	$7.848e-04_{2.2e-04}$
PIHS2	$0.000e+00_{0.0e+00}^{+}$	$\mathbf{7.039e-02_{9.1e-02}}$	$1.216e-01_{1.0e-01}^{+}$	$\mathbf{4.351e-02_{5.1e-02}}$
PIMS1	$1.302e-01_{2.1e-02}^{=}$	$\mathbf{1.355e-01_{2.2e-02}}$	$2.334e-03_{7.4e-04}^{=}$	$\mathbf{2.143e-03_{7.4e-04}}$
PIMS2	$0.000e+00_{0.0e+00}^{=}$	$0.000e+00_{0.0e+00}$	$\mathbf{1.720e+01_{4.2e+00}}^{=}$	$1.739e+01_{3.6e+00}$
PILS1	$\mathbf{2.030e-01_{6.4e-03}}^{-}$	$1.933e-01_{6.9e-03}$	$\mathbf{2.513e-04_{1.2e-04}}^{-}$	$4.392e-04_{1.6e-04}$
PILS2	$6.627e-03_{1.8e-02}^{+}$	$\mathbf{1.577e-01_{1.7e-02}}$	$2.832e-02_{1.9e-02}^{+}$	$\mathbf{1.414e-03_{5.4e-04}}$
NIHS1	$0.000e+00_{0.0e+00}^{=}$	$0.000e+00_{0.0e+00}$	$1.501e+00_{6.7e-03}^{+}$	$\mathbf{1.479e+00_{1.1e-02}}$
NIHS2	$\mathbf{6.530e-01_{2.5e-03}}^{-}$	$6.490e-01_{3.8e-03}$	$\mathbf{2.590e-04_{4.7e-05}}^{-}$	$3.480e-04_{8.0e-05}$
NIMS1	$0.000e+00_{0.0e+00}^{=}$	$0.000e+00_{0.0e+00}$	$1.487e-01_{1.1e-01}^{=}$	$\mathbf{1.381e-01_{1.6e-01}}$
NIMS2	$3.153e-01_{5.9e-02}^{=}$	$\mathbf{3.191e-01_{8.8e-03}}$	$9.132e-04_{4.2e-03}^{=}$	$\mathbf{2.673e-04_{2.1e-04}}$
NILS1	$3.905e-01_{1.9e-02}^{=}$	$\mathbf{3.999e-01_{2.1e-02}}$	$6.149e-04_{5.0e-05}^{=}$	$\mathbf{5.957e-04_{6.2e-05}}$
NILS2	$1.988e-02_{6.2e-02}^{=}$	$\mathbf{3.788e-02_{1.0e-01}}$	$5.799e-01_{1.9e-01}^{=}$	$\mathbf{5.592e-01_{2.2e-01}}$
+/=/-	5/10/3		6/9/3	

Table 6
Decision variables.

Decision variables
$x_1 - x_5$: temperature of 5 zones in the heating furnace
$x_6 - x_7$: temperature of 2 zones in the soaking furnace
$x_8 - x_9$: temperature of 2 zones in the slow cooling furnace
$x_{10} - x_{11}$: temperature of 2 zones in the overaging furnace 1#
$x_{12} - x_{13}$: temperature of 2 zones in the overaging furnace 2#
x_{14} : inlet tension of leveling machine
x_{15} : intermediate tension of leveling machine
x_{16} : outlet tension of leveling machine
x_{17} : rolling force of leveling machine
x_{18} : strip speed

on our proposed algorithm is carried out based on the above nine MO-MFO problems and the result is shown in Fig. 7. Based on different values of *rmp*, we use the average ranking obtained by the Friedman test to analyze the effect of *rmp* on the performance of MFEA/D-DRA. Obviously, we can see from Fig. 7 that the communication between different tasks can improve the performance of the algorithm. Regardless of the IGD metric or the HV metric, the average Friedman ranking is most competitive at $rmp = 0.1$. In addition, the performance of MFEA/D-DRA will deteriorate as *rmp* increases. Therefore, the communication frequency between different tasks should be appropriately determined in practical problems.

4.6. A practical case study

4.6.1. Problem description

In the continuous annealing production process, the setting values (decision variables $\mathbf{x}$) of the production parameters required for different specifications (environmental parameters Pa) of steel strip are different. For the same unit, facing different incoming materials, it can be abstracted into a multifactorial optimization problem. For steel enterprises, steel strip quality (maximization), production energy consumption (minimization) and production capacity (maximization) are the three main goals they pursue, and these goals are interrelated and conflicting with each other. For example, on the premise of ensuring product quality, increasing production capacity requires more energy consumption, and vice versa. Therefore, the optimization of each task is a multiobjective optimization problem. Herein, we consider two distinct specifications of optimization task for steel strip production. The general definition of each optimization task is shown in expression (9).

$$\begin{aligned} &\{\mathbf{x_1}, \mathbf{x_2}\} = \arg\min\{\mathbf{F}(\mathbf{x}|Pa_1), \quad \mathbf{F}(\mathbf{x}|Pa_2)\} \\ &s.t. \quad \mathbf{x}_i \in \mathbf{X}_i, \quad i \in \{1, 2\} \end{aligned} \tag{9}$$

where Pa_1 and Pa_2 represent the two specifications of the strip (also called the production environment parameters, mainly including strip hardness, strip thickness, strip width, Carbon content of strip, Sulfur content of strip, etc.), $\mathbf{x}$ is the decision variables and the meaning of each variable is shown in Table 6. For each multiobjective optimization task, it can be

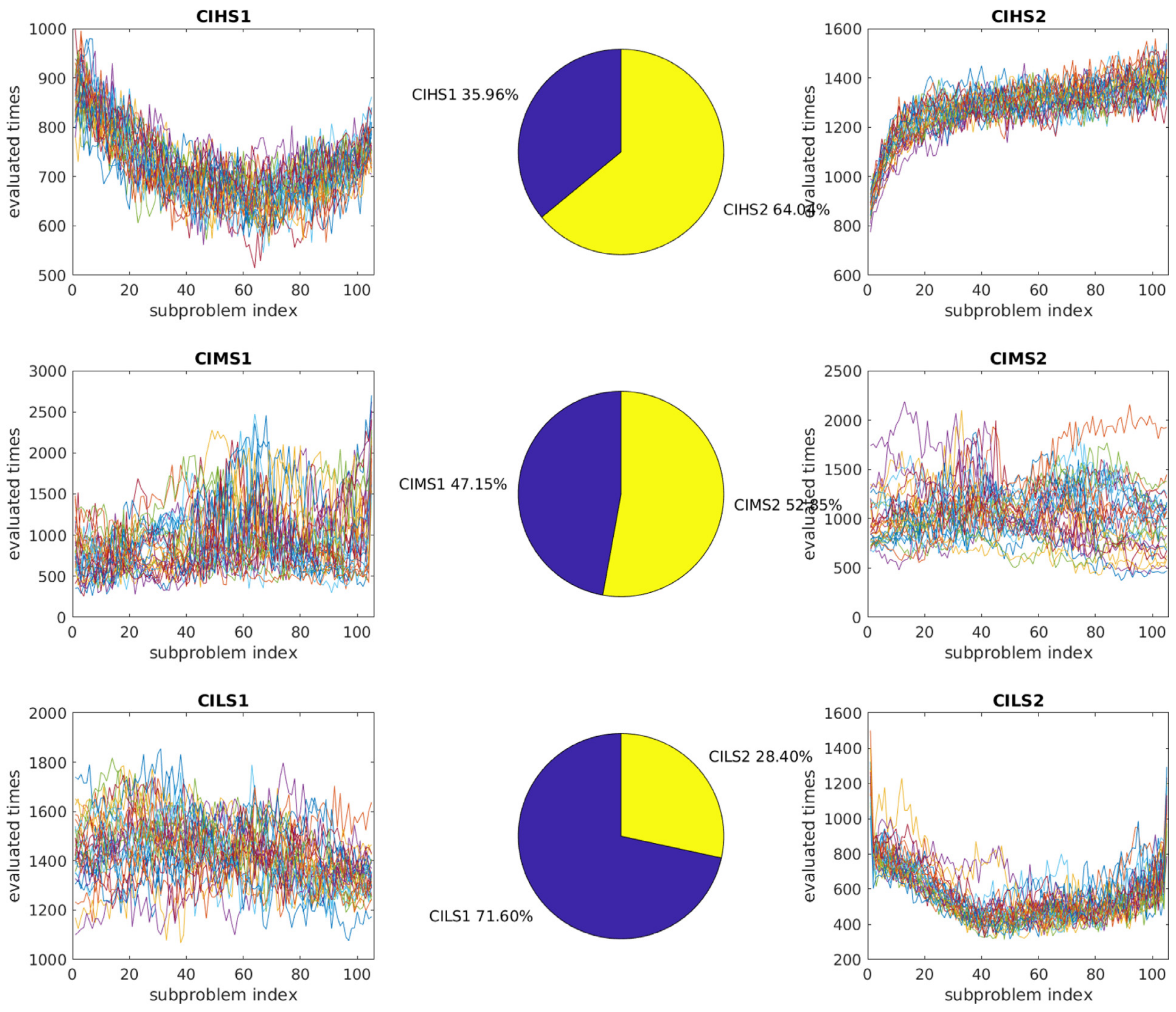


Fig. 6. Computational resource allocation ratio of CIHS CIMS and CILS.

expressed as formula (10).

$$\begin{aligned} &minimize\ \mathbf{F}(\mathbf{x}|Pa) = (|MODEL(\mathbf{x}, Pa) - Q|, \quad \textstyle\sum_{i=1}^{7} x_i/7, \quad -x_{18}) \\ &s.t. \quad x_i^l \le x_i \le x_i^u, \quad i \in \{1, 2, \ldots, 18\} \end{aligned} \tag{10}$$

where Q represents the target hardness of the strip, i.e., product quality. It should be noted that the quality of the strip cannot be directly measured in real time, and thus it is predicted by the environmental parameters Pa and decision variables $\mathbf{x}$ through an offline ensemble learning model $MODEL(Pa, \mathbf{x})$. Therefore, in expression (10), the first objective is to minimize the strip hardness deviation, the second objective is the average temperature of the furnace zones of the heating furnace and the soaking furnace (measurement of energy consumption), and the third objective is the production speed of the strip (the measure of the capacity, by adding a negative sign to become a minimum problem).

4.6.2. Computational results

In this experiment, MO-MFEA [21], MOEA/D-DE [24], and MOEA/D-DRA [44] are also used as comparison algorithms. For each task, the maximum number of non-dominated solutions obtained by the algorithm is set to 300. The maximum number of function evaluations $FEs = 600{,}00$, and with different random seeds, all algorithms run 30 times independently. Since the real PFs of the problem are not known, here an approximate reference PF is constructed from the results of these algorithms after 30 runs [5]. The Mean and Standard Deviation results of HV&IGD metrics obtained by the above algorithms are given in Table 7. Fig. 8 shows the population distribution of the median IGD metric in 30 independent runs of these algorithms (the black dots refer to the reference approximation PFs, the blue circles refer to the results obtained by the algorithm for

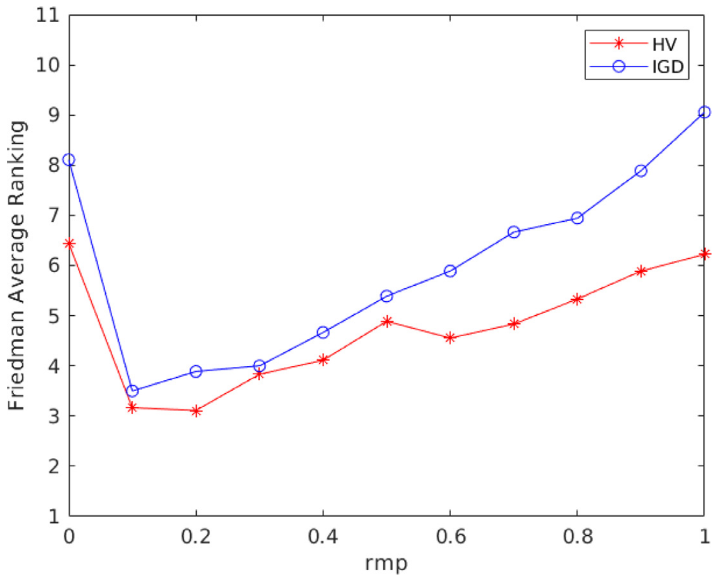


Fig. 7. Average ranking of MFEA/D-DRA with different *rmp* value.

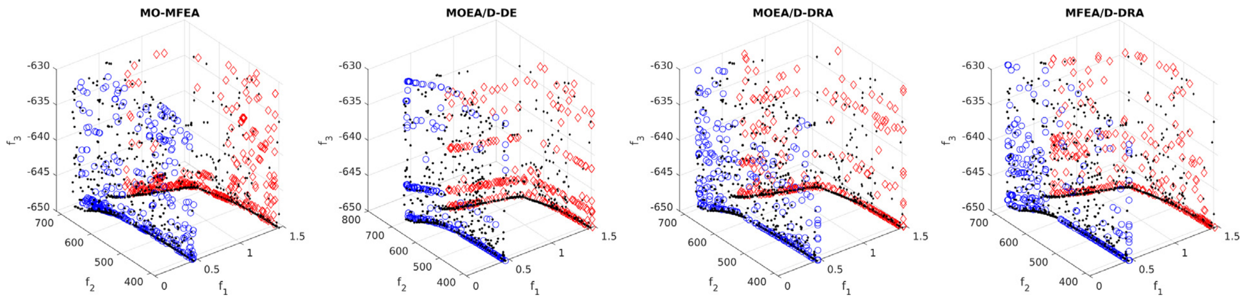


Fig. 8. The population distribution of the median IGD metric in 30 independent runs.

Table 7
The HV&IGD metrics Mean and Standard Deviation performance analysis of MFEA/D-DRA with 3 state-of-the-art algorithms for a practical problem.

Problem	Metric	MO-MFEA	MOEA/D-DE	MOEA/D-DRA	MFEA/D-DRA
Strip1	HV	$1.676e-01_{9.5e-03}{}^{+}$	$1.569e-01_{2.6e-02}{}^{+}$	$1.621e-01_{1.8e-02}{}^{+}$	$\mathbf{1.773e-01_{1.2e-02}}$
Strip2		$\mathbf{1.679e-01_{5.1e-03}}{}^{=}$	$1.656e-01_{6.7e-03}{}^{=}$	$1.654e-01_{1.1e-02}{}^{=}$	$1.662e-01_{7.7e-03}$
Strip1	IGD	$4.272e-03_{1.0e-03}{}^{+}$	$4.016e-03_{1.7e-03}{}^{+}$	$3.681e-03_{1.4e-03}{}^{+}$	$\mathbf{2.785e-03_{3.2e-04}}$
Strip2		$3.431e-03_{9.9e-04}{}^{+}$	$2.945e-03_{6.7e-04}{}^{+}$	$2.886e-03_{1.0e-03}{}^{=}$	$\mathbf{2.535e-03_{3.2e-04}}$

task 1 (Strip1), and the red diamonds refer to the results obtained by the algorithm for task 2 (Strip2)). From the statistical results of HV and IGD metrics, in general, the MFEA/D-DRA algorithm is superior to other algorithms. Although the MO-MFEA algorithm has a better result in terms of HV metric for task 2, there is no significant difference from MFEA/D-DRA. In addition, it can be found that multifactorial optimization is better than single task optimization based on the comparison results between MOEA/D and MOEA/D-DRA.

5. Conclusions

In this paper, a multifactorial evolutionary algorithm is proposed based on the following two strategies: 1) decomposition strategy: the MO-MFO problem is transformed into a series of single-objective optimization subproblems by multiple sets of weight vectors and a scalarizing function, and then a single population is used to optimize these single-objective problems simultaneously; 2) dynamic resource allocation strategy: during the evolution of the population, individuals with high scalar fitness will get more investments, that is, more computing resources will be allocated to them, and the scalar fitness of each individual is updated periodically. In the experiment, MO-MFEA, MOEA/D-DE and MOEA/D-DRA are selected as the comparison algorithms. The experimental results show that the proposed algorithm is very competitive with or even superior to the comparison algorithms on both a variety of benchmark problems and a practical problem.

Further experiments show that the computational resources required for different single-objective subproblems are different. To be more precise, the subproblems that get more rewards should be allocated more investments. In addition, the experimental results and analysis on the influence of random matching probability parameter show that the knowledge transfer among different tasks can improve the optimization effect; however, too frequent knowledge transfer between different tasks will result in poor performance of the algorithm. Therefore, a proper probability of knowledge transfer between different tasks is very important. Further studies will explore the relevance between tasks specifically, so as to maximize the favorable knowledge transfer and reduce the negative transfer between tasks.

Declaration of Competing Interest

None.

CRediT authorship contribution statement

Shuangshuang Yao: Conceptualization, Methodology, Software. **Xianpeng Wang:** Data curation, Writing - original draft.

Acknowledgements

This research was supported by the National Key Research and Development Program of China (2018YFB1700404), the Fund for the National Natural Science Foundation of China (61573086), the Major Program of National Natural Science Foundation of China (71790614), the Fund for Innovative Research Groups of the National Natural Science Foundation of China (71621061), the Major International Joint Research Project of the National Natural Science Foundation of China (71520107004), and the 111 Project (B16009).

References

[1] R.B. Agrawal, K. Deb, R.B. Agrawal, Simulated binary crossover for continuous search space, Complex Syst. 9 (2) (1995) 115–148.
[2] J. Bader, E. Zitzler, Hype: an algorithm for fast hypervolume-based many-objective optimization, Evol. Comput. 19 (1) (2011) 45–76.
[3] K.K. Bali, Y. Ong, A. Gupta, P.S. Tan, Multifactorial evolutionary algorithm with online transfer parameter estimation: MFEA-II, IEEE Trans. Evol. Comput. (2019). 1–1. doi: 10.1109/TEVC.2019.2906927.
[4] M. Basseur, E. Zitzler, Handling uncertainty in indicator-based multiobjective optimization, Int. J. Comput. Intell.Res. 2 (3) (2006) 255–272.
[5] X. Cai, H. Sun, Z. Fan, A diversity indicator based on reference vectors for many-objective optimization, Inf. Sci. 430–431 (2018) 467–486.
[6] R. Chandra, A. Gupta, Y.-S. Ong, C.-K. Goh, Evolutionary multi-task learning for modular knowledge representation in neural networks, Neural Process. Lett. 47 (3) (2018) 993–1009.
[7] R. Chandra, Y.-S. Ong, C.-K. Goh, Co-evolutionary multi-task learning with predictive recurrence for multi-step chaotic time series prediction, Neurocomputing 243 (2017) 21–34.
[8] X. Chen, Y. Ong, M. Lim, K.C. Tan, A multi-facet survey on memetic computation, IEEE Trans. Evol. Comput. 15 (5) (2011) 591–607.
[9] Y. Chen, J. Zhong, L. Feng, J. Zhang, An adaptive archive-based evolutionary framework for many-task optimization, IEEE Trans. Emerging Top.Comput. Intell. (2019) 1–16. In press. doi: 10.1109/TETCI.2019.2916051.
[10] Y. Chen, J. Zhong, M. Tan, A fast memetic multi-objective differential evolution for multi-tasking optimization, in: 2018 IEEE Congress on Evolutionary Computation (CEC), 2018, pp. 1–8, doi:10.1109/CEC.2018.8477722.
[11] I. Das, J.E. Dennis, Normal-boundary intersection: a new method for generating the pareto surface in nonlinear multicriteria optimization problems, SIAM J. Optim. 8 (3) (1998) 631–657.
[12] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197.
[13] K. Deb, S. Tiwari, Omni-optimizer: a generic evolutionary algorithm for single and multi-objective optimization, Eur. J. Oper. Res. 185 (3) (2008) 1062–1087.
[14] J. Ding, C. Yang, Y. Jin, T. Chai, Generalized multitasking for evolutionary optimization of expensive problems, IEEE Trans. Evol. Comput. 23 (1) (2019) 44–58.
[15] J.J. Durillo, A.J. Nebro, F. Luna, E. Alba, On the effect of the steady-state selection scheme in multi-objective genetic algorithms, in: M. Ehrgott, C.M. Fonseca, X. Gandibleux, J.-K. Hao, M. Sevaux (Eds.), Evolutionary Multi-Criterion Optimization, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 183–197.
[16] M. Emmerich, N. Beume, B. Naujoks, An EMO algorithm using the hypervolume measure as selection criterion, in: Proceedings of the Third International Conference on Evolutionary Multi-Criterion Optimization, in: EMO'05, Springer-Verlag, Berlin, Heidelberg, 2005, pp. 62–76.
[17] L. Feng, L. Zhou, J. Zhong, A. Gupta, Y. Ong, K. Tan, A.K. Qin, Evolutionary multitasking via explicit autoencoding, IEEE Trans. Cybern. 49 (9) (2019) 3457–3470.
[18] M. Gong, Z. Tang, H. Li, J. Zhang, Evolutionary multitasking with dynamic resource allocating strategy, IEEE Trans. Evol. Comput. (2019). 1–1. doi: 10.1109/TEVC.2019.2893614.
[19] A. Gupta, J. Mańdziuk, Y.-S. Ong, Evolutionary multitasking in bi-level optimization, Complex Intell. Syst. 1 (1) (2015) 83–95.
[20] A. Gupta, Y. Ong, L. Feng, Multifactorial evolution: toward evolutionary multitasking, IEEE Trans. Evol. Comput. 20 (3) (2016) 343–357.
[21] A. Gupta, Y. Ong, L. Feng, K.C. Tan, Multiobjective multifactorial optimization in evolutionary multitasking, IEEE Trans. Cybern. 47 (7) (2017) 1652–1665.
[22] D. Han, W. Du, W. Du, Y. Jin, C. Wu, An adaptive decomposition-based evolutionary algorithm for many-objective optimization, Inf. Sci. 491 (2019) 204–222.
[23] H. Li, Y. Ong, M. Gong, Z. Wang, Evolutionary multitasking sparse reconstruction: framework and case study, IEEE Trans. Evol. Comput. (2018). 1–1. doi: 10.1109/TEVC.2018.2881955.
[24] H. Li, Q. Zhang, Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II, IEEE Trans. Evol. Comput. 13 (2) (2009) 284–302.
[25] Z. Liang, W. Hou, X. Huang, Z. Zhu, Two new reference vector adaptation strategies for many-objective evolutionary algorithms, Inf. Sci. 483 (2019) 332–349.
[26] R. Liaw, C. Ting, Evolutionary many-tasking based on biocoenosis through symbiosis: a framework and benchmark problems, in: 2017 IEEE Congress on Evolutionary Computation (CEC), 2017, pp. 2266–2273.

[27] Z.-Z. Liu, Y. Wang, P.-Q. Huang, AnD: a many-objective evolutionary algorithm with angle-based selection and shift-based density estimation, Inf. Sci. (2018).
[28] W.K. Mashwani, A. Salhi, A decomposition-based hybrid multiobjective evolutionary algorithm with dynamic resource allocation, Appl. Soft Comput. 12 (9) (2012) 2765–2780.
[29] W.K. Mashwani, A. Salhi, Multiobjective evolutionary algorithm based on multimethod with dynamic resources allocation, Appl. Soft Comput. 39 (C) (2016) 292–309.
[30] K. Miettinen, Nonlinear Multiobjective Optimization, Kluwer Academic Publishers, Boston, 1999.
[31] B.L. Miller, D.E. Goldberg, et al., Genetic algorithms, tournament selection, and the effects of noise, Complex Syst. 9 (3) (1995) 193–212.
[32] J. Mo, Z. Fan, W. Li, Y. Fang, Y. You, X. Cai, Multi-factorial evolutionary algorithm based on M2M decomposition, in: Y. Shi, K.C. Tan, M. Zhang, K. Tang, X. Li, Q. Zhang, Y. Tan, M. Middendorf, Y. Jin (Eds.), Simulated Evolution and Learning, Springer International Publishing, Cham, 2017, pp. 134–144.
[33] A.J. Nebro, J.J. Durillo, M. Vergne, Redesigning the jMetal multi-objective optimization framework, in: Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation, ACM, 2015, pp. 1093–1100.
[34] O. Schuetze, X. Equivel, A. Lara, C.A. Coello Coello, Some comments on GD and IGD and relations to the Hausdorff distance, in: Proceedings of the 12th Annual Conference Companion on Genetic and Evolutionary Computation, ACM, 2010, pp. 1971–1974.
[35] R. Storn, K. Price, Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (4) (1997) 341–359.
[36] J. Sun, H. Zhang, A. Zhou, Q. Zhang, Learning from a stream of non-stationary and dependent data in multiobjective evolutionary optimization, IEEE Trans. Evol. Comput. (2018). 1–1. doi: 10.1109/TEVC.2018.2865495.
[37] L. Tang, X. Wang, Z. Dong, Adaptive multiobjective differential evolution with reference axis vicinity mechanism, IEEE Trans. Cybern. 49 (9) (2019) 3571–3585.
[38] Z. Tang, M. Gong, M. Zhang, Evolutionary multi-task learning for modular extremal learning machine, in: 2017 IEEE Congress on Evolutionary Computation (CEC), 2017, pp. 474–479.
[39] X. Wang, Z. Dong, L. Tang, Multiobjective differential evolution with personal archive and biased self-adaptive mutation selection, IEEE Trans. Syst. Man. Cybern.: Syst. (2018) 1–13. In press. doi: 10.1109/TSMC.2018.2875043.
[40] X. Wang, L. Tang, An adaptive multi-population differential evolution algorithm for continuous multi-objective optimization, Inf. Sci. 348 (2016) 124–141.
[41] Y. Yuan, Y. Ong, L. Feng, A.K. Qin, A. Gupta, B. Da, Q. Zhang, K.C. Tan, Y. Jin, H. Ishibuchi, Evolutionary multitasking for multiobjective continuous optimization: Benchmark problems, performance metrics and baseline results. arXiv preprint arXiv:1706.02766, 2017.
[42] J. Zhang, A. Zhou, K. Tang, G. Zhang, Preselection via classification: a case study on evolutionary multiobjective optimization, Inf. Sci. 465 (2018) 388–403.
[43] Q. Zhang, H. Li, MOEA/D: a multiobjective evolutionary algorithm based on decomposition, IEEE Trans. Evol. Comput. 11 (6) (2007) 712–731.
[44] Q. Zhang, W. Liu, H. Li, The performance of a new version of MOEA/D on CEC09 unconstrained mop test instances, in: 2009 IEEE Congress on Evolutionary Computation, 2009, pp. 203–208.
[45] X. Zheng, A.K. Qin, M. Gong, D. Zhou, Self-regulated evolutionary multi-task optimization, IEEE Trans. Evol. Comput. (2019). 1–1. doi: 10.1109/TEVC.2019.2904696.
[46] A. Zhou, Q. Zhang, Are all the subproblems equally important? Resource allocation in decomposition-based multiobjective evolutionary algorithms, IEEE Trans. Evol. Comput. 20 (2016) 52–64.
[47] L. Zhou, L. Feng, J. Zhong, Y.-S. Ong, Z. Zhu, E. Sha, Evolutionary multitasking in combinatorial search spaces: a case study in capacitated vehicle routing problem, in: 2016 IEEE Symposium Series on Computational Intelligence (SSCI), IEEE, 2016, pp. 1–8.
[48] E. Zitzler, L. Thiele, J. Bader, On set-based multiobjective optimization, IEEE Trans. Evol. Comput. 14 (1) (2010) 58–79.
[49] J. Zou, L. Fu, S. Yang, J. Zheng, G. Ruan, T. Pei, L. Wang, An adaptation reference-point-based multiobjective evolutionary algorithm, Inf. Sci. 488 (2019) 41–57.