Applied Soft Computing Journal 89 (2020) 106104

Contents lists available at ScienceDirect

Applied Soft Computing Journal

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

A constrained multi-objective evolutionary algorithm based on decomposition and dynamic constraint-handling mechanism

Yongkuan Yang, Jianchang Liu *, Shubin Tan

Department of Information Science and Engineering, Northeastern University, Shenyang 110819, China
The State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang 110819, China

ARTICLE INFO

Article history:
Received 23 August 2019
Received in revised form 29 December 2019
Accepted 13 January 2020
Available online 21 January 2020

Keywords:
Constrained multi-objective optimization
MOEA/D
Constraint-handling techniques
Epsilon constraint-handling
Differential evolution

ABSTRACT

Constrained multi-objective optimization problems (CMOPs) are common in real-world engineering application, and are difficult to solve because of the conflicting nature of the objectives and many constraints. Some constrained multi-objective evolutionary algorithms (CMOEAs) have been developed for CMOPs, but they still suffer from the problems of easily getting trapped into local optimal solutions and low convergence. This paper introduces a multi-objective evolutionary algorithm based on decomposition and dynamic constraint-handling mechanism (MOEA/D-DCH) to tackle this issue. Firstly, the dynamic constraint-handling mechanism divides the search modes into the unconstrained search mode and the constrained search mode, which are dynamically adjusted by the generation number and the proportion of feasible solutions in the population. This mechanism could lead to a faster convergence than the traditional constraint-handling mechanisms. For the constrained search mode, an improved epsilon constraint-handling method is used to enhance the diversity of the population. Then, an individual update mechanism based on the best feasible solution of each sub-problem is designed to update the feasible individuals for maintaining the convergence of the feasible solutions. Finally, MOEA/D-DCH dynamically regulates the parameters of the differential evolution operator to enhance the local search ability. Experiments on 21 benchmark test functions are conducted to test MOEA/D-DCH and five other typical CMOEAs. Meanwhile, a real-world problem is employed to evaluate the practical performance of MOEA/D-DCH. MOEA/D-DCH achieves significantly better results than the other five algorithms on most of the test problems. The results indicate the effectiveness and competitiveness of MOEA/D-DCH for solving CMOPs.

1. Introduction

The unconstrained multi-objective optimization problem (UMOP) is well studied in the field of multi-objective optimization [1–3]. However, the constrained multi-objective optimization problem (CMOP) lacks research compared to the UMOP. Actually, the CMOPs are common in engineering application, for example, the problem of load distribution of a multi-source pipeline [4], the portfolio problems [5] and the path-planning problems [6]. Hence, there is a growing need to get better results for CMOPs. Evolutionary algorithms have been applauded for their population-based intelligence search when solving complicated optimization problems. This beneficial property has allowed a booming development for multi-objective evolutionary algorithms (MOEAs) during the past two decades. MOEAs have been reported to perform well on the UMOP; nevertheless, MOEAs face some difficulties in dealing with the constraints during the search process when directly applied to solve CMOPs [7,8]. Therefore, solving CMOPs by MOEAs has attracted much attention in evolutionary optimization.

In the area of evolutionary optimization, a constraint-handling technique (CHT) is used for handling equality or inequality constraints. Various CHTs have been attempted to deal with constrained single-objective optimization problems (CSOPs) [9] in the past two decades. Recently, some researchers have applied exiting CHTs that were initially developed for CSOPs to MOEAs for solving CMOPs [10–12]. Some novel CHTs for CMOPs are proposed as well [13,14]. Generally, the MOEAs for CMOPs are called constrained MOEAs (CMOEAs). According to the comprehensive review conducted by Coello [15], CHTs in CMOEAs can be generally divided into four broad categories as we summarize below: (1) methods based on penalty function, (2) methods based on separation of constraints and objectives, (3) methods based on retaining the infeasible solutions in the population, and (4) hybrid methods.

* Corresponding author at: The State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang 110819, China.
E-mail address: liujianchang@ise.neu.edu.cn (J. Liu).

https://doi.org/10.1016/j.asoc.2020.106104
1568-4946/

(1) Penalty function methods are certainly one of the most simply used CHTs, in which the constraint violation multiplied by a penalty factor is added to the objective functions. Although methods of this type are easy to be executed, they need careful tuning of their penalty factors which are often problem-dependent. A method depending on adaptive penalty factors was put forward by Woldesenbet et al. [10]. It appears simple to be implemented. However, this mechanism still has the problem of improper penalty factors.

(2) Methods based on separation of constraints and objectives try to prevent the demand for tuning the penalty factors. Representative methods of this type are stochastic ranking (SR) [11], constraint dominance principle (CDP) [16,17], epsilon constraint-handling technology (epsilon CHT) [18,19] and so forth. (1) SR stochastically manages the balance between the objectives and constraints. In SR, a user-defined parameter p_f decides the criterion for comparing infeasible solutions, and the criterion is either the value of constraint violation or the objective function. Although SR can obtain some infeasible solutions with better objectives to enhance the diversity of individuals, these superior infeasible solutions cannot be always held in the population, which leads to insufficient convergence; (2) In CDP, the feasible solutions are better than the infeasible ones. For infeasible solutions, the constraint violation is as small as possible. Although CDP is the most popular CHT for CMOPs, it favors feasible solutions over infeasible ones, which may fail to take advantage of the useful information of infeasible solutions; (3) In epsilon CHT, an infeasible individual will be treated as a feasible one in case that its degree of constraint violation is equal to or less than ε. If $\varepsilon = 0$, the epsilon CHT is same as CDP. The epsilon CHT shows better performance for CMOPs with small feasible region in search space. However, it is a tough work to control ε properly.

(3) In methods based on retaining the infeasible individuals in the population, Ray et al. [12] and Peng et al. [20] treated the constraints as another objective, then a CMOP with m objectives was transformed to one with $m+1$ objectives. Therefore, many infeasible solutions could be saved during the evolution process. A parameter-free CHT dependent on the rank of constraint violation degree has been proposed more recently by Ning et al. [21], in which the individuals with low Pareto rank and low rank of the constraint violation will be selected. These methods make use of infeasible solutions; however, there are huge numbers of infeasible individuals in the population, which may cause low convergence speed of feasible solutions.

(4) The hybrid methods integrate different CHTs to handle constraints [7,13,22]. Here we introduce four specific methods. The first is the ensemble of constraint handling methods [22]. It separates the population into three subpopulations, each of which is generated by an independent, unique CHT. The second is the adaptive trade-off model [23] consisting of two different CHTs, one treating constraints as another objective and the other being adaptive penalty functions. The third is the push and pull search (PPS) [13] that divides the search process into two phases: the push phase and the pull phase. In the push phase, PPS pushes the population to the unconstrained Pareto front without handling the constraints. In the pull phase, PPS pulls the population back to the Pareto front. The fourth is another two-stage framework for CMOPs named ToP [14]. In the first stage of ToP, a CMOP is changed into a CSOP by weighting the objective functions into one objective function, and an optimal feasible solution is obtained by solving this CSOP; in the second stage, a CMOEA is adopted to achieve the Pareto optimal feasible solutions.

Taken together, by directly combining the above mentioned CHTs with MOEAs, many CMOEAs have thus been proposed. This architecture provides a simple paradigm for CMOPs. But when solving complicated CMOPs, this architecture may result in unsatisfactory performance such as low convergence speed and poor distribution. How to integrate the particular CHTs and the specific MOEAs in a proper way is worth to study. Besides, developing better CHTs is another trend for CMOPs. Recently, a version of multi-objective evolutionary algorithm based on decomposition (MOEA/D) employing the differential evolution (DE) operator, named MOEA/D-DE [24], has been proposed, which performs well on UMOPs with complicated Pareto sets. A few CMOEAs based on MOEA/D-DE have been proposed for CMOPs [11,13,18], but they suffer from the problems of easily getting trapped into local optimal solutions and/or low convergence. In an attempt to surmount the shortcomings, this paper intends to propose a novel CMOEA, based on decomposition and dynamic constraint-handling mechanism, termed MOEA/D-DCH. MOEA/D-DCH inherits the algorithmic framework of MOEA/D-DE. Compared with the existing decomposition-based CMOEAs, the main contributions of MOEA/D-DCH are as follows:

- A dynamic constraint-handling (DCH) mechanism dynamically adjusts the unconstrained search and constrained search by the generation number and the proportion of feasible solutions in the population. Compared to other constrained handling mechanisms, DCH makes full use of unconstrained search to help the population pass through the infeasible region of objective space, while making the constrained search trade off the feasibility and convergence.
- The constrained search in DCH is designed as an improved epsilon CHT where the setting of epsilon value is adaptively adjusted based on the minimum constraint violation of the population to maintain the diversity. In this way, some infeasible solutions with low constraint violation and better objective values will always be maintained in the population to help the algorithm get more potential feasible solutions in the boundary regions.
- To solve the problem of missing feasible solutions which are replaced by the infeasible ones, the best feasible solution of each sub-problem is saved to an elite feasible solution set and takes part in the procedure of individual-updating to ensure that the feasible solutions of the population are always optimal.
- The proposed algorithm dynamically adjusts the parameter scale of reproduction operator to reinforce the local search capability and further improve the distribution of the population.

The remainder of this paper is structured as follows. Section 2 gives a brief introduction of CMOPs and the cornerstone algorithm MOEA/D-DE. Section 3 gives a deep description of our proposed algorithm MOEA/D-DCH. In Section 4, the effectiveness of MOEA/D-DCH versus other five state-of-the-art algorithms is then validated through comprehensive experiments. The effectiveness of our proposed strategy and the sensitivity of parameters are also further discussed. Finally, Section 5 presents the conclusion and outlines our future work.

2. Preliminaries

2.1. Definition of the constrained multi-objective optimization problem

In our study, the form of the CMOP is mathematically defined as follows:

$$min\ f(x) = [f_1(x)\, f_2(x) \cdots\ f_m(x)],\ \ x = (x_1, \cdots, x_n) \in R^n$$
$$s.t. \begin{cases} h_k(x) = 0 & k = 1, 2, \cdots, K \\ g_j(x) \le 0 & j = 1, 2, \cdots, J \end{cases} \tag{1}$$

where m is the amount of conflicting objectives, $x \in R^n$ is the n-dimensional vector of decision variables, k and j are the respective number of equality constraints and inequality constraints. The search space of a decision vector x is denoted as $S = \left\{x = (x_1, \ldots, x_n)^T \,|L_i \le x_i \le U_i, i = 1, \ldots, n\right\}$. L_i and U_i are the upper bound and lower bound of decision variable x_i, $i = 1, \ldots, n$, respectively.

The constraint violation $\phi(x)$ of a solution x is calculated as:

$$\phi(x) = \sum_j \left\|\max\{0, g_j(x)\}\right\| + \sum_k \left\|\max\{0, |h_k(x)|\}\right\|. \quad (2)$$

If $\phi(x) = 0$, a solution $x = (x_1, \ldots, x_n)^T$ is called a feasible solution. Or else if $\phi(x) > 0$, x is an infeasible solution. Therefore, the search space is divided into two regions: feasible domain and infeasible domain. All the feasible solutions constitute the feasible domain, which can be defined as $\mathrm{T_f} = \{x \in S \subset R^n \,|\phi(x) = 0\}$. All the infeasible solutions constitute the infeasible domain. A solution x is said to dominate another solution y (denoted by $x \prec y$), if both of the following conditions are true: (1) $f_i(x) \le f_i(y)$ for all $i = 1, \ldots, m$, and (2) $f_i(x) < f_i(y)$ in at least one f_i. Let x^* belong to one set, x^* is assumed as a Pareto optimal solution of this set, if no other solution y in this set dominates x^*. The whole Pareto optimal solutions in $\mathrm{T_f}$ make up the Pareto set PS. The Pareto front (PF) is referred to $PF = \{F(x)|x \in PS\}$. The unconstrained PF is defined by *unconstrained* $PF = \{F(x)|x \in PS'\}$ where PS' is the set of the Pareto optimal solution in S.

2.2. MOEA/D-DE algorithm

Accounting for the fact that the appropriate generation of the PF can be obtained by a set of well-distributed weight vectors, Zhang and Li [25] presented a multi-objective evolutionary algorithm based on decomposition (MOEA/D), which decomposes a multi-objective optimization problem (MOP) into N_p scalar-objective optimization sub-problems by N_p weight vectors. N_p is the population size. All the sub-problems are simultaneously optimized. The objective function of each scalar-objective optimization sub-problem is described by the aggregation function which is related to the weight vector and the objective functions of the MOP. Being a version of MOEA/D, MOEA/D-DE [24] uses the weighted Tchebycheff function as the aggregation function. The weighted Tchebycheff function is stated as follows:

$$\text{minimize} \quad g^{te}\left(x \left|W_i, Z^*\right.\right) = \max_{j=1}^{m}\left\{W_i^j \left|f_i(x) - Z_i^*\right|\right\} \quad (3)$$

where $W_i = \left(W_i{}^1, \ldots, W_i{}^m\right)^T \in R^m$ is number i weight vector, i.e., $W_i^j \ge 0 \ \forall i = 1, \ldots, m$ and $\sum_{i=1}^m W_i^j = 1$; $Z^* = \left(Z_1^*, \ldots, Z_m^*\right)^T$ is the ideal point, i.e., $Z_i^* = \min\{f_i(x)\,|x \in R^n\}$. The expression $g^{te}(x\,|W_i, Z^*)$ is also named as the aggregation function of solution x.

MOEA/D-DE [24] selects the DE operator [26] as the search engine. DE is firstly introduced by Storn and Price in 1997 [27]. It generates the trail vector $u_i = \{u_i^1, u_i^2, \ldots, u_i^n\}$ and the formula is as follows:

$$u_i^j = \begin{cases} x_{r1}^j + F(x_{r2}^j - x_{r3}^j) & \text{if } rand < CR \\ x_i^j & \text{else} \end{cases} \quad (4)$$

where x_{r1}, x_{r2} and x_{r3} are three different individuals randomly selected from the population, F and CR are the respective scale coefficient and crossover probability, *rand* is a random number from 0 to 1. After the trail vector has been generated, the offspring individual is finally obtained by polynomial mutation [16].

3. Our proposed algorithm: MOEA/D-DCH

3.1. The framework of the proposed algorithm

On the basis of the framework of MOEA/D-DE, MOEA/D-DCH is able to decompose a constrained multi-objective optimization problem into a number of constrained scalar-objective optimization sub-problems. In MOEA/D-DCH, the constrained search and unconstrained search are dynamically executed by the dynamic constraint-handling mechanism in order to enhance the convergence of the population, which will be introduced in Section 3.2. For the constrained search, we develop an improved epsilon CHT to maintain the diversity of the population by changing the setting value of ε, which will be introduced in Section 3.3. To ensure the convergence of feasible solutions, MOEA/D-DCH applies an elite feasible solution set $M = \left[M^1, M^2, \ldots, M^{N_p}\right]$. Specifically, M is initialized by the initial population, and the best feasible solution for the sub-problem with the number i is always preserved to M^i of M. In the whole search process, the elite feasible solution set takes part in the updating procedure of feasible individuals to ensure the convergence of feasible solutions in each sub-problems. This process will be introduced in Section 3.4. Meanwhile, the feasible individuals are added to an external archive, which is the output of MOEA/D-DCH. A maintaining strategy is required to limit the individual number of the external archive. In MOEA/D-DCH, the maintaining strategy for the external archive is similar to the environmental selection of NSGAII [17]. The difference is that the objective space is normalized when MOEA/D-DCH calculates the crowding distance.

The general steps of MOEA/D-DCH are listed in Algorithm 1. Lines 1–6 initialize some parameters in MOEA/D-DCH. First, a CMOP is decomposed into N_p scalar-objective optimization sub-problems which are associated with N_p weight vectors. Then, the population P, the neighbor indexes $B(i)$, the elite feasible solution set M, the generation number G, the ideal point Z^* and the epsilon value ε are initialized. To be specific, the Euclidean distance is used to measure the closeness between any two weight vectors when calculating $B(i)$. Lines 9–11 generate a new solution y and update the ideal point. Line 12 adds the new feasible solution to the external archive. Line 13 calculates the dynamic adjustment factor d_f based on the dynamic constraint-handling mechanism. In Line 14, the parameter c is used to record the number of solutions replaced by the new solution y. Lines 15–27 update the solution of each sub-problem. Specifically, Lines 15–18 choose the solution of number j sub-problem to be updated, i.e., x^j. Lines 19–26 are the comparison between y and x^j based on the dynamic constraint-handling mechanism, among which Lines 20–23 are the comparison based on the epsilon CHT. Line 25 is the individual update schedule based on the elite feasible solution set. Line 29 updates the parameters of DE operator which will be fully introduced in Section 3.5. Line 30 maintains the external archive. Line 32 calculates ε based on the improved epsilon CHT.

3.2. Dynamic constraint-handling mechanism

Constrained search can trap the population in the local feasible region prematurely whereas unconstrained search can help the population get out of the locally optimal areas. Therefore, we propose a dynamic constraint-handling (DCH) mechanism to enhance the convergence of the population. DCH adaptively adopts constrained search and unconstrained search during the search process. In constrained search, the individual comparison is done through the constraint violation and the objective functions. In unconstrained search, the individual comparison only relies on the objective functions. DCH dynamically adjusts the

Algorithm 1: MOEA/D-DCH

Input: Population size N_p; a set of unit weight vectors $W = \{W_1, W_2, \cdots, W_{Np}\}$;
The size of neighborhood T; the probability of selection from neighbors σ;
Maximal generation number T_{max}; maximal number of solutions replaced by an offspring n_r;
Output: The non-dominated individuals of external archive.

1: Decompose a CMOP into N_p sub-problems associated with weight vectors;
2: Randomly generate a population $P = \{x^1, \ldots, x^{Np}\}$;
3: For each $i = 1, \ldots, N_p$, set $B(i) = \{i^1, \ldots, i^T\}$, where $i^1, \ldots, i^T$ are the T closest weight vector to W_i;
4: The initial elite feasible solution set $M = \{M^1, \ldots, M^{Np}\} = P$;
5: G=0;
6: Calculate the ideal point Z^* and the initial epsilon value $\varepsilon(0)$;
7: **while** $G < T_{max}$ **do**
8:   **for** i=0 to N_p **do**
9:     **if** rand <σ **then** $S = B(i)$ **else** $S = \{1, \ldots, N_p\}$ **end**;
10:     Set $r1=i$ and select two indexes $r2$ and $r3$ from S randomly, and $r2 \neq r3$;
11:     Generate an offspring y from x^{r1}, x^{r2} and x^{r3} by DE operator and polynomial mutation;
12:     **if** $\phi(y)$==0 **then** add y to external archive **end**;
13:     Calculate the dynamic adjustment factor d_f;
14:     Set c=0;
15:     **if** c== n_r or S is empty **then**
16:       Continue;
17:     **else**
18:       Randomly pick an index j from S;
19:       **if** $rand > d_f$ **then**
20:         **if** ($\phi(y) \le \varepsilon(G)$ and $\phi(x^j) \le \varepsilon(G)$) or $\phi(y) == \phi(x^j)$ **then**
21:           **if** $g^{te}(y \mid W_j) < g^{te}(x^j \mid W_j)$ **then** turn to step 25 **end**;
22:         **else if** $\phi(y) < \phi(x^j)$ **then** turn to step 25;
23:         **end**
24:       **else if** $g^{te}(y \mid W_j) < g^{te}(x^j \mid W_j)$ **then**
25:         Update-individual(y, x^j, M^j) according to Algorithm 2 and c=c+1;
26:       **end**
27:     **end**
28:   **end**
29:   Update the parameters of DE operator;
30:   Maintain the external archive;
31:   G=G+1;
32:   Calculate $\varepsilon(G)$;
33: **end**

unconstrained search and constrained search by the dynamic adjustment factor d_f as Eq. (5) shows:

$$d_f = k \cdot {r_f}^G \cdot (1 - G/T_{\max}) \tag{5}$$

where $T_{\max}$ is the maximal generation number, ${r_f}^G$ is the proportion of feasible solutions in the population at number G generation, k is a scale parameter. The dynamic adjustment factor d_f changes with the proportion of feasible solutions and the generation number. Supposing that a uniformly generated random number *rand* from 0 to 1 is more than d_f, the individual comparison is based on the constrained search mode; otherwise, the individual comparison is based on the unconstrained mode. Obviously, the constrained search is more concerned about the feasibility of solutions while the unconstrained search focuses more on the convergence. As Eq. (5) shows, DCH is more inclined for constrained search with the increasing generation number; DCH is more inclined for unconstrained search with the increase of the proportion of feasible solutions in the population.

3.3. The improved epsilon constraint-handling mechanism

When DCH is applied, the working behavior of unconstrained search and constrained search varies according to the corresponding MOEA. In our proposed algorithm MOEA/D-DCH, for the unconstrained search, the individual comparison is based on the Tchebycheff aggregation function; for the constrained search, the epsilon CHT is used to compare the individuals. The individual comparison based on the epsilon CHT is also called the epsilon-constrained comparison. For MOEA/D-DCH, the epsilon-constrained comparison is according to the Tchebycheff aggregation function g^{te}, the constraint violation ϕ and the epsilon value

ε. Specifically, solution x is better than solution y if the following conditions are satisfied [18]:

$$\begin{cases} g^{te}(x) < g^{te}(y), & \text{if } \phi(x) \le \varepsilon, \phi(y) \le \varepsilon \\ g^{te}(x) < g^{te}(y), & \text{if } \phi(x) = \phi(y) \\ \phi(x) < \phi(y), & \text{otherwise} \end{cases} \tag{6}$$

In [19], the setting of ε in epsilon CHT is always seen to decrease along the evolution process. Under the condition that ε is smaller than the smallest value of the constraint violation in the infeasible solutions, the epsilon CHT performs as the same as the CDP [28]. Nonetheless, as a result, the individual comparison between the infeasible solutions is only based on the constraint violation, and the diversity of the population cannot be well guaranteed. In response to this problem, a self-adaptive epsilon setting mechanism by means of the minimum value of constraint violation is proposed as Eq. (7) shows.

$$\varepsilon(G) = \begin{cases} (1-\tau)\,\varepsilon(G-1), & \text{if } {r_f}^G \le {r_d}^G \\ (1+\tau)\,{\phi^G}_{\min}, & \text{if } \varepsilon(G-1) < {\phi^G}_{\min}, {r_f}^G > {r_d}^G \\ {\phi^G}_{\min}, & \text{if } (1-\tau)\,\varepsilon(G-1) < {\phi^G}_{\min}, {r_f}^G = 0 \\ \varepsilon(G-1), & \text{else} \end{cases} \tag{7}$$

where ${r_d}^G = {r_f}^0 + (1-{r_f}^0) * G/T_{\max}$, ${r_f}^0$ is the proportion of feasible solutions in the initial population, $\phi^G_{\min}$ is the minimum constraint violation value of infeasible individuals at number G generation, τ is the expansion factor dominating the changing amplitude of ε. The setting of $\varepsilon(0)$ is the same as the method in [19].

Algorithm 2: The individual update strategy

```
Input:  Offspring y; individual x^j chosen to be updated by y;
        individual M^j of elite feasible population set
        (j is the individual serial number);
Output: The updated x^j, the updated M^j;
if x^j is feasible and y is feasible
    x^j = y;
    M^j = y;
else if x^j is infeasible and y is feasible
    if M^j is feasible and better than y
        x^j = M^j;
    else
        x^j = y;
        M^j = y;
    end
else
    x^j = y;
end
```

Due to the effect of the unconstrained search, the whole population may be infeasible (${r_f}^G = 0$). Under this situation, the $\varepsilon(G)$ will approach ${\phi^G}_{\min}$ if $(1-\tau)\,\varepsilon(G-1) < {\phi^G}_{\min}$. In this way, the population is capable of moving to the feasible region while keeping the diversity of solutions. At the stage with feasible solutions in the population, if $r_f^G > {r_d}^G$ and $\varepsilon(G-1) < {\phi^G}_{\min}$, $\varepsilon(G)$ is set to $(1+\tau)\,{\phi^G}_{\min}$, which can enhance the searching ability at the infeasible areas and obtain feasible solutions in other regions. Once $r_f^G \le {r_d}^G$, $\varepsilon(G)$ will decrease to $(1-\tau)\,\varepsilon(G-1)$, which promotes the population to obtain more feasible solutions. Or else, the $\varepsilon(G)$ stays unchanged. As the number of iteration rises, ${r_d}^G$ will gradually increase. Therefore, the proportion of feasible solutions in the population will increase at a low speed, and some infeasible solutions with low constraint violation and better objective values will always be maintained in the population. To sum up, this mechanism stops the algorithm from prematurely converging to local optimal feasible solutions through maintaining the diversity of the population.

3.4. Individual update schedule

As stated above, MOEA/D-DCH dynamically adjusts the unconstrained search and constrained search. During the unconstrained search, the infeasible solution with the smaller value of Tchebycheff aggregation function will update the feasible solution. In the following constrained search process, these infeasible solutions will be updated by another feasible individual whose value of Tchebycheff aggregation function is bigger than the former feasible individual. Therefore, the convergence of feasible solutions cannot be held. It is explicit that this situation should be avoided. We design an individual update schedule based on the elite feasible solution set as Algorithm 2 shows. The individual x^j being chosen to be updated, the offspring y and the M^j in M are the three input parameters of this schedule.

An offspring replacing an individual of the population can be divided into four situations: feasible offspring replaces the feasible individual; feasible offspring replaces the infeasible individual; infeasible offspring replaces the feasible individual; infeasible offspring replaces the infeasible individual. In the situation of feasible offspring replacing the feasible individual, x^j and M^j are replaced by y. In situation of feasible offspring replacing the infeasible individual, if M^j is feasible and its value of Tchebycheff aggregation function is smaller than y, x^j is replaced by M^j; otherwise, x^j and M^j are replaced by y. In the other two situations, x^j is replaced by y. In this way, the best feasible solution for number i sub-problem is preserved to M^i of M; the individual x^j is always the current optimal feasible solution for number j sub-problem if it is a feasible solution. Therefore, the convergence of the feasible solutions in each sub-problem is maintained.

3.5. Dynamic adjustment of DE parameters

The crossover rate CR is one of the two parameters of the DE operator. The range of CR is from 0.0 to 1.0. A high crossover rate will cause a high difference between the trail vector and the parent individual. For CMOPs, the value of the crossover rate CR is generally set to 1.0 to obtain individuals with higher variability [11,13,18]. We apply the same setting way of CR in MOEA/D-DCH. The F is the other parameter of DE operator. The range of F is generally from 0.0 to 1.0. From the current research results, we can get which parameter setting of F is conducive to global search, and which parameter setting is beneficial to local search [29,30]. A larger F parameter favors global search, and a smaller F parameter facilitates local search. For the aspect of the parameter setting of F at the initial stage, if F is set too low, the diversity of solutions may decrease as the algorithm runs; if F is set too large at the initial stage, the convergence of the algorithm will be too slow. Therefore, the initial value of F usually sets to be 0.5. In DCH, the population will converge as the number of iterations increases. When the population converges, we need to enhance the local search to help the population get more solutions of PF. Therefore, we design a dynamic adjustment mechanism of F parameter as Eq. (8) shows. With the increase of the number of iterations, the change range of F expands. Specifically, the upper bound of F is always 0.5 and the lower bound of F is changed from 0.5 to 0.0. In this way, the local search will be enhanced as the generation increases because the smaller

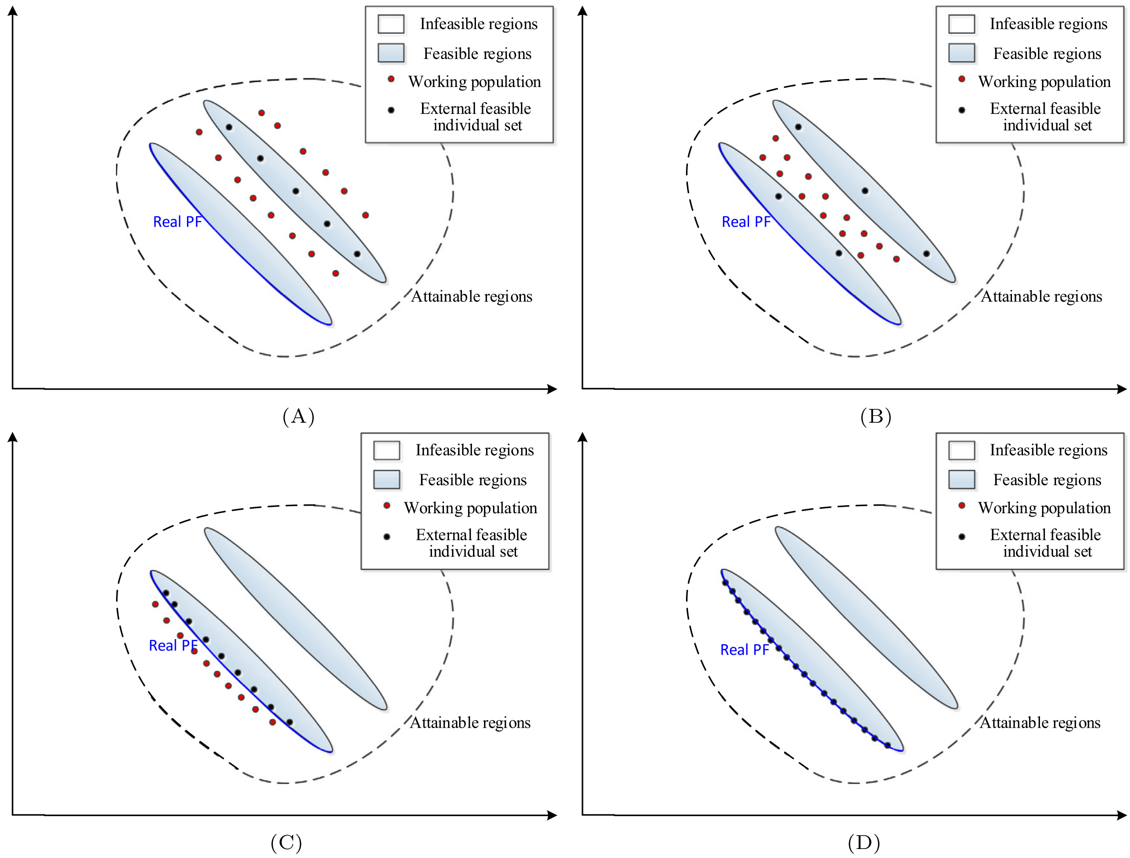

Fig. 1. Procedure of DCH.

value of F could also be achieved.

$$\begin{cases} CR = 1.0 \\ F = 0.5 \cdot (1 - rand \cdot G/T_{\max}) \end{cases} \tag{8}$$

Totally, Fig. 1 further demonstrates the process of MOEA/D-DCH. Firstly, MOEA/D-DCH achieves some feasible solutions in the local feasible space; then, MOEA/D-DCH quickly traverses the local feasible region and reaches the PF; next, some individuals pass through the (current) PF to achieve infeasible solutions with better objective values; in the end, MOEA/D-DCH returns to the PF and realizes a better distribution with the help of dynamic adjustment strategy of DE parameters.

4. Experimental studies

In this section, comprehensive experiments are carried out on 21 benchmark test problems including fourteen large infeasible region CMOPs (LIRCMOPs) [13,31] and seven constrained function (CF) test problems [32], as well as a real-world CMOP, to compare MOEA/D-DCH with five most-known representative CMOEAs, namely, MOEA/D-CDP [11], MOEA/D-SR [11], MOEA/D-IEpsilon [18], MOEA/D-PPS [13] and NSGAIII [33] on LIRCMOPs.

In the following subsections, we first briefly present the benchmark test functions and two performance metrics utilized in the tests. Next, we describe the parameter settings for the implemented algorithms. Afterwards, each algorithm carries out 30 independent runs on each test function. We use the Wilcoxon rank sum test (at a 5% significance level) to investigate whether MOEA/D-DCH is the best in a statistical sense. Then, effectiveness of different roles of MOEA/D-DCH is analyzed. In the end of this section, a real-world CMOP is employed to further examine the algorithm performance.

4.1. Benchmark test problems

The constraint functions of LIRCMOPs consist of controllable distance functions and shape functions [34]. The distance functions control the degree of convergence difficulty of the algorithm. The shape functions make the PF shapes convex or concave. All the fourteen LIRCMOPs have two objective functions, except for LIRCMOP13 and LIRCMOP14 which have three objective functions. As Fig. 2 shows, there is an extremely small feasible region for LIRCMOPs from LIRCMOP1 to LIRCMOP4, which is hard to obtain feasible solutions for CMOEAs. The PFs of LIRCMOP1 and LIRCMOP2 are continuous whereas the PFs of LIRCMOP3 and LIRCMOP4 are segmented continuous. Furthermore, the PFs of the four LIRCMOPs are located in the boundary of constraints. For the other LIRCMOPs, the objective space is separated by numerous infeasible areas, which makes it difficult for the normal constraint processing mechanism to quickly find the actual PF. Specifically, the PFs of LIRCMOPs from LIRCMOP5 to LIRCMOP8 are continuous. The PFs of LIRCMOP5 and LIRCMOP6 are same to the unconstrained PF. The PFs of LIRCMOP7 and LIRCMOP8 are located in the boundary of constraints. The PFs of LIRCMOP9 and LIRCMOP10 are segmented continuous whereas the PFs of LIRCMOP11 and LIRCMOP12 are composed of some points. The seven CF problems have two objectives with one or two nonlinear constraints [32]. The PFs of the seven CF problems are also shown in Fig. 2. Compared to LIRCMOPs, the CFs do not have large infeasible region before their PFs. Specifically, the PFs of CF1–3 are parts of their unconstrained PF. For CF4–7, their PFs are

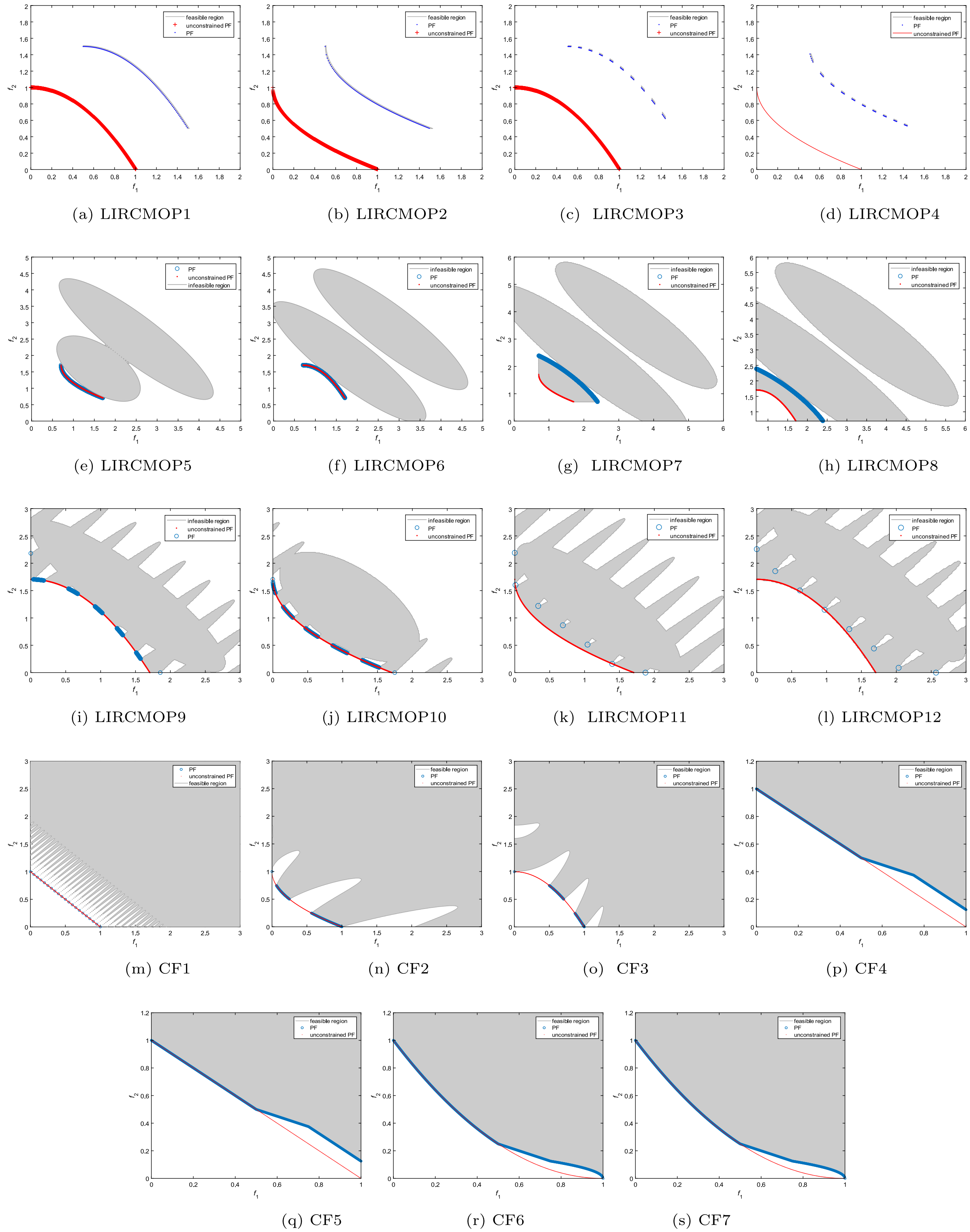

Fig. 2. Illustrations of the PF for the benchmark test problems.

partly located in the unconstrained PF and partly located in the boundary of constraints.

4.2. Performance metrics

To compare and qualify the results gained by MOEA/D-DCH with those gained by other five compared algorithms, two performance metrics are considered: one is the convergence representing the closeness of the non-dominated individuals towards the PF; the other is the diversity checking the distribution of the non-dominated individuals along the PF. We adopt two indicators which are able to simultaneously measure the convergence and diversity. One is the hypervolume (HV) indicator [25]. The other is the inverted generational distance (IGD) indicator [35].

Table 1
Setting of the HV reference point for all the benchmark test instances.

Problem	Reference point	Problem	Reference point	Problem	Reference point
LIRCMOP1	(1.65, 1.65)	LIRCMOP2	(1.65, 1.65)	LIRCMOP3	(1.59, 1.65)
LIRCMOP4	(1.59, 1.65)	LIRCMOP5	(1.88, 1.88)	LIRCMOP6	(1.88, 1.88)
LIRCMOP7	(2.63, 2.63)	LIRCMOP8	(2.63, 2.63)	LIRCMOP9	(2.04, 1.40)
LIRCMOP10	(1.92, 1.88)	LIRCMOP11	(2.06, 2.41)	LIRCMOP12	(2.83, 2.48)
LIRCMOP13	(1.88, 1.88, 1.88)	LIRCMOP14	(1.93, 1.93, 1.93)	CF1	(1.10, 1.10)
CF2	(1.10, 1.10)	CF3	(1.10, 1.10)	CF4	(1.10, 1.10)
CF5	(1.10, 1.10)	CF6	(1.10, 1.10)	CF7	(1.10, 1.10)

The HV metric measures the hypervolume bounded by a reference point and the objective space dominated by the approximate PF obtained by a CMOEA. The algorithm with a higher HV value is supposed to be superior than its counterparts. The reference points used for the benchmark test problems are listed in Table 1.

$$HV = \vee\left(\bigcup_{f\in PF_{approx}} [f_1, r_1] \times \cdots [f_m, r_m]\right) \tag{9}$$

where $\vee$ denotes the Lebesgue measure, PF_{approx} is the approximate PF obtained by the non-dominated individuals of an algorithm, $r \in R^m$ is a reference point dominated by PF.

The definition of IGD is seen as follows:

$$IGD = \sum_{x\in P^*} d(x, PF_{approx}) \Big/ \left|P^*\right| \tag{10}$$

where P^* is a set of points uniformly scattered along the PF, $d(x, PF_{approx})$ measures the minimum Euclidean distance between $x \in P^*$ and the points of PF_{approx}. In contrast, the smaller value of IGD metric is, the better performance of the algorithm is.

4.3. Parameters of MOEA/D-DCH and other CMOEAs

In this subsection, we provide a general presentation of the parameters in the experimental comparison. The detailed settings of parameters for the six CMOEAs are summarized as follows:

(1) Population size: $N_p = 300$ for LIRCMOPs, and $N_p = 100$ for CFs. Terminal condition: for LIRCMOPs and CFs, the algorithm terminates when the number of function evaluations succeeds 150,000 and 100,000, respectively. The number of runs: running 30 times for each algorithm on each benchmark test problem.
(2) Specific parameter settings in each algorithm: all the specific parameter settings for the peer algorithms are identical to the optimal settings suggested in the related articles [11, 13,18,33]; for the proposed algorithm MOEAD-DCH, $k = 10$, $\tau = 0.02$, the probability of choosing parents locally $\sigma = 0.9$, the neighborhood size T is set as 20, the mutation probability of polynomial mutation $P_m = 1/n$ and the distribution index of polynomial mutation is 20.

4.4. Experimental results on LIRCMOP test problems

After 30 independent runs, the statistical results of the IGD values and HV values achieved by our proposed method (i.e., MOEA/D-DCH) and those obtained by the other five algorithms on fourteen LIRCMOP test problems are summarized in Tables 2 and 3, respectively, in which the best results are highlighted in bold.

Under the IGD metric, according to the highlighted results in Table 2, it is obvious that MOEA/D-DCH obtains the best performance among the six compared algorithms on ten out of the fourteen test functions, while MOEA/D-PPS gets the best result on the rest four test problems including LIRCMOP5, LIRCMOP6, LIRCMOP7 and LIRCMOP8. To be specific, no significant difference exists between the IGD results of MOEA/D-DCH and MOEA/D-PPS on LIRCMOP8. Compared with MOEA/D-DCH, MOEA/D-IEpsilon achieves similar results on LIRCMOP3, LIRCMOP4 and LIRCMOP14 for IGD metric. The other three algorithms show poor performance on all the fourteen LIRCMOPs.

Under the HV metric, according to the highlighted results in Table 3, MOEA/D-DCH achieves the best results in all LIRCMOP test functions except LIRCMOP7 and LIRCMOP8. MOEA/D-PPS gets the best results on LIRCMOP7 and LIRCMOP8. MOEA/D-PPS achieves similar results on LIRCMOP5 and LIRCMOP6 compared to MOEA/D-DCH. MOEA/D-IEpsilon gets a similar result as MOEA/D-DCH on LIRCMOP13. The other three algorithms still show poor performance on all LIRCMOP test functions. It is important to note that MOEA/D-PPS cannot achieve feasible solutions at every run on LIRCMOP1, LIRCMOP2 and LIRCMOP3. The reason behind the failure is that MOEA/D-PPS requires too many function evaluations to arrive in the unconstrained PF in the push stage, and it is hard to obtain feasible solutions in the pull stage. In conclusion, MOEA/D-DCH performs the best on IGD and HV metrics for most of the LIRCMOP test functions.

Figs. 3–5 further verify the above results. Fig. 3 plots the non-dominated solutions obtained by MOEA/D-DCH and the five peer algorithms on LIRCMOP5. The PF of LIRCMOP5 is same as the unconstrained PF. MOEA/D-DCH and MOEA/D-PPS can achieve almost the whole PF. The other four algorithms get trapped into local optima. Fig. 4 describes the non-dominated solutions achieved by the algorithms on LIRCMOP8. The PF of LIRCMOP8 is located in the constrained boundary. MOEA/D-DCH, MOEA/D-PPS and MOEA/D-IEpsilon can achieve most parts of PF. The other three algorithms cannot achieve the Pareto optimal solutions. The result of MOEA/D-DCH turns out to get the most similarity with PF compared to the other five algorithms. Fig. 5 describes the non-dominated solutions achieved by the algorithms on LIRCMOP12. The PF of LIRCMOP12 is composed of eight points. As Fig. 5 shows, MOEA/D-DCH achieves all the eight points of PF. MOEA/D-PPS cannot achieve the optimum in some points. Compared with MOEA/D-DCH and MOEA/D-PPS, the other algorithms cannot get all the eight points. MOEA/D-IEpsilon achieves seven points while MOEA/D-SR achieves five points. MOEA/D-CDP obtains four points. NSGAIII only obtains two points. Totally, our proposed algorithm MOEA/D-DCH offers obvious advantages over the majority of LIRCMOP test problems compared with the peer algorithms.

4.5. Experimental results on CF test problems

The statistical results of the IGD values and HV values achieved by MOEA/D-DCH and the other five algorithms on seven CF test problems are summarized in Tables 4 and 5 in which the best results are highlighted in bold. As Table 4 shows, MOEA/D-DCH achieves the best mean IGD results on all the seven CF test problems except CF1 and CF3. For CF1, MOEA/D-IEpsilon obtains the best mean result followed by MOEA/D-PPS and MOEA/D-DCH. MOEA/D-DCH also shows similar result to MOEA/D-PPS on CF1.

Table 2
Comparison results on LIRCMOP1-14 test problems for MOEA/D-DCH and the other five CMOEAs (mean and standard deviation IGD indicator).

Problem	MOEA/D-DCH	MOEA/D-PPS	MOEA/D-IEpsilon	MOEA/D-CDP	MOEA/D-SR	NSGAIII
LIRCMOP1	**6.9386e−3 (1.40e−3)**	NaN (NaN)	1.5746e−2 (4.41e−3) −	1.6768e−1 (5.00e−2) −	3.3918e−1 (1.04e−2) −	2.5873e−1 (2.73e−2) −
LIRCMOP2	**5.2017e−3 (4.71e−4)**	NaN (NaN)	1.1151e−2 (3.44e−3) −	1.7478e−1 (2.98e−2) −	4.2591e−1 (2.07e−1) −	2.3235e−1 (2.08e−2) −
LIRCMOP3	**1.0284e−2 (5.69e−3)**	NaN (NaN)	1.6736e−2 (8.51e−3) ≈	2.5649e−1 (4.25e−2) −	4.1388e−1 (2.21e−1) −	2.7314e−1 (2.00e−2) −
LIRCMOP4	**6.9687e−3 (3.00e−3)**	2.3036e−2 (9.81e−3) −	7.8355e−3 (4.22e−3) ≈	2.2888e−1 (3.62e−2) −	6.2015e−1 (6.97e−2) −	2.5074e−1 (2.99e−2) −
LIRCMOP5	2.1046e−3 (9.37e−5)	**1.8441e−3 (1.27e−4)** +	8.8084e−1 (5.34e−1) −	1.1216e+0 (3.06e−1) −	1.2068e+0 (1.71e−2) −	1.2286e+0 (4.67e−3) −
LIRCMOP6	2.1587e−3 (2.30e−4)	**1.7322e−3 (9.25e−5)** +	9.5451e−1 (5.69e−1) −	1.3453e+0 (4.99e−4) −	1.1978e+0 (4.18e−1) −	1.3449e+0 (5.10e−5) −
LIRCMOP7	3.1984e−2 (3.92e−2)	**1,3632e−2 (1.74e−2)** +	5.7321e−2 (5.42e−2) −	1.6760e+0 (6.65e−4) −	1.1020e+0 (7.58e−1) −	6.0852e−1 (6.81e−1) −
LIRCMOP8	2.8412e−2 (5.01e−2)	**6.2446e−3 (7.10e−3)** ≈	1.9221e−1 (4.14e−1) −	1.5833e+0 (3.60e−1) −	1.5989e+0 (3.48e−1) −	1.1090e+0 (7.19e−1) −
LIRCMOP9	**2.1415e−3 (1.64e−4)**	1.3194e−1 (1.72e−1) −	1.1579e−1 (1.21e−1) −	8.4331e−1 (2.15e−1) −	7.8056e−1 (1.60e−1) −	1.4876e+0 (3.17e−1) −
LIRCMOP10	**2.1862e−3 (1.63e−4)**	3.0629e−2 (7.43e−2) −	3.2448e−3 (7.37e−4) −	3.9743e−1 (5.08e−2) −	2.5554e−1 (1.03e−1) −	9.5287e−1 (5.85e−2) −
LIRCMOP11	**2.9297e−3 (3.14e−4)**	1.8847e−2 (4.27e−2) −	7.0833e−2 (4.30e−2) −	4.1556e−1 (1.27e−1) −	4.3306e−1 (9.56e−2) −	8.4625e−1 (8.57e−2) −
LIRCMOP12	**3.3956e−3 (3.38e−4)**	3.5812e−2 (5.58e−2) −	6.3146e−2 (5.98e−2) −	2.8769e−1 (7.68e−2) −	2.5000e−1 (8.85e−2) −	6.6780e−1 (2.08e−1) −
LIRCMOP13	**6.3992e−2 (4.31e−4)**	6.4037e−2 (1.13e−2) ≈	7.2316e−2 (2.17e−2) −	1.0691e+0 (4.41e−1) −	1.1983e+0 (2.90e−1) −	1.3005e+0 (3.28e−4) −
LIRCMOP14	**6.4809e−2 (1.10e−3)**	6.5539e−2 (4.24e−3) ≈	6.5025e−2 (1.73e−3) ≈	9.2696e−1 (5.32e−1) −	1.0910e+0 (4.10e−1) −	1.2565e+0 (2.12e−4) −
+/−/≈		3/5/3	0/11/3	0/14/0	0/14/0	0/14/0

The best results are highlighted.
+, − and ≈ denote MOEA/D-DCH displays significantly worse performance than the compared algorithms, better performance than the compared algorithms, and no significant difference between the compared algorithms, respectively. NaN means the targeted algorithm cannot achieve feasible solutions every run time.

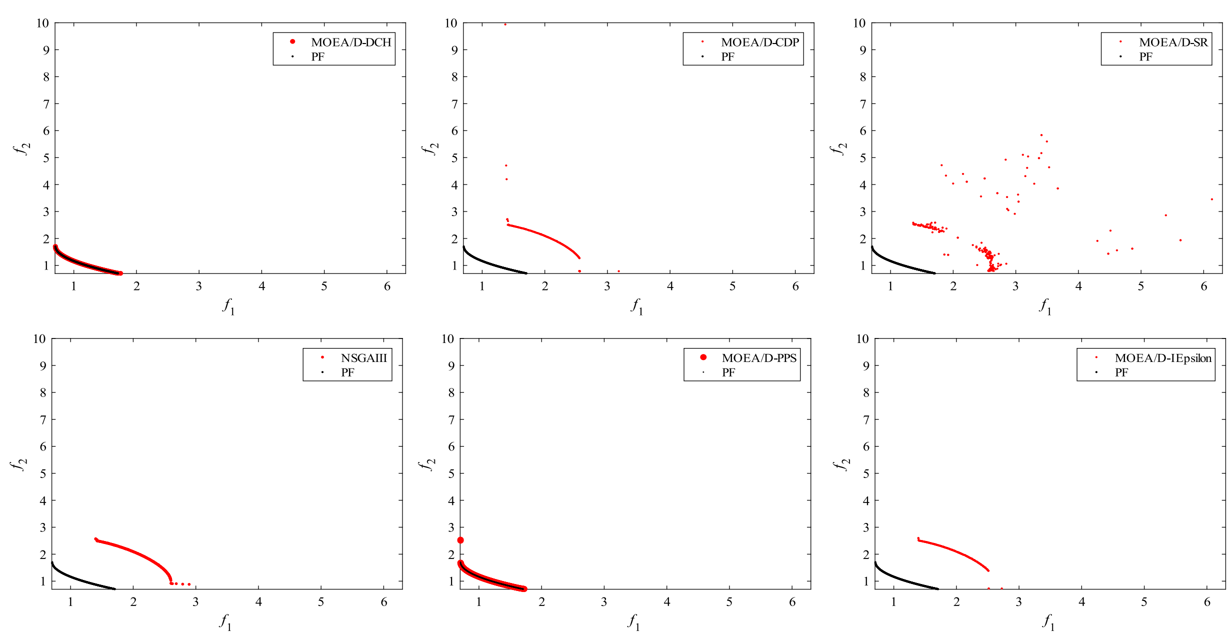

Fig. 3. Non-dominated solutions achieved by the six CMOEAs on LIRCMOP5.

For CF3, NSGAIII achieves the best mean result. MOEA/D-DCH shows similar result to NSGAIII on CF3 as well. For the HV metric, MOEA/D-DCH achieves the best mean results on all the seven CF test functions except CF1 and CF3 as Table 5 shows. MOEA/D-DCH gets the second best mean results on CF1 and CF3. For CF1, MOEA/D-IEpsilon obtains the best mean HV result. NSGAIII realizes the best mean HV result on CF3. Moreover, there are no significant difference between the HV results of NSGAIII and MOEA/D-DCH on CF3. Fig. 6 shows the box plots of IGD value on CF1–7 test problems from the six CMOEAs in 30 independent runs

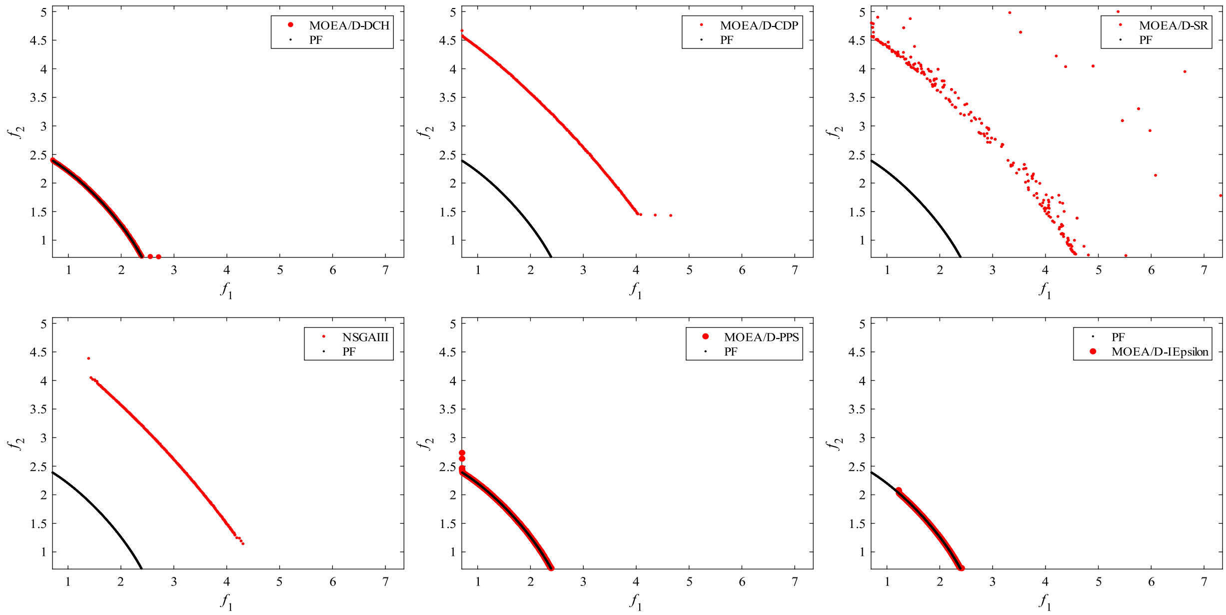

Fig. 4. Non-dominated solutions achieved by the six CMOEAs on LIRCMOP8.

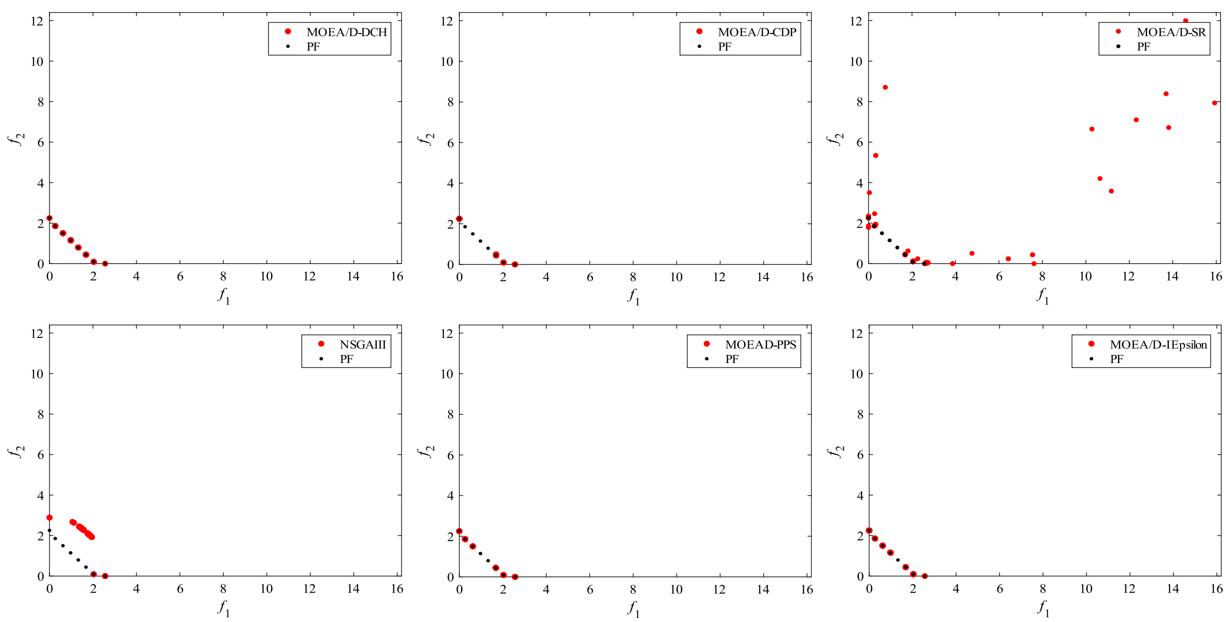

Fig. 5. Non-dominated solutions achieved by the six CMOEAs on LIRCMOP12.

to further demonstrate the results. From Fig. 6, we can observe that MOEA/D-DCH has a superiority for the IGD statistic followed by MOEA/D-CDP. For instance, MOEA/D-DCH shows lower IGD statistical results on most of the seven CF test problems. In particular, for CF2, CF4, CF6, and CF7, other algorithms performs worse statistical characteristics. All the results above mentioned indicate that our proposed algorithm (i.e., MOEA/D-DCH) also show better performance on most of the seven CF test problems.

4.6. Effectiveness of the improved epsilon CHT

To examine the effectiveness of our improved epsilon CHT, the improved epsilon CHT is compared with the traditional epsilon CHT [19] and other two representative CHTs, namely CDP and SR. We replace the improved epsilon CHT in MOEA/D-DCH with the traditional epsilon CHT, CDP, and SR, respectively. Therefore, three variants of MOEA/D-DCH, namely MOEA/D-DCH-CDP, MOEA/D-DCH-SR, and MOEA/D-DCH-Eps are obtained. All the variant algorithms are tested on fourteen LIRCMOP test problems. The IGD metric is used to compare the algorithm performance. After 30 independent runs for each test functions, the results of our algorithm MOEA/D-DCH are compared against MOEA/D-DCH-CDP, MOEA/D-DCH-SR, and MOEA/D-DCH-Eps and expressed in Table 6.

From Table 6, one can observe that MOEA/D-DCH obtains the best mean results in eight out of the fourteen LIRCMOP test problems. Besides, there are no other algorithms that perform significantly better than MOEA/D-DCH for each LIRCMOP test problem. In general, the results verify that the proposed epsilon CHT is better on most of the LIRCMOP test functions compared to the other

Table 3
Performance of MOEA/D-DCH and the other five CMOEAs on LIRCMOP1-14 test problems (mean and standard deviation of HV indicator).

Problem	MOEA/D-DCH	MOEA/D-PPS	MOEA/D-IEpsilon	MOEA/D-CDP	MOEA/D-SR	NSGAIII
LIRCMOP1	**6.4695e−1 (6.81e−4)**	NaN (NaN)	6.3647e−1 (3.72e−3) −	4.2054e−1 (5.07e−2) −	3.2926e−1 (1.20e−2) −	3.3765e−1 (2.53e−2) −
LIRCMOP2	**9.8015e−1 (7.70e−4)**	NaN (NaN)	9.7168e−1 (4.62e−3) −	7.1984e−1 (4.34e−2) −	4.7896e−1 (1.57e−1) −	6.5032e−1 (2.57e−2) −
LIRCMOP3	**5.3735e−1 (3.03e−3)**	NaN (NaN)	5.2981e−1 (5.64e−3) −	3.0026e−1 (3.22e−2) −	2.5088e−1 (7.30e−2) −	2.8370e−1 (1.63e−2) −
LIRCMOP4	**7.7236e−1 (2.28e−3)**	7.7136e−1 (1.07e−2) −	7.6937e−1 (3.60e−3) −	5.3003e−1 (3.56e−2) −	2.6617e−1 (6.91e−2) −	5.1000e−1 (2.93e−2) −
LIRCMOP5	**1.0335e+0 (2.44e−4)**	1.0332e+0 (3.74e−4) ≈	2.7183e−1 (4.59e−1) −	6.7020e−2 (2.60e−1) −	0.0000e+0 (0.00e+0) −	0.0000e+0 (0.00e+0) −
LIRCMOP6	**7.0022e−1 (2.59e−4)**	7.0020e−1 (2.62e−1) ≈	1.5840e−1 (2.53e−1) −	0.0000e+0 (0.00e+0) −	6.4777e−2 (1.89e−1) −	0.0000e+0 (0.00e+0) −
LIRCMOP7	1.9579e+0 (9.82e−2)	**1.9903e+0 (4.85e−2)** ≈	1.8849e+0 (1.29e−1) −	0.0000e+0 (0.00e+0) −	6.3502e−1 (8.36e−1) −	1.1550e+0 (7.55e−1) −
LIRCMOP8	1.9766e+0 (1.19e−1)	**2.0237e+0 (2.39e−2)** ≈	1.7301e+0 (5.04e−1) −	1.0233e−1 (3.96e−1) −	9.5584e−2 (3.70e−1) −	5.9204e−1 (7.53e−1) −
LIRCMOP9	**2.7794e+0 (1.72e−2)**	2.5836e+0 (2.51e−1) −	2.6091e+0 (1.79e−1) −	7.6351e−1 (5.32e−1) −	8.1589e−1 (5.36e−1) −	1.4458e−1 (2.56e−1) −
LIRCMOP10	**2.5554e+0 (3.44e−4)**	2.5074e+0 (1.24e−1) −	2.5523e+0 (1.83e−3) −	1.8002e+0 (1.11e−1) −	2.1008e+0 (2.18e−1) −	2.4747e−1 (6.36e−2) −
LIRCMOP11	**3.4318e+0 (4.28e−2)**	3.4026e+0 (1.26e−1) −	3.2605e+0 (1.41e−1) −	2.1517e+0 (4.71e−1) −	2.1421e+0 (4.61e−1) −	9.3393e−1 (1.32e−1) −
LIRCMOP12	**4.3526e+0 (5.09e−4)**	4.2520e+0 (1.70e−1) ≈	4.2007e+0 (1.60e−1) −	3.4010e+0 (2.90e−1) −	3.5066e+0 (3.04e−1) −	2.0755e+0 (6.84e−1) −
LIRCMOP13	**3.7087e+0 (5.89e−3)**	3.6976e+0 (8.33e−3) ≈	3.6653e+0 (1.50e−1) ≈	5.3994e−1 (1.12e+0) −	2.2886e−1 (5.20e−1) −	3.2636e−3 (3.80e−4) −
LIRCMOP14	**4.0382e+0 (7.26e−3)**	4.0221e+0 (1.17e−2) −	4.0236e+0 (1.34e−2) −	1.0731e+0 (1.76e+0) −	4.6623e−1 (1.06e+0) −	1.0734e−2 (6.06e−4) −
+/−/≈		0/5/6	0/13/1	0/14/0	0/14/0	0/14/0

The best results are highlighted.
+, − and ≈ denote MOEA/D-DCH displays significantly worse performance than the compared algorithms, better performance than the compared algorithms, and no significant difference between the compared algorithms, respectively. NaN means the targeted algorithm cannot achieve feasible solutions every run time.

Table 4
Comparison results on CF1–7 test problems for MOEA/D-DCH and the other five CMOEAs (mean and standard deviation IGD indicator).

Problem	MOEA/D-DCH	MOEA/D-PPS	MOEA/D-IEpsilon	MOEA/D-CDP	MOEA/D-SR	NSGAIII
CF1	2.5865e−3 (2.42e−4)	2.2443e−3 (9.04e−4) ≈	**3.8565e−4 (4.53e−4)** +	3.5123e−3 (1.09e−3) −	6.0029e−3 (2.68e−3) −	2.0031e−2 (1.94e−3) −
CF2	**4.0829e−3 (1.40e−3)**	4.6121e−2 (3.63e−2) −	8.4882e−2 (1.81e−2) −	4.8771e−3 (7.39e−4) −	4.7701e−3 (4.36e−4) −	5.4000e−2 (1.93e−2) −
CF3	2.4770e−1 (1.74e−1)	3.2282e−1 (9.58e−2) ≈	5.0374e−1 (3.11e−1) −	2.6498e−1 (1.30e−1) ≈	3.0936e−1 (1.49e−1) −	**2.2617e−1 (6.99e−2)** ≈
CF4	**6.3582e−2 (4.96e−2)**	7.0382e−2 (6.21e−2) ≈	1.2312e−1 (9.76e−2) −	9.0512e−2 (7.24e−2) ≈	7.3935e−2 (6.03e−2) ≈	1.0542e−1 (4.21e−2) −
CF5	**2.3971e−1 (6.89e−2)**	2.8343e−1 (7.74e−2) −	2.9811e−1 (4.22e−2) −	2.6957e−1 (1.04e−1) ≈	2.5698e−1 (9.22e−2) ≈	2.5049e−1 (2.08e−1) ≈
CF6	**4.5960e−2 (2.58e−2)**	1.4122e−1 (7.45e−2) −	2.5780e−1 (2.00e−3) −	6.8201e−2 (3.36e−2) ≈	8.4932e−2 (3.92e−2) −	8.3748e−2 (4.31e−2) −
CF7	**2.1580e−1 (6.94e−2)**	2.6302e−1 (9.01e−2) −	3.063e−1 (6.99e−2) −	3.1838e−1 (1.53e−1) −	3.1904e−1 (1.68e−1) −	3.4501e−1 (1.65e−1) −
+/−/≈		0/4/3	1/6/0	0/3/4	0/5/2	0/5/2

The best results are highlighted.
+, − and ≈ denote MOEA/D-DCH displays significantly worse performance than the compared algorithms, better performance than the compared algorithms, and no significant difference between the compared algorithms, respectively.

three CHTs. To be specific, MOEA/D-DCH-CDP achieves poorer results on LIRCMOP2, LIRCMOP3 and LIRCMOP4 than MOEA/D-DCH; MOEA/D-DCH-SR experiences a worse outcome on LIRCMOP1, LIRCMOP2, LIRCMOP3, LIRCMOP4 and LIRCMOP11 compared to MOEA/D-DCH; MOEA/D-DCH-Eps performs significantly worse on LIRCMOP1, LIRCMOP2, LIRCMOP3 and LIRCMOP11 than MOEA/D-DCH. There are no significant difference between MOEA/D-DCH and the three variants of MOEA/D-DCH on CMOPs whose PF is blocked by infeasible region such as LIRCMOPs from LIRCMOP5 to LIRCMOP10. The results further demonstrate that with the ability of keeping the diversity of the population in constrained search, the proposed epsilon CHT is especially better than the other three CHTs for CMOPs with extremely small feasible region in objective space such as LIRCMOP2 and LIRCMOP3.

Table 5
Performance of MOEA/D-DCH and the other five CMOEAs on CF1-7 test problems (mean and standard deviation of HV indicator).

Problem	MOEA/D-DCH	MOEA/D-PPS	MOEA/D-IEpsilon	MOEA/D-CDP	MOEA/D-SR	NSGAIII
CF1	6.8114e−1 (4.48e−4)	6.8110e−1 (1.34e−3) ≈	**7.0484e−1 (1.78e−4)** +	6.7981e−1 (1.62e−3) −	6.7665e−1 (3.28e−3) −	6.5451e−1 (2.85-3) −
CF2	**8.1776e−1 (2.27e−3)**	8.1591e−1 (2.50e−3) −	7.8062e−1 (1.20e−2) −	8.1661e−1 (2.60e−3) ≈	8.1693e−1 (2.77e−3) ≈	7.2930e−1 (3.62e−2) −
CF3	2.2873e−1 (6.19e−2)	2.1170e−1 (4.93e−2) ≈	1.5531e−1 (6.23e−2) −	2.1800e−1 (5.31e−2) ≈	1.9556e−1 (6.85e−2) −	**2.3180e−1 (5.58e−2)** ≈
CF4	**5.5857e−1 (4.89e−2)**	5.5361e−1 (6.03e−2) ≈	5.2002e−1 (1.28e−1) ≈	5.3268e−1 (6.23e−2) ≈	5.4564e−1 (6.59e−2) ≈	4.9777e−1 (4.16e−2) −
CF5	**3.8880e−1 (5.51e−2)**	3.6202e−1 (6.15e−2) −	2.9052e−1 (5.97e−2) −	3.7721e−1 (7.15e−2) ≈	3.8591e−1 (6.23e−2) ≈	3.7633e−1 (8.48e−2) ≈
CF6	**7.9779e−1 (2.16e−2)**	6.6703e−1 (8.55e−2) −	4.4543e−1 (5.00e−3) −	7.7959e−1 (1.83e−2) −	7.7390e−1 (1.53e−2) −	7.5776e−1 (3.06e−2) −
CF7	**5.6870e−1 (7.26e−2)**	4.7402e−1 (1.27e−1) −	3.0631e−1 (6.99e−2) −	5.0091e−1 (1.18e−1) −	5.0060e−1 (1.29e−1) −	5.0067e−1 (1.46e−1) −
+/−/≈		0/4/3	1/5/1	0/3/4	0/4/3	0/5/2

The best results are highlighted.
+, − and ≈ denote MOEA/D-DCH displays significantly worse performance than the compared algorithms, better performance than the compared algorithms, and no significant difference between the compared algorithms, respectively.

Table 6
Performance of MOEA/D-DCH and its variants on LIRCMOP1-14 (mean IGD value).

Problem	MOEA/D-DCH	MOEA/D-DCH-Eps	MOEA/D-DCH-CDP	MOEA/D-DCH-SR
LIRCMOP1	**6.9386e−3**	9.0034e−3 −	8.3385e−3 ≈	1.1822e−2 −
LIRCMOP2	**5.2017e−3**	6.5065e−3 −	7.9157e−3 −	8.2484e−3 −
LIRCMOP3	**1.0284e−2**	1.6537e−2 −	1.1373e−1 −	9.5725e−2 −
LIRCMOP4	**6.9687e−3**	9.2568e−3 ≈	1.5241e−1 −	5.0134e−2 −
LIRCMOP5	**2.1046e−3**	2.1638e−3 ≈	2.1116e−3 ≈	2.1483e−3 ≈
LIRCMOP6	2.1587e−3	2.1114e−3 ≈	**2.0861e−3** ≈	2.2086e−3 ≈
LIRCMOP7	3.1984e−2	2.6947e−2 ≈	2.3681e−2 ≈	**1.2462e−2** ≈
LIRCMOP8	2.8412e−2	2.2858e−2 ≈	1.9678e−2 ≈	**1.5852e−2** ≈
LIRCMOP9	**2.1415e−3**	2.3590e−3 ≈	2.2419e−3 ≈	2.2071e−3 ≈
LIRCMOP10	2.1862e−3	2.1886e−3 ≈	**2.0992e−3** ≈	2.1067e−3 ≈
LIRCMOP11	**2.9297e−3**	1.2338e−2 −	7.6484e−3 ≈	1.2315e−2 −
LIRCMOP12	3.3956e−3	**3.3528e−3** ≈	3.3712e−3 ≈	3.5928e−3 ≈
LIRCMOP13	**6.3992e−2**	6.4726e−2 ≈	6.4049e−2 ≈	6.4015e−2 ≈
LIRCMOP14	6.4809e−2	**6.4168e−2** ≈	6.4583e−2 ≈	6.4283e−2 ≈
+/−/≈		0/4/10	0/3/11	0/5/9

The best results are highlighted.
+, − and ≈ denote MOEA/D-DCH displays significantly worse performance than the compared algorithms, better performance than the compared algorithms, and no significant difference between the compared algorithms, respectively.

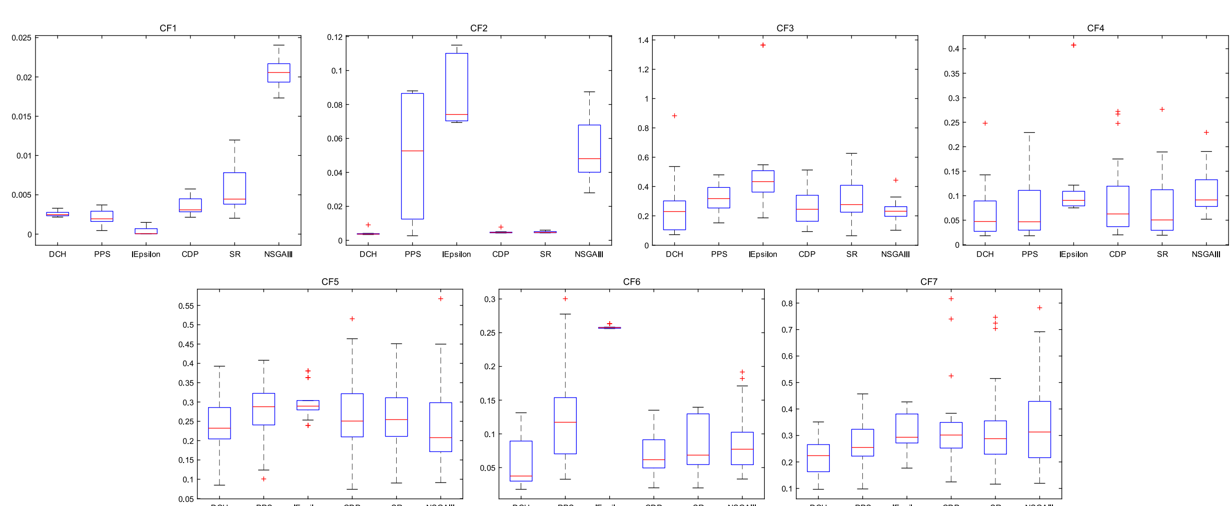

Fig. 6. Box plots of IGD value on CF1–7 test problems from the six CMOEAs in 30 independent runs. In each subplot, DCH, PPS, IEpsilon, CDP and SR represent MOEA/D-CDH, MOEA/D-PPS, MOEA/D-IEpsilon, MOEA/D-CDP and MOEA/D-SR.

Table 7
Performance of MOEA/D-DCH, MOEA/D-DCH with constant DE parameters and MOEA/D-DCH with GA operator on LIRCMOP1-14 (mean and standard deviation of HV value).

Problem	MOEA/D-DCH	F = 0.1	F = 0.3	F = 0.5	F = 0.7	F = 0.9	SBX
LIRCMOP1	**6.4401e−1 (1.27e−3)**	6.0318e−1 (2.48e−2) −	5.5076e−1 (4.07e−2) −	5.3705e−1 (4.55e−2) −	5.3259e−1 (2.47e−2) −	5.1671e−1 (3.29e−2) −	5.1024e−1 (3.85e−2) −
LIRCMOP2	**9.7742e−1 (9.62e−4)**	9.3235e−1 (2.07e−2) −	9.1906e−1 (1.76e−2) −	8.8427e−1 (2.82e−2) −	8.3447e−1 (3.79e−2) −	8.7435e−1 (2.51e−2) −	7.9875e−1 (2.49e−2) −
LIRCMOP3	**5.3598e−1 (2.91e−3)**	3.6816e−1 (2.74e−2) −	4.5910e−1 (2.71e−2) −	4.4540e−1 (2.60e−2) −	4.1684e−1 (2.15e−2) −	4.1967e−1 (3.14e−2) −	3.7788e−1 (3.21e−2) −
LIRCMOP4	**7.6769e−1 (6.93e−3)**	5.7919e−1 (3.31e−2) −	7.0105e−1 (4.45e−2) −	6.8203e−1 (1.99e−2) −	6.6445e−1 (2.83e−2) −	6.6101e−1 (3.52e−2) −	5.7441e−1 (3.88e−2) −
LIRCMOP5	**1.0335e+0 (2.35e−4)**	4.1586e−1 (4.64e−1) −	1.0318e+0 (4.08e−3) −	8.9493e−1 (3.63e−1) ≈	8.9293e−1 (3.63e−1) −	9.9303e−1 (2.17e−2) −	5.1294e−1 (6.27e−2) −
LIRCMOP6	7.0022e−1 (2.42e−4)	0.0000e+0 (0.00e+0) −	6.9804e−1 (6.45e−3) −	**7.0033e−1 (3.03e−4)** ≈	6.9717e−1 (2.85e−3) −	5.1403e−1 (1.13e−1) −	3.7006e−1 (3.65e−2) −
LIRCMOP7	1.9497e+0 (1.14e−1)	1.5813e+0 (4.39e−1) −	1.6952e+0 (5.10e−2) −	1.8253e+0 (1.67e−1) −	**1.9724e+0 (7.60e−2)** ≈	1.8068e+0 (5.02e−1) ≈	1.6822e+0 (5.44e−2) −
LIRCMOP8	**1.9813e+0 (1.06e−1)**	1.0783e+0 (7.90e−1) −	1.7700e+0 (1.67e−1) −	1.9066e+0 (1.86e−1) ≈	1.8491e+0 (2.31e−1) −	1.5544e+0 (1.59e−1) −	1.5944e+0 (8.49e−1) −
LIRCMOP9	**2.7583e+0 (2.49e−2)**	2.4107e+0 (1.80e−1) −	2.6749e+0 (1.53e−1) ≈	2.6618e+0 (8.77e−2) −	2.4722e+0 (2.65e−1) −	1.9506e+0 (1.27e−1) −	2.5924e+0 (6.70e−2) −
LIRCMOP10	**2.5539e+0 (4.62e−4)**	2.4654e+0 (2.15e−1) −	2.5284e+0 (9.05e−2) −	2.5510e+0 (3.42e−3) −	2.4881e+0 (8.49e−2) −	2.2043e+0 (3.49e−2) −	2.5312e+0 (7.80e−3) −
LIRCMOP11	**3.4318e+0 (4.28e−2)**	3.3052e+0 (2.69e−1) −	3.3661e+0 (1.50e−1) −	3.2486e+0 (2.22e−1) −	2.9014e+0 (1.73e−1) −	2.7068e+0 (2.04e−1) −	3.4087e+0 (5.55e−2) −
LIRCMOP12	**4.3027e+0 (1.32e−1)**	4.1085e+0 (2.05e−1) −	4.0479e+0 (1.59e−1) −	3.9958e+0 (7.90e−2) −	3.8960e+0 (1.51e−1) −	3.7621e+0 (1.91e−1) −	4.1064e+0 (8.55e−2) −
LIRCMOP13	3.7087e+0 (5.89e−3)	3.6483e+0 (1.11e−2) −	3.6931e+0 (7.47e−3) −	3.7036e+0 (5.64e−3) −	3.6794e+0 (9.85e−3) −	2.7253e+0 (8.45e−1) −	**3.7569e+0 (1.04e−2)** +
LIRCMOP14	4.0382e+0 (7.26e−3)	4.0177e+0 (7.69e−3) −	4.0335e+0 (8.09e−3) ≈	4.0300e+0 (6.95e−3) ≈	4.0197e+0 (7.35e−3) −	3.1595e+0 (9.35e−1) −	**4.0403e+0 (1.27e−2)** +
+/−/≈		0/14/0	0/11/3	0/10/4	0/13/1	0/13/1	2/12/0

The best results are highlighted.
+, − and ≈ denote MOEA/D-DCH displays significantly worse performance than the compared algorithms, better performance than the compared algorithms, and no significant difference between the compared algorithms, respectively.

4.7. Effectiveness of the parameter settings of DE

To test the effectiveness of dynamic adjustment DE parameters, the scale coefficient F of DE is set to be a constant value as comparison. The values of F are set to 0.1, 0.3, 0.5, 0.7 and 0.9, respectively in our comparison algorithms. Furthermore, MOEA/D-DCH with simulated binary crossover (SBX) operator is also used as a comparison algorithm. All the comparison algorithms are tested on the LIRCMOP test problems. The HV metric is used to compare the algorithm performance. After 30 independent runs, the results of HV metric on fourteen LIRCMOP test functions are shown in Table 7.

As Table 7 shows, MOEA/D-DCH with dynamic adjustment strategy of DE parameters obtains the best performance on all the fourteen test functions except LIRCMOP6, LIRCMOP7, LIRCMOP13 and LIRCMOP14 when compared to MOEA/D-DCH with constant values of F from 0.1 to 0.9 and MOEA/D-DCH with SBX operator. MOEA/D-DCH with $F = 0.5$ and MOEA/D-DCH with $F = 0.7$ achieve the best performance on LIRCMOP6 and LIRCMOP7 respectively. MOEA/D-DCH with SBX operator achieves the best performance on LIRCMOP13 and LIRCMOP14. To be specific, MOEA/D-DCH with the dynamic adjustment strategy of DE parameters presents significantly better results on eight out of the fourteen LIRCMOPs than MOEA/D-DCHs with constant value of F. Compared to MOEA/D-DCH with the dynamic adjustment strategy of DE parameters, MOEA/D-DCH with constant values of F do not show significantly better results on all LIRCMOPs except LIRCMOP13. So it is clear that MOEA/D-DCH with the dynamic DE parameters works relatively better than MOEA/D-DCH with the constant parameters with respect to solving CMOPs. Compared to MOEA/D-DCHs with SBX operator, MOEA/D-DCH with the dynamic adjustment strategy of DE parameters shows significantly better results on all the 14 LIRCMOPs except LIRCMOP13 and LIRCMOP14. It demonstrates that MOEA/D-DCH with the dynamic DE parameters is better than MOEA/D-DCH with SBX operator with respect to solving CMOPs. Compared to the results in Table 3, MOEA/D-DCH with SBX operator shows better mean results on all the 14 LIRCMOPs than three state-of-the-art algorithms including MOEA/D-CDP, MOEA/D-SR and NSGAIII. It indicates that MOEA/D-DCH with SBX operator could also obtain competitive results for CMOPs.

4.8. Sensitivity of MOEA/D-DCH to the expansion factor and the scale parameter

MOEA/D-DCH employs two control parameters: the expansion factor τ and the scale parameter k. To investigate the effects of these two parameters on the performance of MOEA/D-DCH, we have tested in MOEA/D-DCH with 42 combinations of six values of τ (i.e., $\tau = 0.01, 0.02, 0.03, 0.04, 0.05, 0.06$) and seven values of k (i.e., $k = 2, 4, 6, 8, 10, 12, 14$) on LIRCMOP1, LIRCMOP6 and LIRCMOP12. Each combination of τ and k is tested 30 times on each test problem. In our experiments, the other parameters are the same as in Section 4.3. The HV metric is used to compare the algorithm performance.

Fig. 7 shows the contour plots of the median HV-metric values achieved by MOEA/D-DCH with 42 different combinations of τ and k values on LIRCMOP1, LIRCMOP6 and LIRCMOP12.

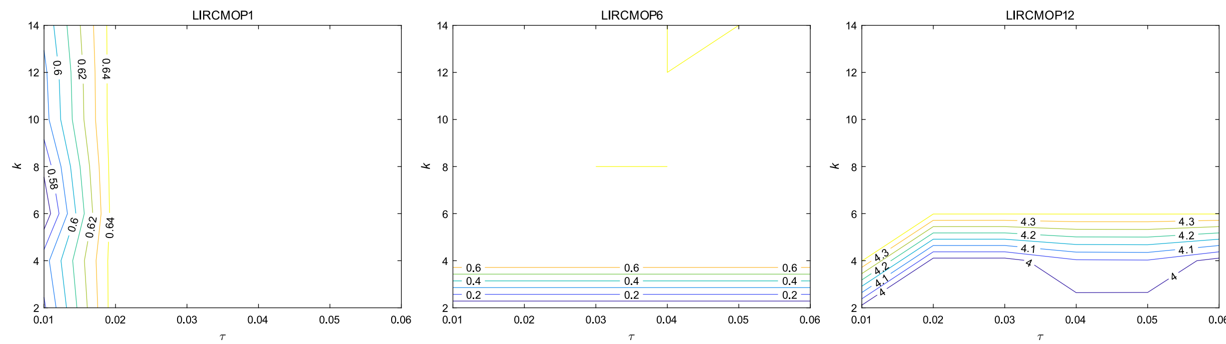

Fig. 7. Contour plots of the median HV-metric values achieved by MOEA/D-DCH with 42 combinations of τ and k values on LIRCMOP1, LIRCMOP6 and LIRCMOP12.

Table 8
Mean and standard deviation of IGD results for NSGAIII, MOEA/D-CDP, MOEA/D-SR, MOEA/D-PPS and MOEA/D-DCH on disk brake problem (the best results are highlighted).

IGD	MOEA/D-DCH	MOEA/D-IEpsilon	MOEA/D-CDP	MOEA/D-SR	NSGAIII	MOEAD-PPS
mean	**2.7237e−2**	7.2269e−2	1.2269e−1	1.2751e−1	7.9200e−2	4.0012e−2
std	**5.36e−4**	3.14e−3	3.20e−2	4.22e−2	6.25e−3	2.53e−3
+/−/≈		−	−	−	−	−

+, − and ≈ denote MOEA/D-DCH displays significantly worse performance than the compared algorithms, better performance than the compared algorithms, and no significant difference between the compared algorithms, respectively.

From Fig. 7, we can see the following results: MOEA/D-DCHs with $\tau = 0.01$ show worse performance than MOEA/D-DCHs with the other τ values on LIRCMOP1; MOEA/D-DCHs with $k = 2$ get worse performance than MOEA/D-DCHs with the other k values on LIRCMOP6; MOEA/D-DCHs with $k = 2$ and $k = 4$ obtain bad performance compared to MOEA/D-DCHs with the other k values on LIRCMOP12. The results demonstrate the parameters τ and k are independent of each other to some extent. Meanwhile, from the results shown in Fig. 7, the combinations of k from 6 to 14 and τ from 0.02 to 0.06 obtain better results on all the three test problems. Their results do not show significant differences with each other on each test problem. Therefore, we recommend that the range of τ is from 0.02 to 0.06 and the range of k is from 6 to 14.

4.9. Application on real-world engineering optimization: disk brake design

In this subsection, a real-world problem is presented to test the practical performance of the proposed algorithm through the comparison between the proposed algorithm and the other five peer algorithms. The disk brake design problem proposed by Ray and Liew [36] is employed as the real-world problem. The objectives are minimizing the stopping time (f_1) and minimizing the disk brake mass (f_2). As can be seen, the disk brake design problem, which is described by the following equation, covers four decision variables and five constraints. The former includes the inner radius and outer radius of the disk (x_1, x_2), the engaging force (x_3), and the number of friction surfaces (x_4). Now the optimization problem is written as follows:

$$\begin{aligned}
&\text{minimize } f_1(x) = 4.9\times 10^{-5}\left(x_2^2 - x_1^2\right)(x_4 - 1)\\
&\text{minimize } f_2(x) = 9.8\times 10^{6}\left(x_2^2 - x_1^2\right)/x_3x_4\left(x_2^3 - x_1^3\right)\\
&s.t.\begin{cases} 20-(x_2-x_1)\le 0\\ 2.5\,(x_4+1)-30\le 0\\ x_3/\pi\left(x_2^2-x_1^2\right)^2-0.4\le 0\\ 2.22\times 10^{-3}x_3\left(x_2^3-x_1^3\right)/\left(x_2^2-x_1^2\right)^2-1\le 0\\ 900-2.66\times 10^{-2}x_3x_4\left(x_2^3-x_1^3\right)/\left(x_2^2-x_1^2\right)\le 0\end{cases}
\end{aligned} \tag{11}$$

where $55 \le x_1 \le 80, 75 \le x_2 \le 110, 1000 \le x_3 \le 3000, 2 \le x_4 \le 20$. For all of the algorithms, the population size N_p is designated to be 200, and each algorithm does not stop until its function evaluation number reaches its terminal value of 50,000. In our experiments, the other parameters are identical to those in Section 4.3. We adopt the IGD metric to evaluate the algorithm performance. The results are summarized in Table 8 after 30 run times.

As Table 8 shows, the result of MOEA/D-DCH is significantly better than the other five algorithms on this real-world problem in terms of IGD metric. MOEA/D-PPS achieves the second best result followed by MOEA/D-IEpsilon and NSGAIII. MOEA/D-CDP and MOEA/D-SR obtain the worst results. In Fig. 8, the population achieved by MOEA/D-DCH and the other algorithms are described in a typical run, respectively. As Fig. 8 shows, MOEA/D-IEpsilon, MOEA/D-CDP, MOEA/D-SR and NSGAIII demonstrate worse performance on distribution compared to MOEA/D-DCH because the individual distribution at ends of their approximate PFs is sparse. Compared to MOEA/D-DCH, some points of MOEA/D-SR and MOEA/D-PPS cannot converge to the PF. Therefore, MOEA/D-DCH achieves the best performance with regard to the convergence and distribution compared to the other peer algorithms.

5. Conclusion

Constrained multi-objective optimization problem (CMOP) is an important research direction of system engineering. Solving CMOPs is full of challenges for evolutionary multi-objective optimization. To this end, this paper develops a practical, general-purpose multi-objective evolutionary algorithm. Specifically, it involves a dynamic constraint-handling mechanism, an individual update strategy based on the best feasible solutions. The dynamic constraint-handling mechanism switches the unconstrained search and constrained search by the proportion of feasible solutions and the generation number, which accelerates the convergence of population. For the constrained search, we develop an improved epsilon CHT to enhance the diversity of the population. Furthermore, we also adopt a dynamic parameter adjustment mechanism of reproduction operator to enhance the local search ability.

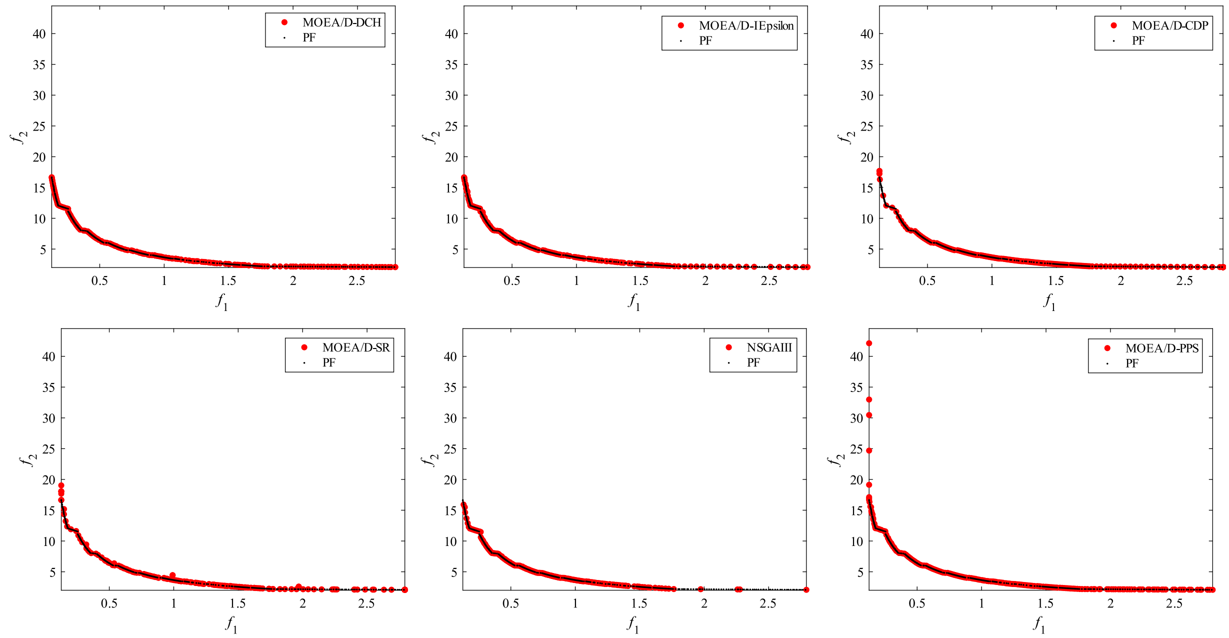

Fig. 8. Population achieved by the six CMOEAs on disk brake problem.

Empirical comparisons have been done by comparing MOEA/D-DCH and the other five state-of-the-art CMOEAs on 21 benchmark test problems. The statistical results indicate that MOEA/D-DCH shows competitive performance on the studied CMOP test problems. MOEA/D-DCH also shows its superiority for solving real-world CMOPs via a real-world test problem. It is also important to emphasize that much work remains to be done in order to enhance the algorithm performance, such as distribution mechanism for the constrained search space. In our future study, we are planning to modify the proposed algorithm to address other real-world problems.

Declaration of competing interest

No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.asoc.2020.106104.

CRediT authorship contribution statement

Yongkuan Yang: Conceptualization, Data curation, Investigation, Methodology, Software, Writing - original draft. **Jianchang Liu:** Funding acquisition, Resources, Project administration, Supervision, Validation. **Shubin Tan:** Formal analysis , Validation , Visualization, Writing - review & editing.

Acknowledgment

This work was supported partly by the National Natural Science Foundation of China under Grant 61703086, and Grant 61773106.

References

[1] H. Wang, L. Jiao, X. Yao, Two arch2: An improved two-archive algorithm for many-objective optimization, IEEE Trans. Evol. Comput. 19 (4) (2015) 524–541, http://dx.doi.org/10.1109/TEVC.2014.2350987.

[2] Y. Wang, H. Li, G.G. Yen, W. Song, MOMMOP: Multiobjective optimization for locating multiple optimal solutions of multimodal optimization problems, IEEE Trans. Cybern. 45 (4) (2015) 830–843, http://dx.doi.org/10.1109/TCYB.2014.2337117.

[3] F. Li, R. Cheng, J. Liu, Y. Jin, A two-stage R2 indicator based evolutionary algorithm for many-objective optimization, Appl. Soft Comput. 67 (2018) 245–260, http://dx.doi.org/10.1016/j.asoc.2018.02.048.

[4] Y. Yang, J. Liu, S. Tan, H. Wang, Application of constrained multi-objective evolutionary algorithm in multi-source compressed-air pipeline optimization problems, IFAC-PapersOnLine 51 (18) (2018) 168–173, http://dx.doi.org/10.1016/j.ifacol.2018.09.294.

[5] A. Ponsich, A.L. Jaimes, C.A.C. Coello, A survey on multiobjective evolutionary algorithms for the solution of the portfolio optimization problem and other finance and economics applications, IEEE Trans. Evol. Comput. 17 (3) (2013) 321–344, http://dx.doi.org/10.1109/TEVC.2012.2196800.

[6] O. Castillo, L. Trujillo, P. Melin, Multiple objective genetic algorithms for path-planning optimization in autonomous mobile robots, Soft Comput. 11 (3) (2007) 269–279, http://dx.doi.org/10.1007/s00500-006-0068-4.

[7] Y. Lin, W. Du, W. Du, Multi-objective differential evolution with dynamic hybrid constraint handling mechanism, Soft Computing (2018) 1–15, http://dx.doi.org/10.1007/s00500-018-3087-z.

[8] F. Qian, B. Xu, R. Qi, H. Tianfield, Self-adaptive differential evolution algorithm with α-constrained-domination principle for constrained multi-objective optimization, Soft Comput. 16 (8) (2012) 1353–1372, http://dx.doi.org/10.1007/s00500-012-0816-6.

[9] E. Mezura-Montes, C.A.C. Coello, Constraint-handling in nature-inspired numerical optimization: Past, present and future, Swarm Evol. Comput. 1 (4) (2011) 173–194, http://dx.doi.org/10.1016/j.swevo.2011.10.001.

[10] Y.G. Woldesenbet, G.G. Yen, B.G. Tessema, Constraint handling in multiobjective evolutionary optimization, IEEE Trans. Evol. Comput. 13 (3) (2009) 514–525, http://dx.doi.org/10.1109/TEVC.2008.2009032.

[11] M.A. Jan, R.A. Khanum, A study of two penalty-parameterless constraint handling techniques in the framework of MOEA/D, Appl. Soft Comput. J. 13 (1) (2013) 128–148, http://dx.doi.org/10.1016/j.asoc.2012.07.027.

[12] T. Ray, H.K. Singh, A. Isaacs, W. Smith, Infeasibility Driven Evolutionary Algorithm for Constrained Optimization, Springer Berlin Heidelberg, 2009, pp. 145–165, http://dx.doi.org/10.1007/978-3-642-00619-7_7.

[13] Z. Fan, W. Li, X. Cai, H. Li, C. Wei, Q. Zhang, K. Deb, E. Goodman, Push and pull search for solving constrained multi-objective optimization problems, Swarm Evol. Comput. 44 (2019) 665–679, http://dx.doi.org/10.1016/j.swevo.2018.08.017.

[14] Z. Liu, Y. Wang, Handling constrained multiobjective optimization problems with constraints in both the decision and objective spaces, IEEE Trans. Evol. Comput. (2019) 1, http://dx.doi.org/10.1109/TEVC.2019.2894743.

[15] C.A. Coello Coello, Constraint-handling techniques used with evolutionary algorithms, in: Proceedings of the 14th Annual Conference Companion on Genetic and Evolutionary Computation, in: GECCO '12, ACM, New York, NY, USA, 2012, pp. 849–872, http://dx.doi.org/10.1145/2330784.2330920.

[16] K. Deb, An efficient constraint handling method for genetic algorithms, Comput. Methods Appl. Mech. Eng. 186 (2) (2000) 311–338, http://dx.doi.org/10.1016/S0045-7825(99)00389-8.

[17] 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, http://dx.doi.org/10.1109/4235.996017.

[18] Z. Fan, H. Li, C. Wei, W. Li, H. Huang, X. Cai, Z. Cai, An improved epsilon constraint handling method embedded in MOEA/d for constrained multi-objective optimization problems, in: 2016 IEEE Symposium Series on Computational Intelligence (SSCI), 2016, pp. 1–8, http://dx.doi.org/10.1109/SSCI.2016.7850224.

[19] T. Takahama, S. Sakai, Efficient constrained optimization by the ε constrained differential evolution with rough approximation using kernel regression, in: IEEE Congress on Evolutionary Computation, 2013, pp. 1334–1341, http://dx.doi.org/10.1109/CEC.2013.6557719.

[20] C. Peng, H.-L. Liu, F. Gu, An evolutionary algorithm with directed weights for constrained multi-objective optimization, Appl. Soft Comput. 60 (2017) 613–622, http://dx.doi.org/10.1016/j.asoc.2017.06.053.

[21] W. Ning, B. Guo, Y. Yan, X. Wu, J. Wu, D. Zhao, Constrained multi-objective optimization using constrained non-dominated sorting combined with an improved hybrid multi-objective evolutionary algorithm, Eng. Optim. 49 (10) (2017) 1645–1664, http://dx.doi.org/10.1080/0305215X.2016.1271661.

[22] B.Y. Qu, P.N. Suganthan, Constrained multi-objective optimization algorithm with an ensemble of constraint handling methods, Eng. Optim. 43 (4) (2011) 403–416, http://dx.doi.org/10.1080/0305215X.2010.493937.

[23] J. Li, Y. Wang, S. Yang, Z. Cai, A comparative study of constraint-handling techniques in evolutionary constrained multiobjective optimization, in: 2016 IEEE Congress on Evolutionary Computation (CEC), 2016, pp. 4175–4182, http://dx.doi.org/10.1109/CEC.2016.7744320.

[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, http://dx.doi.org/10.1109/TEVC.2008.925798.

[25] Q. Zhang, H. Li, MOEA/D: A multiobjective evolutionary algorithm based on decomposition, IEEE Trans. Evol. Comput. 11 (6) (2007) 712–731, http://dx.doi.org/10.1109/TEVC.2007.892759.

[26] Y. Wang, B. Xu, G. Sun, S. Yang, A two-phase differential evolution for uniform designs in constrained experimental domains, IEEE Trans. Evol. Comput. 21 (5) (2017) 665–680, http://dx.doi.org/10.1109/TEVC.2017.2669098.

[27] 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, http://dx.doi.org/10.1023/A:1008202821328.

[28] Y. Yang, J. Liu, S. Tan, H. Wang, A multi-objective differential evolutionary algorithm for constrained multi-objective optimization problems with low feasible ratio, Appl. Soft Comput. 80 (2019) 42–56, http://dx.doi.org/10.1016/j.asoc.2019.02.041.

[29] Y. Wang, Z. Cai, Combining multiobjective optimization with differential evolution to solve constrained optimization problems, IEEE Trans. Evol. Comput. 16 (1) (2012) 117–134, http://dx.doi.org/10.1109/TEVC.2010.2093582.

[30] B. Xu, X. Chen, L. Tao, Differential evolution with adaptive trial vector generation strategy and cluster-replacement-based feasibility rule for constrained optimization, Inform. Sci. 435 (2018) 240–262, http://dx.doi.org/10.1016/j.ins.2018.01.014.

[31] Z. Fan, W. Li, X. Cai, H. Li, C. Wei, Q. Zhang, K. Deb, E. Goodman, Difficulty Adjustable and Scalable Constrained Multi-objective Test Problem Toolkit, Evol. Comput., 0 (ja) 0, 1–28, [in press] Posted Online May 23, 2019. http://dx.doi.org/10.1162/evco_a_00259.

[32] Q. Zhang, A. Zhou, S. Zhao, P.N. Suganthan, W. Liu, S. Tiwari, Multiobjective Optimization Test Instances for the CEC 2009 Special Session and Competition, Technical Report, CES-487, 2008, pp. 1–20.

[33] H. Jain, K. Deb, An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part II: Handling constraints and extending to an adaptive approach, IEEE Trans. Evol. Comput. 18 (4) (2014) 602–622, http://dx.doi.org/10.1109/TEVC.2013.2281534.

[34] Z. Fan, Y. Fang, W. Li, X. Cai, C. Wei, E. Goodman, MOEA/d with angle-based constrained dominance principle for constrained multi-objective optimization problems, Appl. Soft Comput. 74 (2019) 621–633, http://dx.doi.org/10.1016/j.asoc.2018.10.027.

[35] P.A. Bosman, D. Thierens, The balance between proximity and diversity in multiobjective evolutionary algorithms, IEEE Trans. Evol. Comput. 7 (2) (2003) 174–188.

[36] T. Ray, K. Liew, A swarm metaphor for multiobjective design optimization, Eng. Optim. 34 (2) (2002) 141–153, http://dx.doi.org/10.1080/03052150210915.