Information Sciences 578 (2021) 592–614

Contents lists available at ScienceDirect

Information Sciences

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

Two-type weight adjustments in MOEA/D for highly constrained many-objective optimization

Ruwang Jiao [a], Sanyou Zeng [a,*], Changhe Li [b,c,*], Yew-Soon Ong [d]

[a] School of Mechanical Engineering and Electronic Information, China University of Geosciences, Wuhan 430074, China
[b] School of Automation, China University of Geosciences, Wuhan 430074, China
[c] Hubei Key Laboratory of Advanced Control and Intelligent Automation for Complex Systems, China University of Geosciences, Wuhan 430074, China
[d] School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798, Singapore

ARTICLE INFO

Article history:
Received 11 December 2020
Received in revised form 29 June 2021
Accepted 12 July 2021
Available online 15 July 2021

Keywords:
Evolutionary computation
Eonstrained many-objective optimization
Constraint handling
MOEA/D

ABSTRACT

A key issue in evolutionary constrained optimization is how to achieve a balance between feasible and infeasible solutions. The quality of generated solutions in decomposition-based multi-objective evolutionary algorithms (MOEAs) depends strongly on the weights' setting. To fully utilize both the promising feasible and infeasible solutions, this paper proposes two-type weight adjustments based on MOEA/D for solving highly constrained many-objective optimization problems (CMaOPs). During the course of the search, the number of infeasible weights is dynamically reduced, to guide infeasible solutions with better convergence to cross the infeasible barrier, and also to lead infeasible solutions with better diversity to locate multiple feasible subregions. Feasible weights are evenly distributed and keep unchanged throughout the evolution process, which aims to guide the population to search Pareto optimal solutions. The effectiveness of the proposed algorithm is verified by comparing it against six state-of-the-art CMaOEAs on three sets of benchmark problems. Experimental results show that the proposed algorithm outperforms compared algorithms on majority problems, especially on highly constrained optimization problems. Besides, the effectiveness of the proposed algorithm has also been verified on an antenna array synthesis problem.

1. Introduction

Multi-objective optimization problems (MOPs) with more than three objectives, known as many-objective optimization problems (MaOPs), have attracted growing research interest from the evolutionary computation community [39,21]. With the development of multi-objective evolutionary algorithms (MOEAs), a number of many-objective evolutionary algorithms (MaOEAs) have been put forward to search for well-diversified and well-converged Pareto optimal solutions for high dimensional MOPs.

MOEAs can be categorized as Pareto-dominance-based framework, indicator-based framework, and decomposition-based framework. In Pareto-dominance-based algorithms such as NSGA-II [9], the convergence of the algorithm is achieved based on the Pareto-dominance principle. However, as the number of objectives increases, a large number of solutions become non-dominated. These dominance resistance solutions heavily weaken the selection pressure towards the Pareto front

* Corresponding authors.
E-mail address: changhe.lw@gmail.com (C. Li).

https://doi.org/10.1016/j.ins.2021.07.048
0020-0255/

(PF), resulting in the rapid deterioration of the convergence ability of the algorithm. In indicator-based algorithms, a performance indicator such as the hypervolume (HV) [2], is employed to measure the fitness of a solution by estimating its contribution to both the convergence and diversity. However, the computation cost of the HV grows exponentially with the number of objectives increasing. Decomposition-based algorithms decompose a MOP into a set of subproblems and solve them in parallel [15,47,26,37,42]. The MOEA based on decomposition (MOEA/D) [47] is a representative one. In MOEA/D, a number of weights are employed to specify diverse search directions towards multiple subareas of the PF. The population of MOEA/D is supposed to have an extensive spread along the PF due to the weights of a problem are widely distributed. Since decomposition-based algorithms do not utilize Pareto-dominance relation to compare solutions, they are likely to be unaffected by inadequate selection pressure in MaOPs [43]. MOEA/D [47] and MOEA/DD [24] have shown their effectiveness in solving MaOPs.

Up to now, too much attention has been paid to unconstrained MaOPs, there are relatively fewer works that focus on handling constrained MaOPs (CMaOPs). CMaOPs are optimization problems with multiple conflicting optimization objectives subject to several equality and/or inequality constraints, which appear regularly in real-world applications [13,12,38]. A minimization CMaOP can be formulated as:

$$\begin{aligned} &\min && \boldsymbol{F}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \cdots, f_m(\boldsymbol{x})) \\ &\text{subject to}: && g_j(\boldsymbol{x}) \leqslant 0, j = 1, \ldots, q \\ &\text{where} && \boldsymbol{x} = (x_1, \ldots, x_D) \in \Omega \\ & && \Omega = \{\boldsymbol{x} | \boldsymbol{l} \leqslant \boldsymbol{x} \leqslant \boldsymbol{u}\} \\ & && \boldsymbol{l} = (l_1, \ldots, l_D), \boldsymbol{u} = (u_1, \ldots, u_D), \end{aligned} \tag{1}$$

where $\boldsymbol{x}$ is the solution vector of which x_k is within $l_k \leqslant x_k \leqslant u_k$ $(k = 1, \ldots, D)$, $\boldsymbol{F}(\boldsymbol{x})$ is the objective vector that consists of m conflicting objective functions, $g_j(\boldsymbol{x})$ is the j-th inequality constraint. A solution $\boldsymbol{x}$ is called a feasible solution in case $\boldsymbol{g}(\boldsymbol{x}) \leqslant \boldsymbol{0}$; otherwise, it is an infeasible solution. Note that when an equality constraint appears: $h(\boldsymbol{x}) = 0$, it is usually transformed into an inequality constraint by shrinking the constraint boundary: $|h(\boldsymbol{x})| - \xi \leqslant 0$.

CMaOPs are hard to solve since they require a CMaOEA to provide a set of trade-off optimal solutions subject to satisfying all constraints in the high-dimensional objective space. Particularly, some MaOPs may have complex constraints: small feasible regions, or constraint functions with complicated features, such as multimodal, deceptiveness, highly nonlinearity, discontinuity. These problems are more challenging because of the following difficulties:

1. How to find feasible solutions. From the perspective of decision makers, feasible solutions are more important than infeasible solutions. Nevertheless, the constraints change the shape and limit the feasible range of the search space, making it hard to obtain feasible solutions for such problems [41].
2. How to cross multiple disjoint feasible and infeasible regions to approximate the constrained PF. Constraints in the many-objective space incline to divide feasible regions into multiple separate subregions. A CMaOEA is easy to be trapped in some locally feasible or infeasible areas and hard to approach the constrained PF.

The key to overcoming the above two difficulties is to handle the relationship between feasible and infeasible solutions [23,17]. Considering the difficulty in obtaining feasible solutions in the many-objective space, using infeasible solutions with good convergence is beneficial for the population to cross the large infeasible barriers. Besides, utilizing infeasible solutions with good diversity is useful to locate the unexplored feasible regions. More importantly, the Pareto optimal solutions of a CMaOP are usually lying on constraint boundaries. The ideal search should use both the feasible and infeasible solutions to approach these constraint boundaries from both the feasible and infeasible sides of the search space. For decomposition-based MOEAs, the quality of generated feasible and infeasible solutions strongly depends on the distribution of the specified weights.

Therefore, to take full advantages of decomposition-based algorithms to solve highly constrained optimization problems in the many-objective space, this paper proposes a decomposition-based MaOEA with two-type weight vector adjustments (MOEA/D-2WA) to efficiently use both the feasible and infeasible solutions. In MOEA/D-2WA, by transforming the degree of constraint violation as the $(m + 1)$-th objective, two-type weights are constructed: infeasible weights are designed for infeasible solutions, and feasible weights are produced for guiding feasible solutions. The main contribution of this paper is to attempt to solve CMaOPs via decomposition-based MOEAs with the multi-objective constraint-handling technique. To handle complex constraints, the designed two-type weights have distinct aims at different stages: at the early and middle stages, infeasible weights aim to guide the population jump out of the infeasible barrier and local feasible regions, as well as to locate multiple promising subregions where feasible solutions are hard to reach, and together with the feasible weights to search for boundary optima solutions from both the feasible and infeasible sides of the search space. At the later stage, all infeasible weights are gradually removed. The remaining feasible weights are used to guide the population to find a set of feasible Pareto optimal solutions which are uniformly distributed along the PF.

The rest of this paper is organized as follows. Section 2 gives a brief literature review of current CMaOEAs. The details of the proposed algorithm are presented in Section 3. The experiments and comparisons are carried out in Section 4. The discussions, as well as a real-world case study on an antenna array synthesis problem, are provided in Section 5. Finally, Section 6 concludes this paper.

2. Literature review

Constraint-handling techniques play a vital role in CMaOEAs. The current constraint-handling methods based on evolutionary algorithms (EAs) including [30]: feasibility rule, stochastic ranking, ε-constrained method, novel penalty functions, novel special operators, multi-objective approach, and ensemble methods. The crux for these constraint-handling techniques is how to deal with the relationship between objectives and constraints, and how to utilize the information from feasible and infeasible solutions. Based on the utilization level of infeasible solutions, the existing CMaOEAs can be loosely classified into the following two groups.

2.1. Feasibility-driven CMaOEAs

The feasibility-driven methods tend to retain feasible solutions in a population. The constraint dominance principle (CDP) [9] is a representative one which is a combination of feasibility rule and dominance principle for solving CMOPs, and it does not require fine-tuning parameters. It makes constraints take precedence over objectives in dominance relation. To elaborate, for any two solutions, $\boldsymbol{x}_1$ is considered superior to $\boldsymbol{x}_2$ if any one of the following conditions holds:

- $\boldsymbol{x}_1$ is feasible while $\boldsymbol{x}_2$ is not;
- Both of them are infeasible, but the degree of constraint violation of $\boldsymbol{x}_1$ is smaller than the degree of constraint violation of $\boldsymbol{x}_2$;
- Both of them are feasible, but $\boldsymbol{x}_1$ Pareto dominates $\boldsymbol{x}_2$.

C-NSGA-III [16] employs the CDP to compare pairs of solutions, which prefers a feasible solution over an infeasible solution consistently, and an infeasible solution with the smaller constraint violation dominates another infeasible solution with the higher constraint violation. C-MOEA/D [16] adopts the following neighbor solution replacement principle: 1) a feasible solution is superior to an infeasible solution; 2) an infeasible solution with smaller constraint violation is better than an infeasible solution with larger constraint violation; 3) a feasible solution with better scalarizing function value is better than a feasible solution with the worse scalarizing function value.

C-RVEA [4] employs reference vectors to decompose a MaOP into a number of single-objective subproblems. If all solutions are infeasible for a subpopulation, it selects the one with the minimum constraint violation to the next generation; otherwise, the feasible solution with the minimum angle-penalized distance value will be chosen. Similarly, if the number of feasible solutions in the union of parent and offspring populations is larger than the population size *NP*, C-AnD [28] selects the parent population for the next generation from these feasible solutions in terms of their angles; otherwise, the *NP* solutions with smallest constraint violations will be chosen.

An adaptive sorting-based EA (ASEA) [25] was proposed for handling problems with irregular PF. In the environmental selection procedure, if all solutions are infeasible for each subpopulation, the sorting is based on constraint violation values; otherwise, solutions in each subpopulation are ranked on the basis of the convergence, and these solutions with better convergence are further ranked in terms of the diversity.

2.2. Infeasibility-assisted CMaOEAs

Roughly speaking, EAs assisted by infeasible solutions for solving CMOPs and CMaOPs can be classified into the following four categories.

The first category retains infeasible solutions with better diversity. For example, in addition to the feasibility and dominance relationship, the angle-based CDP [10] considers the angle information between two solutions to make infeasible solutions with better diversity to survive. In C-MOEA/DD [24], the survival of infeasible solutions is determined by their constraint violations and distribution. If an infeasible solution has the largest constraint violation but is associated with a sparse subarea, it still has the chance to survive to enhance the population diversity.

The second category preserves infeasible solutions with better convergence. In I-DBEA [1], the ε level for distinguishing the feasible and infeasible solutions is adaptively updated based on the average level of the constraint violation and the feasibility ratio of the population. The dual-grid MOEA/D [14] uses two types of weight grids: one is for feasible solutions and the other is for infeasible solutions. An offspring is generated from its neighborhood in the two grids with the same weight grid. Thus, an infeasible solution with better fitness can be used to help the population to get across the local feasible area. Recently, a problem transformation technique [18,19] was proposed, which transforms a CMaOP into a dynamic CMaOP for handling constraints and optimizing objectives simultaneously, to help the population cross the large and discrete infeasible barriers.

The third category employs an additional population or an archive to coevolution. The cooperative differential evolution algorithm [41] splits the whole population into $m + 1$ subpopulations. The first m subpopulations solve the allocated constrained m single-objective optimization problems, while the $(m + 1)$-th archive population is used to save constrained non-dominated solutions from subpopulations and guide subpopulations to search along the entire PF. The two-archive EA (C-TAEA) [23] keeps two collaborative and complementary archives simultaneously: one is the convergence-oriented archive,

which maintains the convergence and feasibility of the population to drive the population towards the PF; the other is the diversity-oriented one, which aims to provide more diversified information. The coevolutionary constrained multi-objective optimization framework (CCMO) [36] evolves two populations: one is to address the original optimization problem, the other is to deal with a helper problem derived from the original one. Meanwhile, the two populations hold weak cooperation to generate high-quality offsprings.

The last category is based on two-phase optimization. The push and pull search (PPS) [11] divides the whole search into two phases. In the first phase, constraints are ignored. The population is pushed towards the unconstrained PF without considering any constraints. In the second phase, an improved ε constraint-handling method is employed to pull the population to the constrained PF. The cooperative evolutionary framework [31] is based on improved directed weights [32] which consists of two switchable phases: the first phase focus on exploring both feasible regions and the entire space, while the second phase aims to find Pareto optimal solutions. In ToP [27], the first phase tries to search for promising feasible regions by converting a CMOP to a constrained single-objective optimization problem. The second phase can be implemented by any CMOEA to search for final optimal solutions.

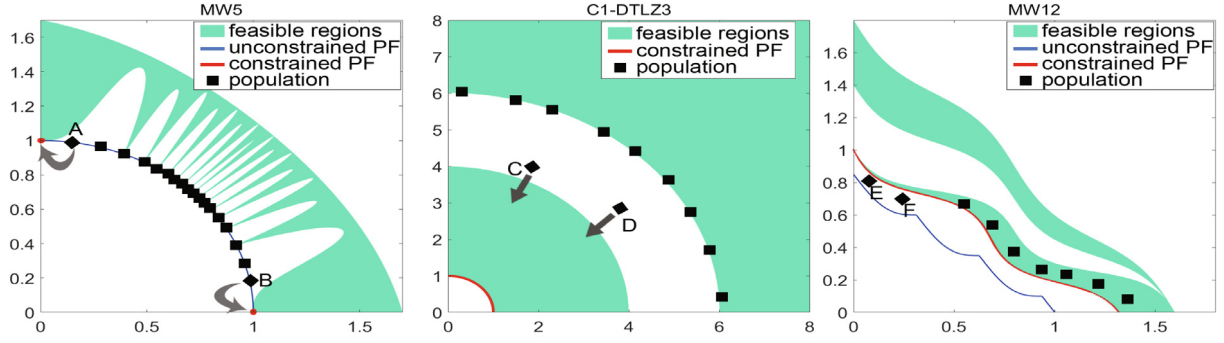
2.3. Discussions and Motivation

From the above reviews, we can find that the feasibility-driven CMaOEAs are appealing for solving CMaOPs due to their promising advantages: they are easy to be implemented and free of set parameters, they are also capable of pushing the population towards the feasible area promptly. However, the degree of constraint violation of a CMaOP does not monotonically decrease when it converges to the feasible area, the constraint-handling techniques adopted in feasibility-driven CMaOEAs prefer feasible solutions and neglect the objective information carried by infeasible solutions, which may lead to premature convergence in solving CMaOPs.

The usage of informative infeasible solutions is advantageous for the search in the many-objective space. On the one hand, infeasible solutions with good diversity are conducive to escape from the locally feasible areas and locate the unexplored feasible areas, i.e., in Fig. 1(a), infeasible solutions A and B are helpful to locate the isolated boundary Pareto optimal solutions. On the other hand, infeasible solutions with good convergence are beneficial to cross the infeasible barriers, i.e., in Fig. 1(b), infeasible solutions C and D can drag the population to pass through the infeasible area to get to the global feasible area. More importantly, the Pareto optimal solutions of a CMaOEA are likely to lie on constraint boundaries [34]. To effectively approach these boundaries, using the information of infeasible solutions (i.e., infeasible solutions E and F in Fig. 1(c)) can get close to the Pareto optimal solutions from both the feasible and infeasible sides of the search space.

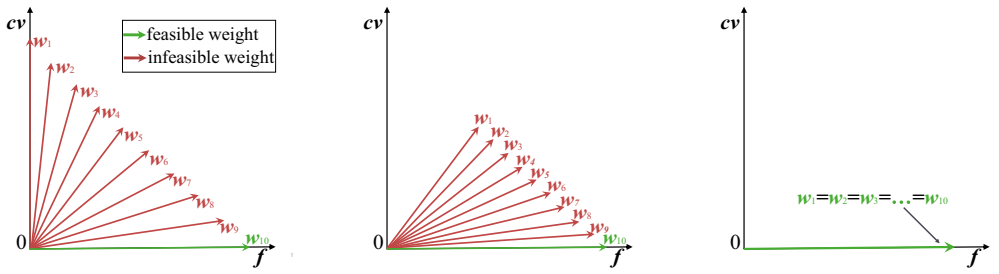
Although infeasibility-assisted CMaOEAs have shown promising in solving CMaOPs, these methods encounter the following major issues: 1) most CMaOEAs only use the single type of infeasible solution, but they seldom consider utilizing the infeasible solutions with good diversity and convergence simultaneously; 2) methods based on dual-population often adopt an auxiliary population or an archive to retain promising infeasible solutions. However, they often take too much unnecessary effort to store and update these infeasible solutions; 3) in methods based on two-phase optimization, the switch moment between the two phases is a crucial issue. Different from the existing approaches of adopting two phases or two populations to distinguish and utilize feasible and infeasible solutions, based on the multi-objective constraint-handling technique, this study proposes a two-type weight adjustment strategy in MOEA/D for utilizing the potential feasible and infeasible solutions simultaneously.

Figs. 2,3 depict the weight adjustment process for a constrained single-objective optimization problem and a constrained bi-objective optimization problem, respectively. They all regard the constraint violation as an objective and set extra weights



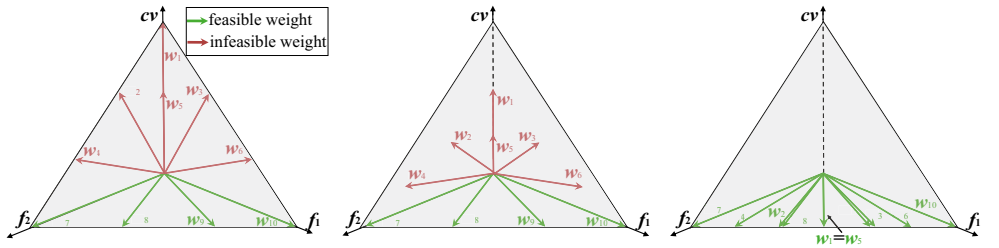
(a) Infeasible solutions with good diversity (b) Infeasible solutions with good convergence (c) Infeasible solutions for locating constraint boundaries

Fig. 1. The illustration of effects of three kinds of infeasible solutions.



(a) early stage (b) middle stage (c) last stage

Fig. 2. Weight settings for a constrained single-objective optimization problem at the early, middle, and last stages, respectively.



(a) early stage (b) middle stage (c) last stage

Fig. 3. Weight settings for a constrained bi-objective optimization problem at the early, middle, and last stages, respectively.

for the constraint violation objective. For constrained single-objective optimization, the value of infeasible weights can be gradually adjusted to 0 [33,40]. At the later stage of the search process, all weights coincide with just one weight vector. This weight vector locates on the feasible region which means its constraint violation equals 0. In this way, all solutions of the population can converge to a feasible optimum.

Nevertheless, unlike constrained single-objective optimization which aims to find just one optimum, constrained multi- and many-objective optimization needs to locate a set of trade-off solutions. In [32], the authors proposed a directed weight setting for utilizing both the feasible infeasible solutions to solve constrained bi- and tri-objective optimization problems. As illustrated in Fig. 3, it gradually adjusts infeasible weight values of $(m + 1)$-dimensional to 0, which means it pushes the infeasible weights to the feasible objective space. However, two problems will arise: 1) the final obtained weights will result in the uneven distribution of the feasible objective space, as shown in Fig. 3(c), which cannot lead to a set of well-distributed Pareto optimal solutions; 2) the setting of population size (number of weights) is a crucial issue. For example, in Das and Dennis's systematic method [5], the number of weights depends on the number of objectives. When using multi-objective constraint-handling techniques regarding the constraint violation (cv) as an objective, the $(m + 1)$-th space weights should be constructed. Nevertheless, it is not fair to compare it with an algorithm with the m-objective weight setting.

Motivated by the above considerations, this paper proposes a two-type weight setting and adjustment method in MOEA/D to fully utilize the infeasibility information to guide the search towards promising regions. In the proposed method, an additional layer weights in the infeasible objective space are designed for saving more informative infeasible solutions. During the course of the search, certain infeasible weights with larger $(m + 1)$-dimensional values will be gradually deleted. At last, all infeasible weights will be removed, to drag the population from promising infeasible regions to optimal feasible regions. The feasible weights stay still in the whole evolutionary procedure, to produce a set of well-distributed Pareto optimal solutions. The gradually removed infeasible weights will not affect the uniform distribution of feasible weights, and the number of final obtained solutions is guided by the original m-objective feasible weights. Thus it is easy and fair to make a comparison with others.

3. The Proposed Algorithm: MOEA/D-2WA

The pseudo-code of the proposed MOEA/D-2WA is outlined in Algorithm 1. The overall algorithm framework is similar to MOEA/D [47]. It begins with the generation of NP feasible weights and NP infeasible weights by different weight generation methods. Accordingly, $2\times NP$ initial solutions can be generated. In the main while-loop, if the halting criterion is not met, i.e., the number of function evaluations is less than the maximum number of function evaluations ($t < maxT$), as the dynamic

constraint boundary $\boldsymbol{\varepsilon}$ shrinks dynamically, the number of weights and solutions is also reduced accordingly. Fig. 4 presents the flowchart of the proposed MOEA/D-2WA. In the following subsections, the implementation process of each major step in Fig. 4 will be introduced in detail.

Algorithm 1: MOEA/D-2WA

Input: NP: population size for feasible weight vectors;
T: the size of neighborhood;
δ: the probability of selection from the neighborhood;
n_r: the maximal number of solutions replaced by an offspring

Output: The feasible nondominated solutions.

GenerateInitialWeights(NP); /* **Algorithm 2***/
Determine the neighbors of each weight vector;
Initialize the population $\boldsymbol{P}$ with $2\times NP$ solutions;
Initialize the initial dynamic constrained boundary $\boldsymbol{\varepsilon}$;
t=$2\times NP$;
while the halting criterion is not met **do**
  Shrink the dynamic constrained boundary $\boldsymbol{\varepsilon}$ according to Eq. (6);
  UpdateWeights($\boldsymbol{W},\boldsymbol{\varepsilon}$); /* **Algorithm 3***/
  if $|\boldsymbol{W}| < |\boldsymbol{P}|$ **then**
    Determine the neighbors of each weight vector;
    UpdatePopulation($\boldsymbol{P},\boldsymbol{\varepsilon},\boldsymbol{W}$). /* **Algorithm 4***/
  end
  for i = 1 to $|\boldsymbol{P}|$ **do**
    if *uniform(0,1)*$< \delta$ **then**
      $E \leftarrow B(i)$;
    else
      $E \leftarrow \{1,\cdots,|\boldsymbol{P}|\}$.
    end
    Apply genetic operators to create an offspring $\boldsymbol{x}_{new}$;
    Update the ideal point $\boldsymbol{z}^* = (z_1^*,\cdots,z_m^*,0)$;
    Replace neighbor solutions. /***Section 3.5***/
  end
end
return feasible nondominated solutions.

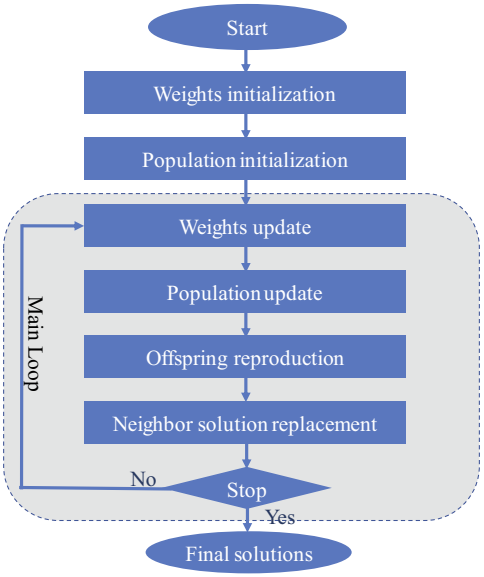


Fig. 4. Flow chart of MOEA/D-2WA.

3.1. Constraint Handling

The constraint violation is often used to deal with constrained optimization problems. In this paper, the constraint violation is defined as the normalized violation of $\boldsymbol{x}$ on all constraints:

$$cv(\boldsymbol{x}) = \frac{1}{q}\sum_{i=1}^{q}\frac{G_i(\boldsymbol{x})}{\max\limits_{\boldsymbol{x}\in\boldsymbol{P}}\{G_i(\boldsymbol{x})\}}, \tag{2}$$

where $\boldsymbol{P}$ denotes the initial population, $G_i(\boldsymbol{x})$ represents the degree of the constraint violation of $\boldsymbol{x}$ on the i-th constraint:

$$G_i(\boldsymbol{x}) = \max\{g_i(\boldsymbol{x}), 0\}, i = 1, 2, \ldots, q. \tag{3}$$

In Eq. (2), if the $\max\limits_{\boldsymbol{x}\in\boldsymbol{P}}\{G_i(\boldsymbol{x})\} < 1, i = 1, 2, \cdots, q$, the $\max\limits_{\boldsymbol{x}\in\boldsymbol{P}}\{G_i(\boldsymbol{x})\}$ will be replaced by 1.

If the search is only guided by the degree of constraint violation cv, the population may be stuck in an infeasible area [49]. Thus, this paper employs the dynamic multi-objective technique [45,46] to handle constraints. It was originally proposed to handle constrained single-objective optimization problems, here we extend it to solve CMaOPs. It recasts an m-objective CMaOP $(f_1(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x}))$ as an $(m+1)$-objective CMaOP $(f_1(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x}), cv(\boldsymbol{x}))$ by regarding the constraint violation as an objective.

Initially, the concept of dynamic constraint boundary $\boldsymbol{\varepsilon}$ is varied by the number of generations [45]. Nevertheless, since the number of weights in this paper is dynamically changed, which means the number of subproblems or the population size at each generation is not a constant, we cannot decrease the dynamic constraint boundary $\boldsymbol{\varepsilon}$ according to generations. However, an alternative method could be used which shrinks the dynamic constraint boundary $\boldsymbol{\varepsilon}$ in terms of the number of function evaluations. Bearing this in mind, a CMaOP can be transformed into a dynamic CMaOP:

$$\begin{aligned} &\min && \boldsymbol{F}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \cdots, f_m(\boldsymbol{x}), cv(\boldsymbol{x})) \\ &\text{subject to}: && \boldsymbol{g}(\boldsymbol{x}) \leqslant \boldsymbol{\varepsilon}^{(t)} \\ &\text{where} && \boldsymbol{\varepsilon}^{(t)} = (\varepsilon_1^{(t)}, \cdots, \varepsilon_q^{(t)}), \\ & && t = 0, 1, \cdots, maxT \end{aligned} \tag{4}$$

where $\boldsymbol{\varepsilon}^{(t)}$ denotes the dynamic constraint boundary, t represents the number of function evaluations, and $maxT$ is the maximum number of function evaluations. A solution satisfies $\boldsymbol{g}(\boldsymbol{x}) \leqslant \boldsymbol{\varepsilon}^{(t)}$ is called an $\boldsymbol{\varepsilon}$-feasible solution; otherwise, it is an $\boldsymbol{\varepsilon}$-infeasible solution.

The problem formulation of Eq. (4) differs from the original CMaOP problem formulation in Eq. (1) with two changes: 1) a constraint violation objective $cv(\boldsymbol{x})$ is added; 2) the original constraint boundary 0 is relaxed to the $\boldsymbol{\varepsilon}$. To establish an equivalent problem transformation, the $\boldsymbol{\varepsilon}$ constraint boundary is dynamically reduced along with the number of function evaluations. In this way, when t reaches $maxT, \boldsymbol{\varepsilon}^{(maxT)} = \boldsymbol{0}$ and $cv(\boldsymbol{x}) = 0$, which means Eq. (4) has the form equivalent to Eq. (1):

$$\begin{aligned} &\min && \boldsymbol{F}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x}), 0) \\ &\text{subjectto}: && \boldsymbol{g}(\boldsymbol{x}) \leqslant \boldsymbol{0} \end{aligned} \tag{5}$$

The shrink of the dynamic constraint boundary $\boldsymbol{\varepsilon}$ employs the exponential formula of the simulated annealing algorithm [20]:

$$\varepsilon_i^{(t)} = A_i e^{-\left(\frac{t}{B_i}\right)^{cp}} - \delta, i = 1, 2, \cdots, q \tag{6}$$

where δ is a close-to-zero value ($\delta = 1e-8$), cp controls the decreasing trend of $\boldsymbol{\varepsilon}$.

In order to make the initial population P achieves $\boldsymbol{\varepsilon}$ feasible, the maximal violation of a constraint in the initial population $\boldsymbol{P}$ is taken as the initial dynamic constraint boundary: $\varepsilon_i^{(0)} = \max_{\boldsymbol{x}\in\boldsymbol{P}}\{G_i(\boldsymbol{x})\}, i = 1, 2, \cdots, q$ at $t = 0$ for $G_i(\boldsymbol{x})$. The ultimate dynamic constraint boundary is $\boldsymbol{\varepsilon}^{(maxT)}$=0. From the initial and the ultimate dynamic constraint boundary, the constants A_i and B_i in Eq. (6) can be obtained, respectively:

$$\begin{cases} A_i = \varepsilon_i^{(0)} + \delta \\ B_i = \dfrac{MaxT}{\sqrt[cp]{ln\left(\frac{\varepsilon_i^{(0)}+\delta}{\delta}\right)}} \end{cases}. \tag{7}$$

3.2. Initial Weights Generation

By taking the degree of constraint violation as an objective, the objective vector $\boldsymbol{F}(\boldsymbol{x})$ consists of m + 1 objectives, i.e., $(f_1(\boldsymbol{x}), \cdots, f_m(\boldsymbol{x}), cv(\boldsymbol{x}))$. In line with the feasibility of the (m + 1)-th objective $cv(\boldsymbol{x})$, the whole objective space $\mathbb{R}^{m+1}$ can be divided into two parts: feasible objective space and infeasible objective space. The feasible objective space is the objective space composed of all feasible solutions:

$$\mathbb{R}_f^{m+1} = \{\boldsymbol{F}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \cdots, f_m(\boldsymbol{x}), cv(\boldsymbol{x}))|cv(\boldsymbol{x}) = 0, \boldsymbol{x} \in \Omega\}$$

Algorithm 2: GenerateInitialWeights(*NP*)

Input: NP: the number of feasible weights

Output: Weight set $\boldsymbol{W}$ including NP feasible weights and NP infeasible weights

Using two-layer method [8] to generate NP feasible weights to form a weight set $\boldsymbol{W}$, i.e., $\boldsymbol{w}_i = (w_{i1}, \cdots, w_{im})$, and set their m+1 dimension to 0, i.e., $\boldsymbol{w}_i = (w_{i1}, \cdots, w_{im}, 0)$, $i \in \{1, \cdots, NP\}$;

Randomly generate enough (m+1)-dimensional infeasible weights to form a weight set $\boldsymbol{R}$;

while $|\boldsymbol{W}| < 2 \times NP$ **do**

  for each $\boldsymbol{r}$ in $\boldsymbol{R}$ **do**

    Assign $\pi(\boldsymbol{r})$=$\boldsymbol{w}$: $\arg\min_{\boldsymbol{w}\in\boldsymbol{W}} angle(\boldsymbol{r}, \boldsymbol{w})$;

  end

  $\boldsymbol{r}^*$=$\arg\max_{\boldsymbol{r}\in\boldsymbol{R}} angle(\boldsymbol{r}, \pi(\boldsymbol{r}))$;

  $\boldsymbol{W} = \boldsymbol{W} \bigcup \{\boldsymbol{r}^*\}$;

  $\boldsymbol{R} = \boldsymbol{R} \backslash \boldsymbol{r}^*$.

end

return $\boldsymbol{W}$.

Similarly, the infeasible objective space is the objective space composed of all infeasible solutions:

$$\mathbb{R}_{inf}^{m+1} = \{\boldsymbol{F}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \cdots, f_m(\boldsymbol{x}), cv(\boldsymbol{x}))|cv(\boldsymbol{x}) > 0, \boldsymbol{x} \in \Omega\}$$

The weights designed in the feasible objective space are called **feasible weights** and those designed in the infeasible objective space are called infeasible weights, respectively. A feasible weight $\boldsymbol{w}$ satisfies $w_{m+1} = 0$, and an infeasible weight $\boldsymbol{w}$ satisfies $w_{m+1} > 0$. Ideally, a feasible weight corresponds to a feasible solution. However, for some problems with small

feasible regions, it is hard to obtain feasible solutions at the early stage of evolution, so the solutions guided by feasible weights are not always feasible. Nevertheless, once feasible solutions are found, the feasible weights will lead to a good distribution of these feasible solutions. This study has the following considerations for the setting of two-type weights:

- For feasible weights, the CMaOEA aims to find a set of feasible Pareto optimal solutions which are uniformly distributed along the PF. In the many-objective space, the uniform distribution of solutions highly depends on the weights' setting. If the distribution or number of feasible weights fluctuates frequently, the obtained solutions will also change regularly, which cannot guarantee the high-quality approximated Pareto optimal solutions. For this reason, feasible weights should be well-predefined and stay unchanged throughout the whole evolution process.
- For infeasible weights, the primary purpose of these weights is to exploit informative infeasible solutions, as the objective information of these infeasible solutions contribute to helping the population jump out of the infeasible barrier or local feasible regions, and also locating multiple feasible subregions. They serve distinct purposes compared with the feasible weights. These infeasible weights are just auxiliary, they do not need a very evenly distribution. On the one hand, the Pareto optimal solutions of a CMaOP are all feasible, so these infeasible weights should be removed at last. Therefore, in this paper, combined with the constraint-handling technique depicted in Section 3.1, the number of infeasible weights is dynamically reduced to 0.

The pseudo-code of the initial weight generation is given in Algorithm 2. For feasible weights, we employ the two-layer (a boundary layer and an inside layer) approach [8] to generate *NP* feasible weight vectors to form a weight set $\boldsymbol{W}$. The weight set $\boldsymbol{W}$ is sampled from a unit simplex. Traditional, the total number (*NP*) of weights for an m-objective problem with p divisions is

$$NP = C_{m+p-1}^{p} \tag{8}$$

However, if the number of objectives is large, most of the weights will locate on the boundary, which could aggravate the performance of the guided subpopulation on maintaining the population diversity. To tackle this issue, two-layer weights which include a boundary layer and an inside layer can be used to handle MaOPs [8]. Each generated weight vector $\boldsymbol{w}$ satisfies $\sum_{k=1}^{m} w_k = 1$. To guarantee these *NP* weights feasible, we set their m + 1 dimension to 0, i.e., $\boldsymbol{w}_i = (w_{i1}, \cdots, w_{im}, 0), i \in \{1, \cdots, NP\}$.

Usually, a *priori* knowledge about test instances is unavailable, making it difficult to know the geometric properties of test functions (e.g., the shape and the size of feasible regions). Thus, to be consistent with the number of feasible weights, a set of *NP* infeasible weights are generated as follows. Firstly, enough (e.g., 5000) weights are uniformly random generated [48] for forming the weight set $\boldsymbol{R}$. To guarantee these weights are infeasible, each weight vector $\boldsymbol{r}$ in $\boldsymbol{R}$ satisfies $\sum_{k=1}^{m+1} r_k = 1, r_k \geqslant 0$ for $k \in \{1, \cdots, m\}$ and $r_{m+1} > 0$. Secondly, we find the weight in $\boldsymbol{R}$ with the largest angle to $\boldsymbol{W}$, add it to $\boldsymbol{W}$ and remove it from $\boldsymbol{R}$. The angle of two weights is calculated as

$$angle(\boldsymbol{r}, \boldsymbol{w}) = \arccos\left(\frac{\boldsymbol{r} \bullet \boldsymbol{w}}{||\boldsymbol{r}||_2 \cdot ||\boldsymbol{w}||_2}\right) \tag{9}$$

where $\boldsymbol{r} \bullet \boldsymbol{w}$ represents the inner product of two weights $\boldsymbol{r}$ and $\boldsymbol{w}$, and $||.||$ calculates the norm of a weight. It is clear that the angle between $\boldsymbol{r}$ and $\boldsymbol{w}$ is in the range of $[0, \frac{\pi}{2}]$. Afterward, if the size of $\boldsymbol{W}$ equals to $2 \times NP$, stop and return $\boldsymbol{W}$. Otherwise, repeat the above process.

3.3. Weights update

During the evolution process, the weight set $\boldsymbol{W}$ is updated according to the changes of the dynamic constraint boundary $\boldsymbol{\varepsilon}$. In other words, with the shrink of the dynamic constraint boundary $\boldsymbol{\varepsilon}$, some infeasible weights are no longer useful for the search since they are far from feasible regions. To this end, if the (m + 1)-th dimension of a weight $\boldsymbol{w}$ is greater than the $\boldsymbol{\varepsilon}$

level, which means solutions guided by this weight have larger constraint violation, so this weight vector is not conducive for solutions to approach the feasible area. Accordingly, this ineffective weight should be removed in time.

Algorithm 3: UpdateWeights($\boldsymbol{W}, \varepsilon$)

Input: Weight set $\boldsymbol{W}$, dynamic constrained boundary ε

Output: Updated weight set $\boldsymbol{W}$

```
for each w in W do
    if w_{m+1} > ||ε^(t)|| / ||ε^(0)|| then
        W = W \ w.
    end
end
return W.
```

Algorithm 3 depicts the procedure of updating weight set. For each weight, if the value of its $(m + 1)$-th dimension is larger than the ε level, it will be excluded from the weight set $\boldsymbol{W}$. Note that due to the $(m + 1)$-th dimension value w_{m+1} of a weight $\boldsymbol{w}$ is in the normalized range of [0,1], this paper compares the normalized ε level $\frac{||\varepsilon^{(t)}||}{||\varepsilon^{(0)}||}$ with w_{m+1}, since $\frac{||\varepsilon^{(t)}||}{||\varepsilon^{(0)}||}$ has also normalized in the range of [0,1], where $\varepsilon^{(0)}$ represents the initial maximum constraint violation. After deleting useless weights, the number of subproblems will be reduced accordingly.

3.4. Population Update

Ideally, each weight corresponds to one distinct solution in the decomposition-based MOEAs. The aim of the population update is to delete the superfluous $(|\boldsymbol{P}| - |\boldsymbol{W}|)$ solutions from the population $\boldsymbol{P}$, so that to keep the number of solutions of population $\boldsymbol{P}$ is the same as the number of subproblems (weight set $\boldsymbol{W}$). Algorithm 4 describes the process of updating the population $\boldsymbol{P}$. Based on the value of dynamic constraint boundary ε, the population can be divided into an ε-feasible set $\boldsymbol{S}_1 = \{\boldsymbol{x}|\boldsymbol{g}(\boldsymbol{x}) <= \varepsilon, \boldsymbol{x} \in \boldsymbol{P}\}$ and an ε-infeasible set $\boldsymbol{S}_2 = \{\boldsymbol{x}|\exists g_i(\boldsymbol{x}) > \varepsilon_i, i \in \{1, \ldots, q\}, \boldsymbol{x} \in \boldsymbol{P}\}$. We prefer to select ε-feasible solutions in $\boldsymbol{S}_1$. To be specific, two situations are taken into our consideration:

- If the size of ε-feasible set $\boldsymbol{S}_1$ is greater than the number of weight set $\boldsymbol{W}$, which means a total of $(|\boldsymbol{S}_1| - |\boldsymbol{W}|)$ solutions need to be deleted from the ε-feasible set $\boldsymbol{S}_1$. Herein, the deletion of ε-feasible solutions is based on the diversity of the ε-feasible set $\boldsymbol{S}_1$. In the many-objective space, the angle could reflect the diversity degree. The vector angle is the included angle between two solutions in the normalized objective space. In this paper, the normalized objective vector of a solution is calculated as follows. First, the ideal point $\boldsymbol{z}^{min} = (z_1^{min}, \cdots, z_m^{min})$ and the nadir point $\boldsymbol{z}^{max} = (z_1^{max}, \cdots, z_m^{max})$ in $\boldsymbol{S}_1$ can be found, where z_k^{min} and z_k^{max} denote the minimal and maximal values of the k-th objective of the ε-feasible set $\boldsymbol{S}_1$, respectively, $k = 1, \cdots, m$. Afterward, for the i-th solution $\boldsymbol{x}_i$, its objective vector $\boldsymbol{F}(\boldsymbol{x}_i)$ is normalized as $\boldsymbol{F}'(\boldsymbol{x}_i) = (f'_1(\boldsymbol{x}_i), \cdots, f'_m(\boldsymbol{x}_i))$, where $f'_k(\boldsymbol{x}_i)$ is computed according to

$$f'_k(\boldsymbol{x}_i) = \frac{f_k(\boldsymbol{x}_i) - z_k^{min}}{z_k^{max} - z_k^{min}}, k = 1, \cdots, m. \tag{10}$$

  Then, the included angle for every two solutions in the normalized objective space can be calculated:

$$angle(\boldsymbol{F}'(\boldsymbol{x}_i), \boldsymbol{F}'(\boldsymbol{x}_j)) = \arccos\left(\frac{\boldsymbol{F}'(\boldsymbol{x}_i) \bullet \boldsymbol{F}'(\boldsymbol{x}_j)}{||\boldsymbol{F}'(\boldsymbol{x}_i)||_2 \cdot ||\boldsymbol{F}'(\boldsymbol{x}_j)||_2}\right), \tag{11}$$

  and find two solutions with the smallest angle, which signifies these two solutions have the poorest diversity. At last, the one with the larger degree of constraint violation among these two solutions will be deleted. The above deletion operator is repeated until the number of the solution set $\boldsymbol{S}_1$ is equal to the number of weight set $\boldsymbol{W}$.

- If $|\boldsymbol{S}_1| < |\boldsymbol{W}|$, which means the number of $\boldsymbol{\varepsilon}$-feasible solutions is less than the number of weight set, there are too many $\boldsymbol{\varepsilon}$-infeasible solutions that impede the search for regions with smaller constraint violations. Under this circumstance, solutions with a larger degree of constraint violation should be removed in time. Herein, we sort the solutions in $\boldsymbol{S}_2$ in terms of their degree of constraint violation, and delete the worst ($|\boldsymbol{P}|$-$|\boldsymbol{W}|$) solutions from the sorted $\boldsymbol{S}_2$.

Algorithm 4: UpdatePopulation($\boldsymbol{P}, \boldsymbol{\varepsilon}, \boldsymbol{W}$)

Input: Population $\boldsymbol{P}$, dynamic constrained boundary $\boldsymbol{\varepsilon}$, weight vector set $\boldsymbol{W}$

Output: Updated population $\boldsymbol{P}$

Divided the population $\boldsymbol{P}$ into an $\boldsymbol{\varepsilon}$-feasible set $\boldsymbol{S}_1 = \{\boldsymbol{x} | \boldsymbol{g}(\boldsymbol{x}) <= \boldsymbol{\varepsilon}, \boldsymbol{x} \in \boldsymbol{P}\}$ and an $\boldsymbol{\varepsilon}$-infeasible set $\boldsymbol{S}_2 = \{\boldsymbol{x} | \exists g_i(\boldsymbol{x}) > \varepsilon_i, i \in \{1, ..., q\}, \boldsymbol{x} \in \boldsymbol{P}\}$;
if $|\boldsymbol{S}_1| \geq |\boldsymbol{W}|$ **then**
  Delete all solutions in $\boldsymbol{S}_2$ from $\boldsymbol{P}$;
  Find the ideal point $\boldsymbol{z}^{min}$, where $z_k^{min} = \min_{\boldsymbol{x} \in \boldsymbol{S}_1} f_k(\boldsymbol{x}), k = 1, \cdots, m$;
  Find the nadir point $\boldsymbol{z}^{max}$, where $z_k^{max} = \max_{\boldsymbol{x} \in \boldsymbol{S}_1} f_k(\boldsymbol{x}), k = 1, \cdots, m$;
  Normalize the objective vector for each solution in $\boldsymbol{S}_1$ according to Eq. (10);
  while $|\boldsymbol{P}| > |\boldsymbol{W}|$ **do**
    Calculate the acute angle between every two solutions in $\boldsymbol{S}_1$ according to Eq. (11);
    Find two solutions $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ with minimum angle;
    if $cv(\boldsymbol{x}_i) > cv(\boldsymbol{x}_j)$ **then**
      $\boldsymbol{P} = \boldsymbol{P} \backslash \boldsymbol{x}_i$;
    else
      $\boldsymbol{P} = \boldsymbol{P} \backslash \boldsymbol{x}_j$.
    end
  end
else
  Sort solutions in $\boldsymbol{S}_2$ according to their degree of constraint violation;
  Delete the worst ($|\boldsymbol{P}|$-$|\boldsymbol{W}|$) solutions from $\boldsymbol{S}_2$.
end
return population $\boldsymbol{P}$.

3.5. Neighbor solution replacement principle

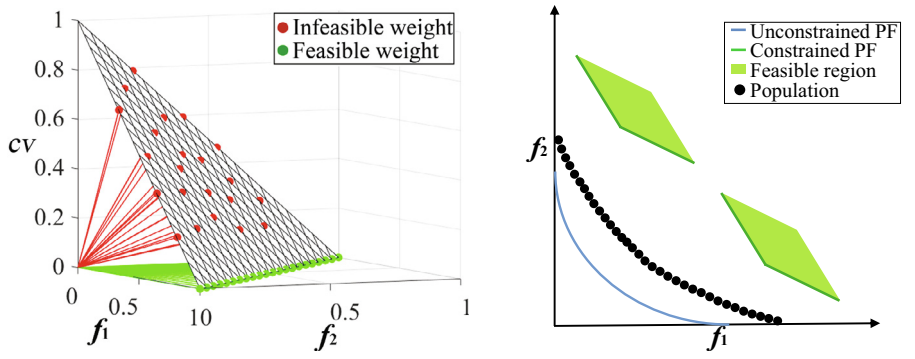
In line 21 of Algorithm 1, when an offspring $\boldsymbol{x}_{new}$ is compared with a randomly selected solution $\boldsymbol{x}$ from its neighborhood, the $\boldsymbol{\varepsilon}$-feasibility of both solutions will be checked. The neighbor solution $\boldsymbol{x}$ will be replaced by the child solution $\boldsymbol{x}_{new}$ under the following three scenarios:

- $\boldsymbol{x}_{new}$ is $\boldsymbol{\varepsilon}$-feasible while $\boldsymbol{x}$ is $\boldsymbol{\varepsilon}$-infeasible;
- $\boldsymbol{x}_{new}$ and $\boldsymbol{x}$ are both $\boldsymbol{\varepsilon}$-infeasible, but the constraint violation of $\boldsymbol{x}_{new}$ is smaller than $\boldsymbol{x}$, i.e., $cv(\boldsymbol{x}_{new}) < cv(\boldsymbol{x})$;
- $\boldsymbol{x}_{new}$ and $\boldsymbol{x}$ are both $\boldsymbol{\varepsilon}$-feasible, but the aggregation function value of $\boldsymbol{x}_{new}$ is better than the aggregation function value of $\boldsymbol{x}$.

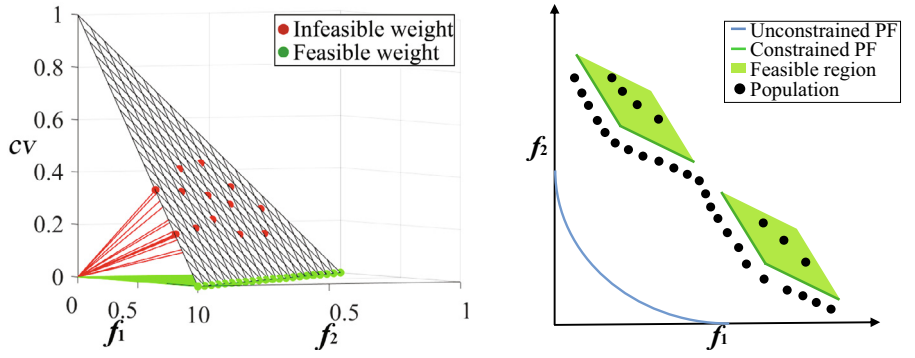
The aggregation function is used to distinguish the quality of solutions. In this paper, the penalty-based boundary intersection (PBI) method is employed as the aggregation function, because of its promising performance on MaOPs [8]. The optimization formulation of PBI method is:

$$\min f(\boldsymbol{x} | \boldsymbol{w}, \boldsymbol{z}^*) = d_1 + \theta d_2 \tag{12}$$

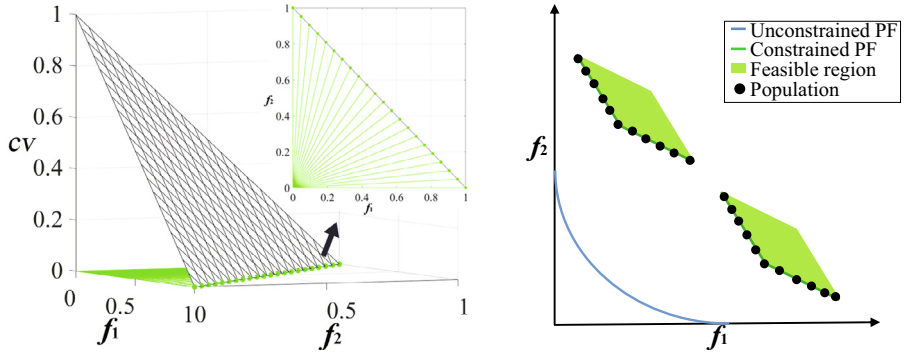
where d_1 denotes the distance from the ideal point $\boldsymbol{z}^*$ to the solution vector $\boldsymbol{F}(\boldsymbol{x})$ along the reference line specified by a weight $\boldsymbol{w}$, and d_2 represents the distance from the reference line to the $\boldsymbol{F}(\boldsymbol{x})$:



(a) At the early stage



(b) At the middle stage



(c) At the later stage

Fig. 5. Schematic representation of weight adjustments (left figures) and solutions' movement (right figures) at the early, middle, and later stages of evolution on a bi-objective constrained optimization problem, respectively.

$$
\begin{aligned}
d_1 &= \frac{|(\boldsymbol{F}(\boldsymbol{x})-\boldsymbol{z}^*)^T\boldsymbol{w}|}{||\boldsymbol{w}||} \\
d_2 &= ||\boldsymbol{F}(\boldsymbol{x}) - \boldsymbol{z}^* - d_1\frac{\boldsymbol{w}}{||\boldsymbol{w}||}||
\end{aligned}
\tag{13}
$$

where $\boldsymbol{F}(\boldsymbol{x})$ is the transformed $(m+1)$ objectives: $\boldsymbol{F}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \cdots, f_m(\boldsymbol{x}), cv(\boldsymbol{x}))$. Here, each ideal point $\boldsymbol{z}^* = (z_1^*, \cdots, z_m^*, z_{m+1}^*)$ is specified by the best value of each objective $f_i(\boldsymbol{x})$ and the constraint violation $cv(\boldsymbol{x})$ among evaluated solutions, i.e., $z_i^* = \min_{\boldsymbol{x}\in\boldsymbol{P}} f_i(\boldsymbol{x}), i = 1, \cdots, m$, and $z_{m+1}^* = \min_{\boldsymbol{x}\in\boldsymbol{P}} cv(\boldsymbol{x})$. Note that the best value of the constraint violation $cv(\boldsymbol{x})$ is known, which is equal to 0. Therefore, z_{m+1}^* can be replaced with 0, i.e., $z_{m+1}^* = 0$.

To make a clear explanation of the working mechanism of the proposed MOEA/D-2WA, an example is presented in Fig. 5. In this schematic representation, a bi-objective constrained optimization problem is transformed into a tri-objective opti-

mization problem by adding the constraint violation cv as an objective. As illustrated in Fig. 5(a), at the early stage of evolution, the feasible weights and infeasible weights are evenly distributed in the objective and constraint violation spaces. Infeasible solutions with good convergence can drive the population to cross the infeasible barrier quickly and towards the unconstrained PF. In Fig. 5(b), at the middle stage of evolution, infeasible weights away from the feasible objective space are gradually removed, to give infeasible solutions with smaller constraint violation a great chance to survive. Meanwhile, feasible weights could push solutions to locate feasible areas and drive infeasible solutions that are close to the constraint boundary to move to feasible regions. As depicted in Fig. 5(c), at the final stage of evolution, all infeasible weights are deleted, the feasible weights are used to guide the population to search for Pareto optimal solutions in the feasible region.

4. Experimental results

4.1. Compared algorithms

Six state-of-the-art and representative CMaOEAs were chosen as competitors of MOEA/D-2WA: C-NSGA-III [16], C-MOEA/D [16], I-DBEA [1], PPS [11], C-TAEA [23], and C-AnD [28]. Similar to this paper, C-NSGA-III, C-MOEA/D, I-DBEA, PPS, and C-TAEA belong to the decomposition-based MaOEAs by using weight vectors (reference vectors), while C-And can be classified as the angle-based MaOEA. As for the constraint-handling techniques, C-NSGA-III, C-MOEA/D, and C-AnD all adopt CDP to compare pairs of solutions, I-DBEA employs ε-constrained method, while PPS and C-TAEA belong to two-phase optimization and two-archive approach, respectively. All experiments were carried out in the open-source software PlatEMO [35].

4.2. Test instances

The proposed and compared algorithms were tested on three well-known test suites: C-DTLZ [8], DC-DTLZ [23], and MW [29]. The estimated feasible ratios for these three sets of test suites are presented in Table S-I of the supplementary file. It can be seen that most of the test problems have small feasible regions, thus they can fully evaluate the performance of a CMaOEA.

4.3. Parameter settings

In MOEA/D-2WA, the simulated binary crossover (SBX) [6] and polynomial mutation (PM) [7] were employed as the reproduction operators:

- SBX: crossover probability p_c=0.9 and distribution index η_c=20;
- PM: mutation probability p_m=1/D and distribution index η_m=20;

The common parameters in MOEA/D:

- Penalty Parameter in PBI: θ = 5 [47,24,8];
- Neighborhood size: T = 20;
- Maximum number of solutions replaced by an offspring: n_r=2 [22];
- Probability used to choose in the neighborhood: δ=0.9;

Other settings:

- The control parameter cp = 4;
- Independently runs for each algorithm: 30.

The number of weights and the population size that MOEA/D-2WA adopted, and the termination condition (the maximum number of function evaluations $maxT$) of compared algorithms for each test problem are listed in Tables S-II and S-III in the supplementary file, respectively. The other parameters for each compared algorithm are set according to their original papers.

The following two indicators are employed to measure the quality of obtained solutions by each algorithm.

Inverted Generational Distance (IGD) [3]: it measures the mean distance from the points in the true PF $\boldsymbol{P}^*$ to their closest solution in the obtained population $\boldsymbol{P}$:

$$IGD(\boldsymbol{P}^*, \boldsymbol{P}) = \frac{\sum_{\boldsymbol{x}\in\boldsymbol{P}^*} dist(\boldsymbol{x}, \boldsymbol{P})}{|\boldsymbol{P}^*|} \tag{14}$$

where dist($\boldsymbol{x}$,$\boldsymbol{P}$) is the Euclidean distance between $\boldsymbol{x}$ and its nearest neighbor in $\boldsymbol{P}$. A small IGD value indicates that the obtained population is close to the true PF as well as having a good distribution, which could measure both the convergence

and diversity simultaneously. In this paper, approximately 10,000 uniformly distributed points sampled on each constrained PF are employed as the $\boldsymbol{P}^*$ for calculating IGD.

Hypervolume (HV) [50]: it measures the volume of the objective space enclosed by the obtained population $\boldsymbol{P}$ and a reference point $\boldsymbol{z}^r$:

$$HV(\boldsymbol{P}) = L(\bigcup_{\boldsymbol{x}\in\boldsymbol{P}}[x_1, z_1^r] \times \cdots \times [x_m, z_m^r]) \tag{15}$$

where $L(.)$ denotes the Lebesgue measure. A large HV value represents a good dominance relation. In our experiments, each objective is first normalized into [0,1] according to the ideal point and the nadir point of the constrained PF. Then, the reference point $\boldsymbol{z}^r$ is set to $(1.1, \cdots, 1.1)^T$. The obtained solutions that do not dominate reference point $\boldsymbol{z}^r$ are discarded for the HV calculation.

Note that the two metrics only consider feasible solutions in the final population. When an algorithm can not find a feasible solution over 30 runs consistently, the mean IGD and HV values are replaced by "NaN".

4.4. Performance on the C-DTLZ test suite

The detailed mean IGD and HV values of seven compared algorithms on the C-DTLZ test suite over 30 runs are listed in Table S-IV and Table S-V of the supplementary file, respectively. The best results for each instance are highlighted with a grey background.

For the C1-DTLZ1 problem, there is a narrow feasible region above the PF. Although the feasible ratio of this problem is 0%, it does not pose too much difficulty to the feasibility-driven CMaOEAs. They can easily drag the population to the feasible area. So this problem tests how to distribute the population evenly on the PF. As can be seen in Table S-IV, CMOEA/D and the proposed MOEA/D-2WA have better performance on this problem. We should also notice that PPS cannot find feasible solutions consistently on the C1-DTLZ1 test problem with 5 objectives. This may be because PPS consists of two-stage optimization, so it needs more function evaluations for this problem.

The C1-DTLZ3 problem has a highly multi-modal landscape and a barrier of infeasible areas, presenting challenges to CMaOEAs in terms of convergence. A CMaOEA is easy to be trapped in the local feasible area and cannot cross the infeasible barrier easily. The results demonstrate that MOEA/D-2WA has an edge in solving this kind of problem. The obtained population of most CMaOEAs can converge to the boundary of a local feasible region, but they are difficult to jump over the infeasible barrier to approach the global feasible area. For MOEA/D-2WA, infeasible solutions with better convergence will beat feasible solutions with worse fitness at the early and middle stages. So this is the reason why MOEA/D-2WA can jump over the infeasible barrier to reach the global feasible region.

For the C2-DTLZ2 problem, the constraints make the PF partially feasible, so its constrained PF is composed of several feasible parts, which poses challenges to CMaOEAs in terms of diversity. The results show that no algorithm can perform best on the C2-DTLZ2 test problem with all different numbers of objectives. Concretely, PPS has the best performance on the tri-objective problem, C-TAEA has the edge over others on 8-objective and 15-objective problems, C-AnD beats others on the 5-objective problem, while the proposed MOEA/D-2WA performs best on the 10-objective problem.

For C3-DTLZ1 and C3-DTLZ4 problems, the constrained PF is located on the constraint boundaries. The feasible regions for these two test problems are also very large. Unlike the feasibility-driven CMaOEAs which approach the constrained PF only from the feasible region, the proposed MOEA/D-2WA is capable of retaining certain promising infeasible solutions during the course of the search, which is useful for the population to approximate the Pareto optimal solutions located on constraint boundaries from both the feasible and infeasible sides.

The statistical results in Table 1 shows that, in terms of the IGD indicator, MOEA/D-2WA is significantly better than C-NSGA-III, C-MOEA/D, I-DBEA, PPS, C-TAEA, and C-AnD on 14, 21, 25, 22, 22, and 20 test problems, respectively. By contrast, these six competitors surpass MOEA/D-2WA on less than four test problems. In terms of the HV indicator, MOEA/D-2WA beats its six competitors on 8, 18, 25, 18, 13, and 14 test problems, which is larger than the number of test problems that MOEA/D-2WA is significantly worse than its competitors except for C-NSGA-III. Table 1 also reports the multiple-problem

Table 1
Statistical test results on the C-DTLZ test suite in terms of IGD and HV, respectively.

Algorithm	IGD				HV			
	$+/-/\approx$	R^+	R^-	α=0.05	$+/-/\approx$	R^+	R^-	α=0.05
MOEA/D-2WA vs C-NSGA-III	14/3/8	259.0	41.0	+	8/10/7	120.5	179.5	$\approx$
MOEA/D-2WA vs C-MOEA/D	21/2/2	313.0	12.0	+	18/5/2	215.0	85.0	$\approx$
MOEA/D-2WA vs I-DBEA	25/0/0	300.0	0.0	+	25/0/0	325.0	0.0	+
MOEA/D-2WA vs PPS	22/2/1	296.0	29.0	+	18/5/2	215.5	84.5	$\approx$
MOEA/D-2WA vs C-TAEA	22/2/1	288.0	37.0	+	13/9/3	155.5	169.5	$\approx$
MOEA/D-2WA vs C-AnD	20/3/2	248.0	52.0	+	14/8/3	160.5	139.50	+

"+", "-", and "≈" denote the number of the performance of MOEA/D-2WA is significantly better than, worse than, and equal to its peers, respectively. R^+, R^- represent the sum of ranks, $R^+ > R^-$ means that the algorithm of this paper is better than the compared algorithm and vice versa.

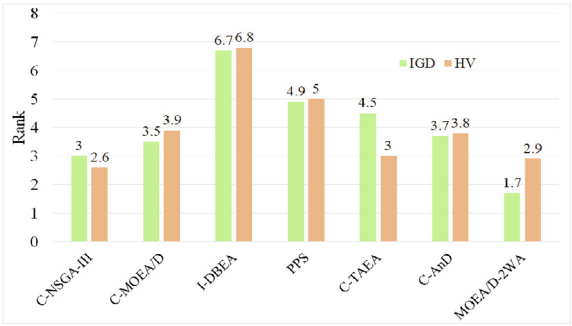


Fig. 6. Friedman's test among seven CMaOEAs on the C-DTLZ test suite in terms of IGD and HV, respectively. The smaller the ranking, the better the performance of an algorithm.

Wilcoxon's test results. As can be seen, in terms of the IGD indicator, the significant difference can be observed in all cases, while in terms of the HV indicator, MOEA/D-2WA performs significantly better than I-DBEA and C-AnD at the $\alpha = 0.05$ significance level. Friedman's test in Fig. 6 illustrates that the proposed MOEA/D-2WA ranks first and second in terms of IGD and HV values among the seven compared algorithms in solving the C-DTLZ test suite, respectively.

4.5. Performance on the DC-DTLZ test suite

The average values of seven algorithms on the DC-DTLZ test suite in terms of the IGD and HV indicators over 30 runs are listed in Table S-VI and Table S-VII, respectively. Different from the C-DTLZ test suite, in which the constraints are constructed on the objective space, the DC-DTLZ suite designs constraints on the decision space, so it is more challenging to be solved.

DC1-DTLZ1 and DC1-DTLZ3 are the first kind of instances. The constrained PF is divided into several isolated parts by some infeasible strips. The obtained population is easily getting trapped into a local feasible region and is hard to find all feasible subareas. The results reveal that the proposed MOEA/D-2WA has the distinct performance on these two problems. Particularly on the DC1-DTLZ1 test problem, e.g., for the tri-objective DC1-DTLZ1 problem, MOEA/D-2WA missed the top tiny feasible subpart. However, an interesting phenomenon can be observed that MOEA/D-2WA can obtain the best results on many-objective DC1-DTLZ3 test problems (i.e., DC1-DTLZ3 with 8, 10, and 15 objectives).

DC2-DTLZ1 and DC2-DTLZ3 belong to the second class of test problems. Its constrained PF is the same as the unconstrained PF, but only a small part of the region close to the PF is feasible. The feasible ratio of this kind of problem is tiny. Moreover, there exist many local optima in infeasible regions, making it much harder for solutions to escape. Only MOEA/D-2WA can converge to the constrained PF on both these two problems, while the others were trapped in local infeasible optima on certain problems.

DC3-DTLZ1 and DC3-DTLZ3 can be generated to the third kind of test instances, which are a mixture of the first two kinds of test instances. As the number of objectives increases, the feasible region becomes smaller and smaller. C-TAEA, C-And, and the proposed MOEA/D-2WA have better performance on these two problems. Specifically, C-TAEA performs best on DC3-DTLZ3 with 3, 5, 8, and 10 objectives and DC3-DTLZ1 with 3 objectives. C-And has the best performance on DC3-DTLZ1 with 5, 8, and 10 objectives. Interestingly, MOEA/D-2WA obtains the best results among these seven algorithms on both 15-objective DC3-DTLZ1 and 15-objective DC3-DTLZ3 test problems. The feasible ratios of 15-objective DC3-DTLZ1 and DC3-DTLZ3 test problems are both 0%, which indicates MOEA/D-2WA has better performance on highly-constrained optimization problems.

From the statistical test results in Table 2, we can find that MOEA/D-2WA is significantly better than C-NSGA-III, C-MOEA/D, I-DBEA, PPS, C-TAEA, and C-AnD on most test problems in terms of both IGD and HV indicators, which is far more than the

Table 2
Statistical test results on the DC-DTLZ test suite in terms of IGD and HV, respectively.

Algorithm	IGD				HV			
	+/−/≈	R^+	R^-	α=0.05	+/−/≈	R^+	R^-	α=0.05
MOEA/D-2WA vs C-NSGA-III	22/5/3	400.0	65.0	+	16/7/7	339.0	126.0	+
MOEA/D-2WA vs C-MOEA/D	20/3/7	414.5	50.5	+	19/4/7	380.5	54.5	+
MOEA/D-2WA vs I-DBEA	30/0/0	465.0	0.0	+	27/0/3	463.5	1.5	+
MOEA/D-2WA vs PPS	24/3/3	428.5	36.5	+	23/2/5	396.5	68.5	+
MOEA/D-2WA vs C-TAEA	18/9/3	305.0	160.0	≈	16/9/5	257.5	177.5	≈
MOEA/D-2WA vs C-AnD	17/7/6	378.0	87.0	+	19/7/4	336.5	98.5	+

"+", "-", and "≈" denote the number of the performance of MOEA/D-2WA is significantly better than, worse than, and equal to its peers, respectively. R^+, R^- represent the sum of ranks, $R^+ > R^-$ means that the algorithm of this paper is better than the compared algorithm and vice versa.

Fig. 7. Friedman's test among seven CMaOEAs on the DC-DTLZ test suite in terms of IGD and HV, respectively. The smaller the ranking, the better the performance of an algorithm.

number of problems where the performance of MOEA/D-2WA is worse than its competitors. As for the multiple-problem Wilcoxon's test results, it is evident that MOEA/D-2WA provides higher R^+ values than R^- values in all the cases. The Friedman's test was carried out on all DC-DTLZ problems in terms of IGD and HV. It is obvious from Fig. 7 that MOEA/D-2WA has the smallest ranking in terms of both IGD and HV, which indicates that MOEA/D-2WA shows the best overall performance than the compared algorithms in solving the DC-DTLZ test suite.

4.6. Performance on the MW test suite

The detailed mean IGD and HV values of seven compared algorithms on the MW test suite over 30 runs are shown in Table S-VIII and Table IX, respectively. It is worth noting that only MW4, MW8, and MW14 problems can be scaled to MaOPs, so the proposed algorithm and compared CMaOEAs be tested only on these three CMaOPs.

The results show that MOEA/D-2WA has the best performance on the MW4 and MW8 problems, and performs poorer on the MW14 problem. The reason can be attributed that although these three test problems all have tiny feasible ratios in the search space, the different objectives of the MW14 problem are scaled to different ranges. So, uniformly distributed weights' setting in the feasible space may not work well on this kind of test problem.

The statistical results in Table 3 suggest that MOEA/D-2WA can surpass its competitors on most test problems in terms of the IGD indicator, and can provide very competitive results in terms of the HV indicator. Concerning the Friedman's test in Fig. 8, MOEA/D-2WA ranks first and second among these seven algorithms in terms of IGD and HV indicators, respectively.

5. Further Discussions

5.1. How two-type weight set works

The salient property of MOEA/D-2WA is it can fully use both promising infeasible and feasible solutions. The utilization of the infeasibility information could be helpful to the search in the many-objective space. To give a deep understanding of the advantage of the proposed two-type weight setting, we investigate the search behavior of MOEA/D-2WA and CMOEA/D on three bi-objective optimization problems in that they all adopt MOEA/D as the base optimization algorithm. However, concerning the constraint-handling technique, they are essentially different. CMOEA/D belongs to the feasibility-driven CMaOEA, while the proposed MOEA/D-2WA uses the information of both infeasible and feasible solutions.

C2-DTLZ2, C1-DTLZ3, and C3-DTLZ1 are chosen as test problems in that they possess diverse and representative characteristics. For example, C2-DTLZ2 has three disjoint and tiny feasible regions so that it can test the diversity of an algorithm.

Table 3
Statistical test results on the MW test suite in terms of IGD and HV, respectively.

Algorithm	IGD				HV			
	$+/-/\approx$	R^+	R^-	α=0.05	$+/-/\approx$	R^+	R^-	α=0.05
MOEA/D-2WA vs C-NSGA-III	9/4/2	59.0	46.0	$\approx$	6/8/1	52.0	68.0	$\approx$
MOEA/D-2WA vs C-MOEA/D	10/4/1	95.0	25.0	+	12/1/2	60.5	59.5	$\approx$
MOEA/D-2WA vs I-DBEA	15/0/0	120.0	0.0	+	15/0/0	120.0	0.0	+
MOEA/D-2WA vs PPS	11/3/1	98.0	22.0	+	12/0/3	91.0	14.0	+
MOEA/D-2WA vs C-TAEA	12/3/0	85.0	35.0	$\approx$	7/6/2	41.0	64.0	$\approx$
MOEA/D-2WA vs C-AnD	8/6/1	48.0	72.0	$\approx$	7/7/1	38.0	67.0	$\approx$

"+", "-", and "$\approx$" denote the number of the performance of MOEA/D-2WA is significantly better than, worse than, and equal to its peers, respectively. R^+, R^- represent the sum of ranks, $R^+ > R^-$ means that the algorithm of this paper is better than the compared algorithm and vice versa.

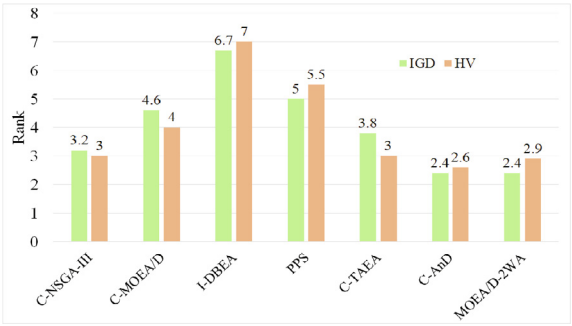
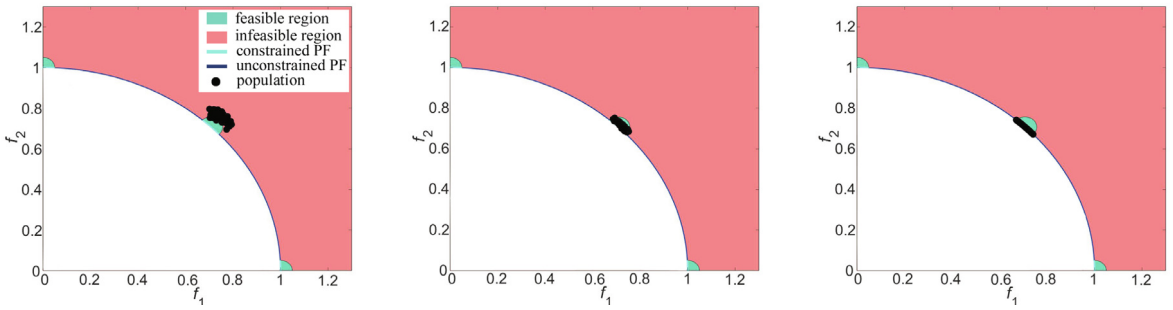


Fig. 8. Friedman's test among seven CMaOEAs on the MW test suite in terms of IGD and HV, respectively. The smaller the ranking, the better the performance of an algorithm.

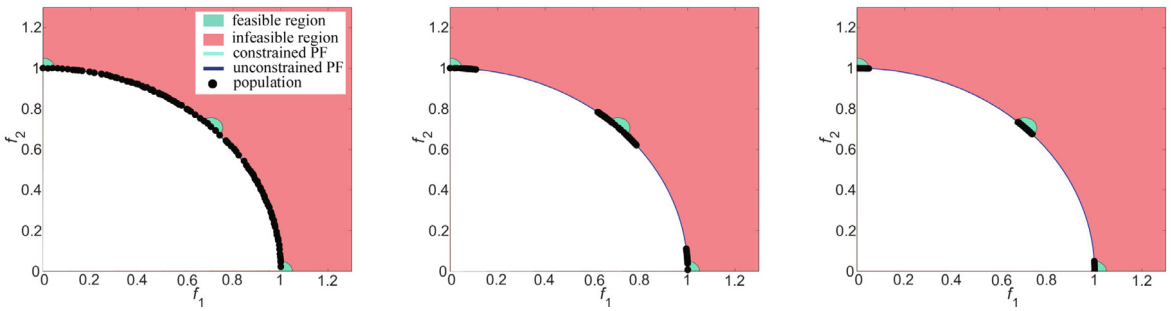
The C1-DTLZ3 problem has large feasible areas, and there is an infeasible gap between the global feasible region and the local feasible region. The difficulty of this problem lies in convergence to the global PF, given how easy it is to get trapped at a local PF. For the C3-DTLZ1 problem, the constrained PF is located on the boundary of the feasible and infeasible regions.

Figs. 9–11 plot the obtained solutions at the early, middle, and later stages of CMOEA/D and MOEA/D-2WA on bi-objective C2-DTLZ2, C1-DTLZ3, and C3-DTLZ1 problems, respectively. Note that all results plotted in figures are with the median IGD value across 10 independent runs.

Infeasible solutions with good diversity contribute to locate multiple feasible subregions: our expectations are closely borne out by Fig. 9. The figures illustrate that MOEA/D-2WA enables isolated infeasible solutions to be preserved to locate multiple disjoint and tiny feasible subregions, this can be attributed to the setting of infeasible weights at the early and middle stages, which could retain more promising infeasible solutions. Accordingly, at the later stage of evolution, all

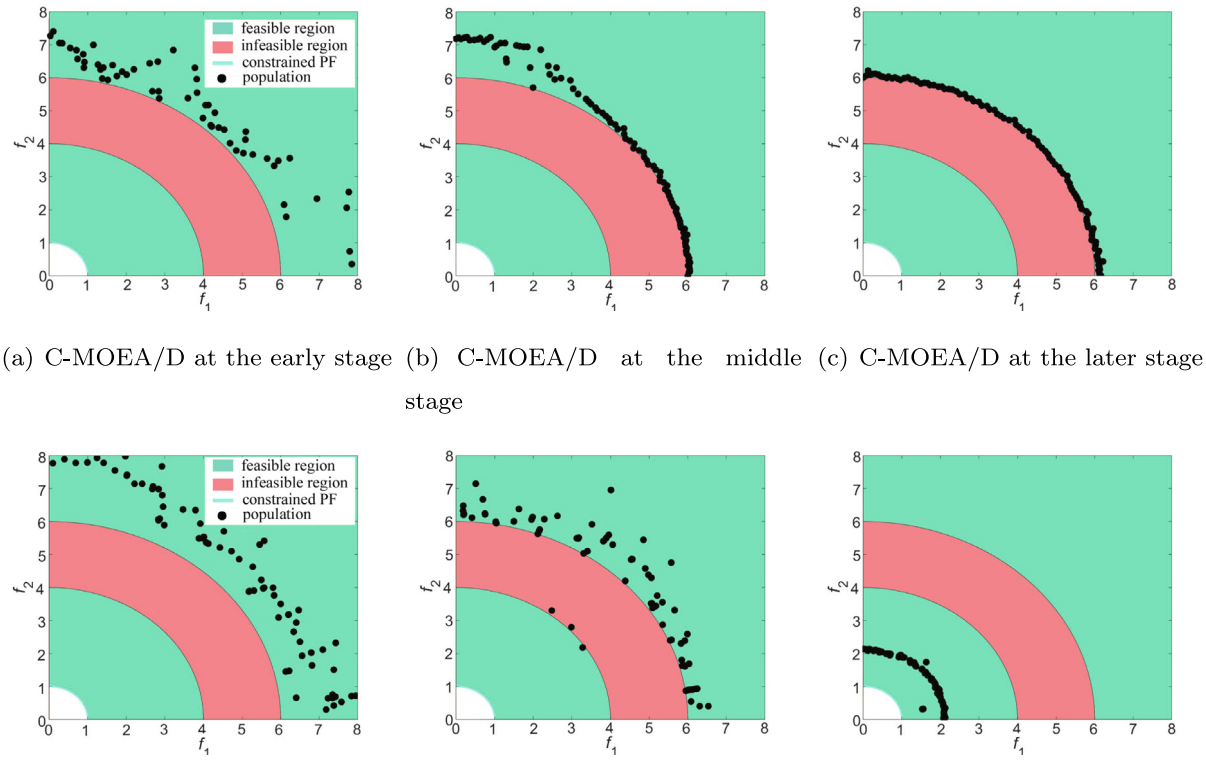


(a) C-MOEA/D at the early stage (b) C-MOEA/D at the middle stage (c) C-MOEA/D at the later stage



(d) MOEA/D-2WA at the early stage (e) MOEA/D-2WA at the middle stage (f) MOEA/D-2WA at the later stage

Fig. 9. Populations at the early, middle, and later stages of C-MOEA/D, and MOEA/D-2WA on bi-objective C2-DTLZ2 problem.



(a) C-MOEA/D at the early stage (b) C-MOEA/D at the middle stage (c) C-MOEA/D at the later stage

(d) MOEA/D-2WA at the early stage (e) MOEA/D-2WA at the middle stage (f) MOEA/D-2WA at the later stage

Fig. 10. Populations at the early, middle, and later stages of C-MOEA/D, and MOEA/D-2WA on bi-objective C1-DTLZ3 problem.

infeasible weights are gradually removed, the population is guided only by feasible weights. Infeasible solutions move towards feasible parts. Thus the constrained PF with three separate pieces can be found at last by MOEA/D-2WA. By contrast, for CMOEA/D, the strong feasibility selection pressure can miss the promising feasible regions easily and only converge to a subarea of the constrained PF.

Infeasible solutions with good convergence are helpful to get across the infeasible barrier or local feasible regions: as depicted in Fig. 10, the population of CMOEA/D converges to the boundary of $f_1^2 + f_2^2 = 6^2$ at the middle stage, but it is hard to jump over the infeasible band since the CDP that CMOEA/D adopted overemphasizes feasible solutions, even if the convergence of these feasible solutions is much worse than that of some infeasible solutions. Conversely, for MOEA/D-2WA, guided by infeasible weights, a few infeasible solutions with better convergence have the capacity to drag the population to cross the infeasible barrier.

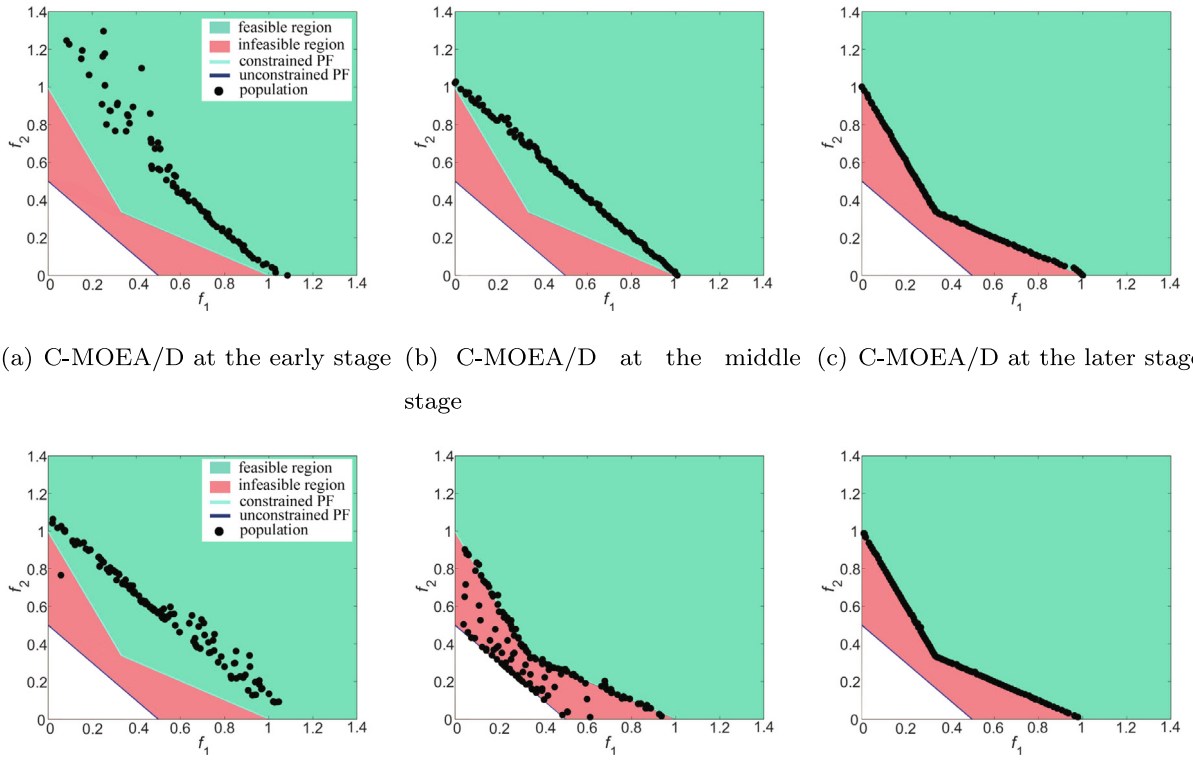
Infeasible solutions are useful to explore the Pareto optimal solutions located on the boundary of the feasible region from the infeasible side: as depicted in Fig. 11, CMOEA/D converges to the constrained PF from the feasible region. For MOEA/D-2WA, a large number of infeasible solutions lying close to the constrained PF, leading the search to converge to the constrained PF from both the feasible and infeasible sides of the search space.

5.2. Parameter analysis

The shrink of the dynamic ε constraint boundary in Eq. (6) adopts the exponential function of the simulated annealing algorithm. It has a crux parameter *cp*, which controls the decreasing trend of the dynamic ε constraint boundary. In this subsection, the sensitivity analysis of *cp* is conducted as follows.

Fig. 12 plots the changes of the dynamic ε constraint boundary with different *cp* values. It can be observed that the dynamic ε constraint boundary shrinks very fast with $cp = 2$. When the value of *cp* increases, the shrinkage speed of the dynamic ε constraint boundary also slows down.

Fig. 13 plots the performance, in terms of IGD, of MOEA/D-2WA with different *cp* values (2, 4, 6, 8, and 10) on the MW4 and MW8 test problems over 30 independent runs, respectively. It can be observed that the optimal value of *cp* is problem-dependent. To elaborate, the performance on the MW4 problem with 5, 8, 10, and 15 objectives does not fluctuate greatly for



(a) C-MOEA/D at the early stage (b) C-MOEA/D at the middle stage (c) C-MOEA/D at the later stage

(d) MOEA/D-2WA at the early stage (e) MOEA/D-2WA at the middle stage (f) MOEA/D-2WA at the later stage

Fig. 11. Populations at the early, middle, and later stages of C-MOEA/D, and MOEA/D-2WA on bi-objective C3-DTLZ1 problem.

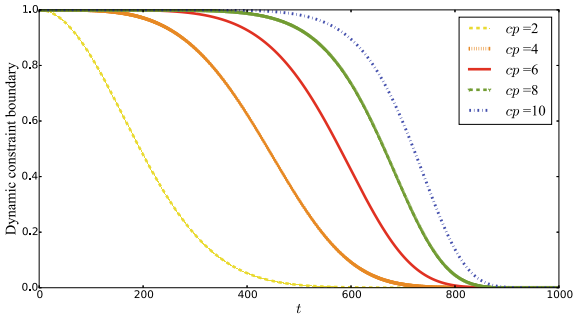


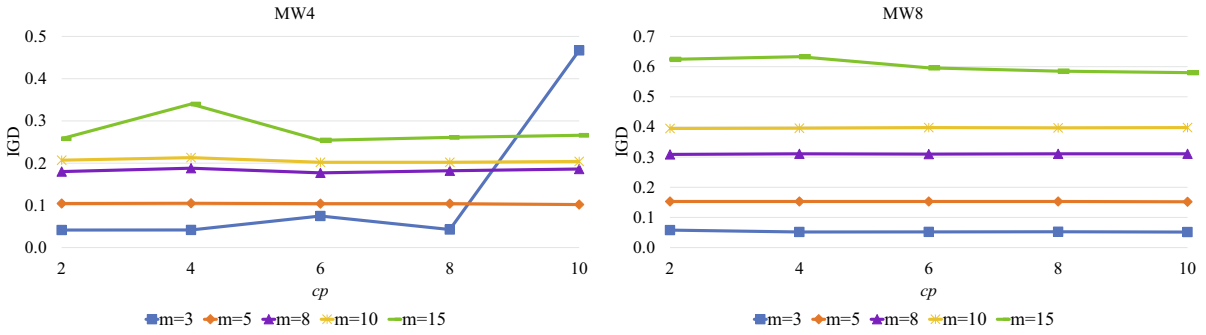
Fig. 12. Changes of the dynamic ε constraint boundary with different cp values ($\varepsilon^{(0)} = 1$ and $maxT$ = 1000).

different cp settings. For the MW4 problem with 3 objectives, the results in Fig. 13(a) suggest that MOEA/D-2WA has poor performance with $cp = 10$. For the MW8 problem, the performance of MOEA/D-2WA with different values of cp in Fig. 13(b) is not sensitive to the setting of cp.

However, in general, MOEA/D-2WA is robust with regard to $cp = 2, 4, 6, 8$. Therefore, cp is in the range of 2 to 8 could be the better choice for most problems.

5.3. Solving a real-world antenna array synthesis problem

In this subsection, the performance of MOEA/D-2WA and its competitors on a real-world antenna array synthesis problem is investigated. The goal of antenna array synthesis is to find the suitable excitation vector and the layout of the elements to produce an ideal radiation pattern. An equally-spaced antenna array synthesis problem [44] is considered, which consists



(a) MW4 (b) MW8

Fig. 13. Average IGD values obtained by MOEA/D-2WA with different values of *cp* on the MW4 and MW8 test problems over 30 runs, respectively.

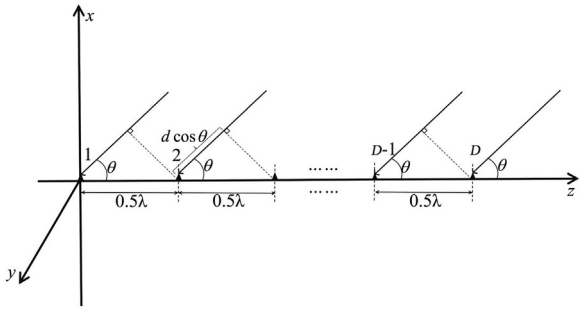


Fig. 14. Geometric of the linear array.

of four objectives: the error between the desired linear array factor and the designed array factor, the maximum sidelobe level (MSLL), the number of sparse array elements, and the dynamic excited voltage range. An inequality constraint is used to generate the desired MSLL. As shown in Fig. 14, this problem consists of 19 array elements distributed on the *z*-axis, each array element has its amplitude excitation and phase excitation, so it contains a total of 38 decision variables.

5.3.1. Problem Formulation

The array synthesis problem can be formulated as

$$\begin{aligned} &\min && \boldsymbol{F}(\boldsymbol{a},\boldsymbol{\beta}) = (e(\boldsymbol{a},\boldsymbol{\beta}), MSLL(\boldsymbol{a},\boldsymbol{\beta}), N_{sp}(\boldsymbol{a}), R(\boldsymbol{a})) \\ &\text{st}: && MSLL(\boldsymbol{a},\boldsymbol{\beta}) \leqslant -18dB \\ &\text{where} && 0 \leqslant a_i \leqslant 1, \\ & && -180^\circ \leqslant \beta_i \leqslant 180^\circ, \\ & && i = 1,\ldots,D. \end{aligned} \tag{16}$$

The first objective is the error between the desired linear array factor and the designed array factor:

$$e(\boldsymbol{a},\boldsymbol{\beta}) = \sqrt{\frac{\sum_{i=1}^{N}[f_i^{(0)} - f(\boldsymbol{a},\boldsymbol{\beta},\theta_i)]^2}{\sum_{i=1}^{N}[f_i^{(0)}]^2}} \tag{17}$$

where $f_i^{(0)}$ is the ideal array factor at the i-th angle, which is calculated as:

$$f^{(0)} = \begin{cases} \text{cosecant}(\cos\theta), \text{if } 0.1 \leqslant \cos\theta \leq 0.5; \\ 0, \quad \text{otherwise}. \end{cases} \tag{18}$$

The linear array factor $f(\boldsymbol{a},\boldsymbol{\beta},\theta)$ is:

$$f(\boldsymbol{a},\boldsymbol{\beta},\theta)=\sum_{i=1}^{D}a_i\cdot e^{j(\frac{2\pi}{\lambda}d\cdot\cos\theta+\beta_i)} \tag{19}$$

where θ $(0^\circ\leqslant\theta\leq 180^\circ)$ is the antenna array direction angle, it is split as $\theta_i\in\{0^\circ,1^\circ,\ldots,180^\circ\},N=181$. $\boldsymbol{a}=(a_1,a_2,\ldots,a_D),\boldsymbol{\beta}=(\beta_1,\beta_2,\ldots,\beta_D),D$=19 is the number of array elements, a_i and β_i are the amplitude excitation and phase excitation of the i-th antenna element, respectively. d represents the distance from the initial position to the i-th array element. In this paper, the distance between two adjacent array elements is a constant, which is equal to 0.5λ, where λ is the wavelength.

The second objective is the MSLL:

$$MSLL(\boldsymbol{a},\boldsymbol{\beta})=20\cdot\log_{10}(\max|\frac{f(\boldsymbol{a},\boldsymbol{\beta},\theta_{SL})}{f(\boldsymbol{a},\boldsymbol{\beta},\theta_{max})}|) \tag{20}$$

where θ_{SL} represents the angle within the sidelobes band except the range of the main lobe. θ_{max} denotes the main lobe angle.

The third objective is used to minimize the number of sparse array elements:

$$N_{sp}(\boldsymbol{a})=\frac{1}{1+SparseNum} \tag{21}$$

where *SparseNum* is the sum of sparse antenna elements. An array element is regarded as a sparse one on condition that its amplitude excitation satisfies $a<1e-4$.

The fourth objective is the dynamic excited voltage range:

$$R(\boldsymbol{a})=\frac{\max(|\boldsymbol{a}|)}{\min(|\boldsymbol{a}|)} \tag{22}$$

where $\max(|\boldsymbol{a}|)$ and $\min(|\boldsymbol{a}|)$ represent the maximal and minimal of the amplitude excitation, respectively.

The constraint is to generate the desired *MSLL* that less than -18 dB.

5.3.2. Synthesis results

To estimate the proportion of the feasible space to the entire search space of this antenna array synthesis problem, this study randomly samples 1,000,000 solutions with a uniform distribution in the decision space. Accordingly, we can obtain the estimated feasible ratio of this problem is approximately 0.0%. Thus it can be concluded that this antenna array synthesis problem belongs to a highly constrained optimization problem.

In our experiment, the population size is set to 120, and the number of maximum function evaluations is 200,000. Three performance metrics were employed to compare different algorithms. Since this array synthesis problem is highly constrained, finding feasible solutions is very challenging for these algorithms. So the success rate (SR) is regarded as the first performance metric, which represents the percentage of the successful runs. A run is considered successful if it can find feasible solutions. The second and third metrics are IGD and HV, respectively. Note that the true PF of this array synthesis problem is unknown, this paper employs the obtained feasible nondominated solutions by seven algorithms in 30 runs as the true PF. Note that the obtained ideal point and nadir point are (0.2766, 0.0465, 0.0556, 1.0116) and (1.6336, 0.5512, 1.0, 18.075), respectively, which are used to normalize each objectives for HV calculation in Eq. (15).

Table 4 summarizes the results of seven algorithms in terms of SR, IGD, and HV, respectively. In terms of SR, C-MOEA/D, I-DBEA, C-TAEA, and MOEA/D-2WA can find feasible solutions consistently. In contrast, PPS can find feasible solutions on 14/30 runs, C-NSGA-III and C-And failed to obtain feasible solutions in all 30 runs. In the light of IGD, C-MOEA/D achieves the best results among seven competitors, followed by MOEA/D-2WA. Based on HV, the proposed MOEA/D-2WA performs best among all compared algorithms.

Overall, the highly competitive synthesis results suggest that MOEA/D-2WA is also suitable for solving real-world antenna array synthesis problems.

6. Conclusions

Weight vectors play a vital role in guiding solutions towards the desired subregions. To fully use both the promising feasible and infeasible solutions, this paper has proposed two-type weight adjustments in MOEA/D, referred to as MOEA/D-2WA, for solving CMaOPs. In MOEA/D-2WA, two-type weights have been constructed, i.e., infeasible weights and feasible weights. Infeasible weights are beneficial to the search mainly reflected in three aspects: 1) infeasible solutions with good convergence are helpful to cross the large, separate, and multi-modal infeasible barrier; 2) infeasible solutions with good diversity are useful to locate multiple feasible subregions; 3) infeasible solutions contribute to explore Pareto optimal solutions seated on constraint boundaries from the infeasible side. During the course of the search, the number of infeasible weights is dynamically reduced, which switches the search from the optimality priority to the feasibility priority. At the later stage of evolution, feasible weights are used to balance the diversity and convergence of the final feasible Pareto optimal solutions.

Table 4
Comparative results of array synthesis problem obtained by C-NSGA-III, C-MOEA/D, I-DBEA, PPS, C-TAEA, C-AnD, and MOEA/D-2WA in terms of SR, IGD, and HV metrics, respectively.

Metric	C-NSGA-III[16]	C-MOEA/D[16]	I-DBEA[1]	PPS[11]	C-TAEA[23]	C-AnD[28]	MOEA/D-2WA
SR	0%	100%	100%	46.67%	100%	0%	100%
IGD	NaN	1.4800e+0 ≈	6.6667e+6 +	NaN	2.8887e+0 +	NaN	1.4999e+0
HV	NaN	3.3882e-2 +	1.5293e-2 +	NaN	3.3271e-2 +	NaN	4.3976e-2

"+", "-", and "≈" represent the performance of MOEA/D-2WA is significantly better than, significantly worse than, and significantly equal to the other peers according to the Wilcoxon's rank sum test at a 0.05 significance level, respectively; "NaN" stands for the corresponding algorithm cannot find feasible solutions in all 30 runs.

Six state-of-the-art CMaOEAs were employed as competitors. A total of 70 test problems with up to 15 objectives from C-DTLZ, DC-DTLZ, and MW test suites are adopted for comparison. It is demonstrated by the experimental results that MOEA/D-2WA is superior overall to six compared peer algorithms. Especially, MOEA/D-2WA is capable of handling highly constrained optimization problems. Besides, the effectiveness of the proposed MOEA/D-2WA was also validated on a real-world antenna array synthesis problem.

In this paper, feasible weights are predefined and evenly distributed in a unit simplex. However, for some problems with the irregular PF such as disconnected or degenerate, the uniformly distributed weights may not work well for decomposition-based MOEAs. Therefore, one possible direction for future research is to investigate the adapt weights setting.

CRediT authorship contribution statement

Ruwang Jiao: Methodology, Writing - original draft. **Sanyou Zeng:** Supervision, Validation, Writing - review & editing. **Changhe Li:** Supervision, Validation, Writing - review & editing. **Yew-Soon Ong:** Supervision, Validation.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This work was supported in part by the National Natural Science Foundation of China under Grant 62076226 and 61673355, in part by the Hubei Provincial Natural Science Foundation of China under Grant 2015CFA010, in part by the 111 project under Grant B17040, in part by the Fundamental Research Funds for National Universities, China University of Geosciences(Wuhan) under Grant CUGGC02, in part by the Data Science and Artificial Intelligence Research (DSAIR) Center at Nanyang Technological University, Singapore.

The source code of this study can be downloaded from https://github.com/RuwangJiao/MOEA-D-2WA.

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at https://doi.org/10.1016/j.ins.2021.07.048.

References

[1] M. Asafuddoula, T. Ray, R. Sarker, A decomposition-based evolutionary algorithm for many objective optimization, IEEE Trans. Evol. Comput. 19 (3) (2014) 445–460.
[2] J. Bader, E. Zitzler, HypE: An algorithm for fast hypervolume-based many-objective optimization, Evol. Comput. 19 (1) (2011) 45–76.
[3] P.A.N. Bosman, D. Thierens, The balance between proximity and diversity in multiobjective evolutionary algorithms, IEEE Trans. Evol. Comput. 7 (2) (2003) 174–188.
[4] R. Cheng, Y. Jin, M. Olhofer, B. Sendhoff, A reference vector guided evolutionary algorithm for many-objective optimization, IEEE Trans. Evol. Comput. 20 (5) (2016) 773–791.
[5] 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.
[6] K. Deb, R.B. Agrawal, et al, Simulated binary crossover for continuous search space, Complex Systems 9 (2) (1995) 115–148.
[7] K. Deb, M. Goyal, A combined genetic adaptive search (geneas) for engineering design, Computer Sci. Inform. 26 (1996) 30–45.
[8] K. Deb, H. Jain, An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints, IEEE Trans. Evol. Comput. 18 (4) (2014) 577–601.
[9] 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.
[10] Z. Fan, Y. Fang, W. Li, X. Cai, C. Wei, E.D. Goodman, MOEA/D with angle-based constrained dominance principle for constrained multi-objective optimization problems, Appl. Soft Computing 74 (2019) 621–633.

[11] 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.
[12] X.-Z. Gao, M.S.R. Nalluri, K. Kannan, D. Sinharoy, Multi-objective optimization of feature selection using hybrid cat swarm optimization, Science China Technol. Sci. 64 (3) (2021) 508–520.
[13] Hossein Gitinavard, Mohsen Akbarpour Shirazi, Seyed Hassan Ghodsypour, A bi-objective multi-echelon supply chain model with pareto optimal points evaluation for perishable products under uncertainty, Scientia Iranica 26 (5) (2019) 2952–2970.
[14] H. Ishibuchi, T. Fukase, N. Masuyama, Y. Nojima, Dual-grid model of MOEA/D for evolutionary constrained multiobjective optimization, in: Proceedings of the Genetic and Evolutionary Computation Conference, ACM, 2018, pp. 665–672.
[15] H. Ishibuchi, T. Murata, A multi-objective genetic local search algorithm and its application to flowshop scheduling, IEEE Trans. Syst., Man, .Cybern., Part C (Appl. Rev.) 28 (3) (1998) 392–403.
[16] 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.
[17] R. Jiao, S. Zeng, C. Li, A feasible-ratio control technique for constrained optimization, Inf. Sci. 502 (2019) 201–217.
[18] R. Jiao, S. Zeng, C. Li, W. Pedrycz, Evolutionary constrained multi-objective optimization using nsga-ii with dynamic constraint handling, in: 2019 IEEE Congress on Evolutionary Computation, IEEE, 2019, pp. 1635–1642.
[19] R. Jiao, S. Zeng, C. Li, S. Yang, Y.-S. Ong, Handling constrained many-objective optimization problems via problem transformation, IEEE Trans. Cybern. (2020), https://doi.org/10.1109/TCYB.2020.3031642.
[20] S. Kirkpatrick, M.P Gelatt, C D.and Vecchi. Optimization by simulated annealing. Science, 220(4598):671–680, 1983.
[21] B. Li, J. Li, K. Tang, X. Yao, Many-objective evolutionary algorithms: A survey, ACM Computing Surveys (CSUR) 48 (1) (2015) 13.
[22] H. Li, Q. Zhang, Multiobjective optimization problems with complicated pareto sets, MOEA/D and NSGA-II, IEEE Trans. Evol. Comput. 13 (2) (2008) 284–302.
[23] K. Li, R. Chen, G. Fu, X. Yao, Two-archive evolutionary algorithm for constrained multiobjective optimization, IEEE Trans. Evol. Comput. 23 (2) (2018) 303–315.
[24] K. Li, K. Deb, Q. Zhang, S. Kwong, An evolutionary many-objective optimization algorithm based on dominance and decomposition, IEEE Trans. Evol. Comput. 19 (5) (2014) 694–716.
[25] C. Liu, Q. Zhao, B. Yan, S. Elsayed, T. Ray, R. Sarker, Adaptive sorting-based evolutionary algorithm for many-objective optimization, IEEE Trans. Evol. Comput. 23 (2) (2018) 247–257.
[26] H.-L. Liu, F. Gu, Q. Zhang, Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems, IEEE Trans. Evol. Comput. 18 (3) (2013) 450–455.
[27] Z.-Z. Liu, Y. Wang, Handling constrained multiobjective optimization problems with constraints in both the decision and objective spaces, IEEE Trans. Evol. Comput. 23 (5) (2019) 870–884.
[28] 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. 509 (2020) 400–419.
[29] Z. Ma, Y. Wang, Evolutionary constrained multiobjective optimization: Test suite construction and performance comparisons, IEEE Trans. Evol. Comput. 23 (6) (2019) 972–986.
[30] 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.
[31] C. Peng, H.-L. Liu, E. Goodman, A cooperative evolutionary framework based on an improved version of directed weight vectors for constrained multiobjective optimization with deceptive constraints, IEEE Trans. Cybern. (2020), to be published.
[32] C. Peng, H.-L. Liu, F. Gu, An evolutionary algorithm with directed weights for constrained multi-objective optimization, Appl. Soft Computing 60 (2017) 613–622.
[33] C. Peng, H.-L. Liu, F. Gu, A novel constraint-handling technique based on dynamic weights for constrained optimization problems, Soft. Comput. 22 (12) (2018) 3919–3935.
[34] H.K. Singh, K. Alam, T. Ray, Use of infeasible solutions during constrained evolutionary search: A short survey, in: Australasian Conference on Artificial Life and Computational Intelligence, Springer, 2016, pp. 193–205.
[35] Y. Tian, R. Cheng, X. Zhang, Y. Jin, PlatEMO: A MATLAB platform for evolutionary multi-objective optimization [educational forum], IEEE Comput. Intell. Mag. 12 (4) (2017) 73–87.
[36] Y. Tian, T. Zhang, J. Xiao, X. Zhang, Y. Jin, A coevolutionary framework for constrained multi-objective optimization problems, IEEE Trans. Evol. Comput. 25 (1) (2021) 102–116.
[37] A. Trivedi, D. Srinivasan, K. Sanyal, A. Ghosh, A survey of multiobjective evolutionary algorithms based on decomposition, IEEE Trans. Evol. Comput. 21 (3) (2016) 440–462.
[38] R. Vakili, M. Akbarpour Shirazi, Hossein Gitinavard, Multi-echelon green open-location-routing problem: A robust-based stochastic optimization approach, Scientia Iranica 28 (2) (2021) 985–1000.
[39] C. Von Lücken, B. Barán, C. Brizuela, A survey on multi-objective evolutionary algorithms for many-objective problems, Comput. Optim. Appl. 58 (3) (2014) 707–756.
[40] B.-C. Wang, H.-X. Li, Q. Zhang, Y. Wang, Decomposition-based multiobjective optimization for constrained evolutionary optimization, IEEE Trans. Syst., Man, Cybern.: Syst. 51 (1) (2021) 574–587.
[41] J. Wang, G. Liang, J. Zhang, Cooperative differential evolution framework for constrained multiobjective optimization, IEEE Trans. Cybern. 49 (6) (2018) 2060–2072.
[42] R. Wang, Q. Zhang, T. Zhang, Decomposition-based algorithms using pareto adaptive scalarizing methods, IEEE Trans. Evol. Comput. 20 (6) (2016) 821–837.
[43] Y. Xiang, Y. Zhou, M. Li, Z. Chen, A vector angle-based evolutionary algorithm for unconstrained many-objective optimization, IEEE Trans. Evol. Comput. 21 (1) (2016) 131–152.
[44] Q. Xu, S. Zeng, F. Zhao, R. Jiao, C. Li, On formulating and designing antenna arrays by evolutionary algorithms, IEEE Trans. Antennas Propag. 69 (2) (2021) 1118–1129.
[45] S. Zeng, R. Jiao, C. Li, X. Li, J.S. Alkasassbeh, A general framework of dynamic constrained multiobjective evolutionary algorithms for constrained optimization, IEEE Transactions on Cybernetics 47 (9) (2017) 2678–2688.
[46] S. Zeng, R. Jiao, C. Li, R. Wang, Constrained optimisation by solving equivalent dynamic loosely-constrained multiobjective optimisation problem, Int. J. Bio-Inspired Comput. 13 (2) (2019) 86–101.
[47] Q. Zhang, H. Li, MOEA/D: A multiobjective evolutionary algorithm based on decomposition, IEEE Trans. Evol. Comput. 11 (6) (2007) 712–731.
[48] Q. Zhang, W. Liu, H. Li. The performance of a new version of MOEA/D on cec09 unconstrained mop test instances. In IEEE Congress on Evolutionary Computation, pages 203–208. IEEE, 2009.
[49] Q. Zhu, Q. Zhang, Q. Lin, A constrained multiobjective evolutionary algorithm with detect-and-escape strategy, IEEE Trans. Evol. Comput. 24 (5) (2021) 938–947.
[50] E. Zitzler, L. Thiele, Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach, IEEE Trans. Evol. Comput. 3 (4) (1999) 257–271.