Information Sciences 579 (2021) 89–102

Contents lists available at ScienceDirect

Information Sciences

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

A dual-population algorithm based on alternative evolution and degeneration for solving constrained multi-objective optimization problems

Juan Zou [a,b,1], Ruiqing Sun [a,b,*,1], Shengxiang Yang [d], Jinhua Zheng [a,b,c]

[a] Key Laboratory of Intelligent Computing and Information Processing, Ministry of Education, School of Computer Science and School of Cyberspace Science of Xiangtan University, Xiangtan, Hunan Province, China
[b] Faculty of Informational Engineering, University of Xiangtan, Xiangtan 411105, China
[c] Hunan Provincial Key Laboratory of Intelligent Information Processing and Application, Hengyang 421002, China
[d] School of Computer Science and Informatics, De Montfort University, Leicester LE1 9BH, UK

ARTICLE INFO

Article history:
Received 28 March 2021
Received in revised form 20 July 2021
Accepted 23 July 2021
Available online 27 July 2021

Keywords:
Constrained multi-objective optimization
Evolutionary algorithm
Dual population

ABSTRACT

It is challenging to solve constrained multi-objective optimization problems (CMOPs). Different from the traditional multi-objective optimization problem, the feasibility, convergence, and diversity of the population must be considered in the optimization process of a CMOP. How these factors are balanced will affect the performance of the constrained multi-objective optimization algorithm. To solve this problem, we propose a dual-population multi-objective optimization evolutionary algorithm. The proposed algorithm can make good use of its secondary population and alternative between evolution and degeneration according to the state of the secondary population to provide better information for the main population. The test results of three benchmark constrained multi-objective optimization problem suites, and four real-world constrained multi-objective optimization problems show that the algorithm is better than existing dual-population multi-objective optimization, especially when there is a distance between the unconstrained PF and the constrained PF.

1. Introduction

There are many multi-objective optimization problems in the real world. These multi-objective optimization problems may have one or more equality or inequality constraints [1]. Such problems are called constrained multi-objective optimization problems (CMOPs) [2] and are used for various purposes, including gait optimization of humanoid robots [3], allocation of web services [4], parallel machine scheduling [5], and lightweight design of automobiles [6]. Without loss of generality, CMOPs can be defined as follows [7]:

* Corresponding author.
E-mail address: sunny0331@foxmail.com (R. Sun).
[1] The first two authors contributed equally to this work.

https://doi.org/10.1016/j.ins.2021.07.078
0020-0255/

$$\begin{cases} min\ F(X) = (f_1(X), f_2(X), \ldots, f_m(X)), \\ subject\ to\ X \in \Omega \\ \qquad\qquad g_i(x) \le 0,\ i = 1, \ldots, p \\ \qquad\qquad h_j(x) = 0,\ j = 1, \ldots, q \end{cases} \tag{1}$$

where $X = (x_1, x_2, \ldots, x_n)$ is a D-dimensional decision variable vector from the decision space Ω, and $F(X)$ is an objective function vector that consists of m conflicting objective functions. Additionally, it has p inequality constraints $g_i(x)$ and q equality constraints $h_j(x)$. For a CMOP, the degree of violation of the jth constraint can be defined using $c_j(x)$ [8]:

$$c_j(x) = \begin{cases} max(0, g_j(x)), & j = 1, \ldots, p \\ max(0, |h_j(x) - \delta|), & j = p+1, \ldots, p+q \end{cases} \tag{2}$$

where δ is a small enough relaxation factor (usually taken as 1e-6) to relax equality constraints [9] since equality constraints are very difficult to satisfy. The solution of a CMOP usually sums the violation degree of each constraint to judge its feasibility:

$$CV(x) = \sum_{j=1}^{p+q} c_j(x), \tag{3}$$

If the $CV(x)$ is equal to 0, then the solution is feasible; otherwise, it is infeasible. The larger the $CV(x)$ value, the greater the violation degree of the solution to the overall constraint. The solution space satisfying all constraints is called the feasible region; otherwise, it is called the infeasible region.

For any two solutions $x_a, x_b \in \Omega$, the solution x_a pareto dominates [10] x_b if and only if $f_i(x_a) \leqslant f_i(x_b)$ for each $i \in \{1, \ldots, m\}$and $f_j(x_a) < f_j(x_b)$ for at least one $j \in \{1, \ldots, m\}$, denoted as $x_a \prec x_b$. If there are no mutually dominating solutions in a set, this set constitutes a Pareto optimal set. The mapping of all Pareto optimal sets in the objective space constitutes the unconstrained PF, and the mapping of all Pareto optimal sets satisfying the constraints in the objective space is called the constrained PF. The process of an algorithm to solving a CMOP is to converge to the constrained Pareto Front and obtain an ideal constrained PF approximation [11].

Influenced by the evolution process of natural organisms, an evolutionary algorithm is a random search algorithm based on population [12]. After nearly two decades of development, evolutionary algorithms have become a mature global optimization method with high robustness and broad applicability [13]. At present, multi-objective evolutionary algorithms can be roughly divided into three categories [14]: the first is based on Pareto dominated algorithms, represented by algorithms such as NSGAII [15]; the second is based on decomposition, represented by MOEA/D [16]; the third is based on indicators, represented by SMS-EMOA [17].Compared with traditional multi-objective optimization algorithms, few constrained multi-objective optimization algorithms (CMOEAs) exist, and the gap in this field is attracting more attention from researchers [18].

In the process of solving CMOPs, the balance of *convergence*, *diversity*, and *feasibility* of the population is crucial because it determines the effectiveness of a CMOEA. In existing dual-population CMOEAs, the main population usually evolves with all constraints and mainly considers the *feasibility* of three elements. The secondary population usually evolves without considering any constraints but considers the *diversity* and *convergence* of three elements. In a dual-population algorithm (actually in all CMOEAs), whether the infeasible objective space can be used reasonably is very important because it will influence the diversity and convergence of the main population. Based on this fact, this paper proposes a dual-population CMOEA to solve CMOPs. The main advantages of this algorithm are as follows:

1) The proposed algorithm has a higher utilization of its secondary population, which means the feasible regions are explored more carefully, and it is easier to find a small feasible region. Therefore, the diversity and convergence of the main population are improved.
2) The proposed algorithm has a dynamic cooperation intensity between two populations rather than constant. Two populations will enhance cooperation at the appropriate time to enhance the searchability of the objective space.

The remainder of this paper is organized as follows: Section 2 reviews several constraint handling techniques and the classification of existing CMOPs. The details of our algorithm(we called CAEAD in short) are presented in Section 3. To evaluate the performance of our algorithm, the results of a series of experiments are presented and analyzed in Section 4. Finally, Section 5 concludes this paper.

2. Related works and motivation

In this section, the existing CMOEAs and several constraint handling techniques used in our algorithm are reviewed. We also divide CMOPs into three categories according to their unconstrained and constrained Pareto fronts. It is the classification of CMOPs that provides motivation for the following work.

2.1. Existing CMOEAs with constraint handling techniques

CMOPs have two parts. One part consists of objective functions, which determine the convergence and diversity of the population, and the other part is the constraint, which determines the feasibility of the population.

Fonseca and Flemming [19] said that the constraint has a higher priority in the early constrained multi-objective framework than the objective function. The algorithm in [20] directly discards the infeasible solution. The result is a loss of selection pressure, which leads to a population that may not converge to the PF.

The Constraint Domination Principle (CDP) [15] is one of the most commonly used constraint handling techniques that has evolved from the feasibility principle of constraint single object optimization [21]. In CDP rules, the traditional dominant rules are changed, and solution x_a is said to constraint dominate another solution x_b if it satisfies one of the following rules:

$$x_a \prec x_b \Rightarrow \begin{cases} x_a \text{ is feasible; } \mathrm{x_b} \text{ is infeasible} \\ \text{both } \mathrm{x_a} \text{ and } \mathrm{x_b} \text{ are feasible, and } \mathrm{x_a} \prec \mathrm{x_b} \\ \text{both } \mathrm{x_a} \text{ and } \mathrm{x_b} \text{ are infeasible, and } \mathrm{CV(x_a)} < \mathrm{CV(x_b)} \end{cases} \tag{4}$$

In the CDP method, the constraint is always higher than the objective value, saving the solution in the feasible region as much as possible. Wang et al. [22] divided the population into m sub-populations; each sub-population focused on only one objective value and used CDP for optimization. In [23], Zeng et al. applied CDP to an orthogonal multi-objective evolutionary algorithm framework. In most cases, the CDP is available and effective, but when there is a sizeable infeasible region, CDP rules will restrict the ability of the population to cross the infeasible region, thereby affecting the convergence of the population. The CDP method has been improved [24]. When the CV is less than the set threshold, the infeasible solution is regarded as a feasible solution. An angle-based CDP is used in [25]. When two individuals are not feasible, and the included angle is less than the threshold θ, the individual with less constraint violation is always better than the larger individual. When the included angle is greater than the threshold θ and the ratio of feasible solutions is less than the threshold r, the dominating individual is always better than the dominated individual; otherwise, the two individuals will not be compared. In [26], individuals determine the search radius by angle and l_p-norm, and individuals with a larger search radius have the priority to be selected in the next generation.

The Epsilon constraint handling approach [27] obtains a good solution by relaxing the constraint violation value.

$$x_a \prec x_b \Rightarrow \begin{cases} x_a \prec x_b, & \text{if } \mathrm{CV(x_a), CV(x_b)} \leqslant \varepsilon \\ \mathrm{x_a} \prec \mathrm{x_b}, & \text{if } \mathrm{CV(x_a) = CV(x_b)} \\ \mathrm{CV(x_a)} < \mathrm{CV(x_b)}, & \text{Otherwise} \end{cases} \tag{5}$$

When both individuals have the same constraint violation value, the approach compares them with their objective values. When the constraint violation values of two individuals are less than Epsilon, the two individuals are regarded as feasible solutions and compared with the traditional dominance relationship. When both individuals are not feasible solutions, the individual with the smallest constraint violation value always dominates the individual with the largest constraint violation value. The method [27] of adjusting Epsilon value is as follows:

$$\varepsilon(k) = \begin{cases} \varepsilon(0)(1 - \frac{k}{Tc})^{cp}, & 0 < k < T_c, \varepsilon(0) = \phi(x^{\theta}), \\ 0, & k \geqslant T_c, \end{cases} \tag{6}$$

where x_θ is the sum of the individuals with the largest violation values of the top θ constraint in the initial population. In the beginning, the Epsilon value is set to be large enough to let the population evolve without considering any constraints. With the evolution process, the Epsilon value gradually decreases and becomes 0 in the T_c generation. When the Epsilon value changes to 0, the Epsilon method is the same as the CDP method. Priority is given to constraint violations. The original Epsilon method does not perform well in dealing with large infeasible regions because the Epsilon value continuously decreases with evolution generations. In order to solve this situation, Fan et al. proposed the IEpsilon method [28]. When the proportion of feasible solutions in the population is greater than a certain threshold, the Epsilon value will increase suddenly, helping the population cross the infeasible region. Therefore, the key of the Epsilon method is knowing how to update the Epsilon value.

Using multiple populations to balance objective values and constraints is also one of the ways to solve CMOEAs. In [29], the population is divided into two sub-populations, one population is more inclined to optimize the feasibility, and the other population is more inclined to optimize the objective value. In [30], each constraint is optimized simultaneously by different sub-populations. In [31], only the objective value is considered in the optimization of population DA, while the objective value and constraints are both considered in the optimization of population CA, which is regarded as the final population output. In [32], the algorithm uses weak cooperation between two populations, only exchanging individuals on environmental selection.

Researchers also divide the evolutionary process into different stages and use different strategies at different stages. In [14], when the proportion of feasible solutions of a population is less than λ, the population considers the objective value and constraint equally important in evolution. When feasible solutions are greater than λ, the constraint is considered more important than the objective value. ToP [33] divides the whole optimization process into two stages, the first stage optimizes

a single objective with constraints, and the second stage optimizes all objectives with constraints. In [34], a detect-and-escape strategy is designed. Which can help a population jump out of a local optimum when it is trapped in a local optimum. In [35], the algorithm adds constraints one by one at different stages of evolution.

2.2. Motivation of this work

Inspired by [36], the existing CMOPs can be divided into three types according to the relationship between the unconstrained and constrained PFs. Fig. 1 shows the schematic diagram of the three CMOPs.

Type I CMOPs have the same unconstrained and constrained PFs. Since the secondary population of the existing dual-population algorithm (e.g., CTAEA [31], CCMO [32]) does not consider any constraint violations, it can quickly converge to the unconstrained PF, thus leading the main population to it as well. Therefore, existing dual-population CMOEAs perform very well in type I CMOPs.

In type II CMOPs, the unconstrained PF is a part of the constrained PF, and the constrained PF can also be composed of some feasible region boundaries. Type II CMOPs are more complex than type I CMOPs. Because of these reasons, CTAEA and CCMO can easily obtain the overlapped part of the PFs, but it is slightly tricky for the non-coincidence part. Therefore, the performance of the existing dual-population algorithm is good.

In the third type of CMOP, the unconstrained PF and the constrained PF are completely non-coincident, and the constrained PF is wholly composed of feasible region boundaries. In type III CMOPs, the secondary population can help the main population across the infeasible region and provide some information when it passes by the constrained PF. In the existing dual-population CMOEAs, the secondary population takes no constraint during the evolution process, which means the secondary population is always at the unconstrained PF after reaching the unconstrained PF. In this case, the offspring generated by the secondary population often have a certain distance from the constrained PF. The larger the distance, the worse the algorithm's performance since offspring are often generated near its population. Therefore, the performance of the existing dual-population algorithm is not good enough since the secondary population wastes the evaluation times.

3. Proposed algorithm

3.1. Procedure of CAEAD

Algorithm 1: The framework of CAEAD

Input: Population size: N, Objectives: M, The termination criterion;
Output: Final population P

$Pop1 \leftarrow$ Random generation population with size N;
$Pop2 \leftarrow$ Random generation population with size N;
ε_0=inf, $stage$ = $Evolution$;
while *termination criterion is not satisfied* **do**
  $Parent1 \leftarrow$ Select N individual by Tournament selection from $Pop1$;
  $Parent2 \leftarrow$ Select N individual by Tournament selection from $Pop2$;
  $Off1 \leftarrow$ Generate offspring according to $Parent1$;
  $Off2 \leftarrow$ Generate offspring according to $Parent2$;
  $obj_k \leftarrow$ Sum of overall objective values of k generation in $Pop2$;
  $\delta = |obj_k - obj_{k-1}|$;
  $Nc \leftarrow$ Proportion of nondominated solutions in $Pop2$;
  if $\delta < \epsilon$ *and* $Nc == 1$ *and* $stage == Evolution$ **then**
    ε_k = Maximum CV value in $Pop2$;
    $stage$ = $Degeneration$;
  end if
  $Off3 = \varnothing$;
  if $stage == Degeneration$ **then**
    $Off3 \leftarrow$ Generate offspring according to $Parent1$ and $Parent2$;
    $stage$, ε_k = $UpdateEpsilon(\tau, \varepsilon_k, \varepsilon_{max})$;
  end if
  $Pop1 \leftarrow Pop1 \cup Off1 \cup Off2 \cup Off3$;
  $Pop2 \leftarrow Pop2 \cup Off2$;
  $Pop1 \leftarrow$ Select N individuals from $Pop1$ using EnviromentalSelection with CDP method;
  $Pop2 \leftarrow$ Select N individuals from $Pop2$ using EnviromentalSelection with ε method;
end while
Return $Pop1$;

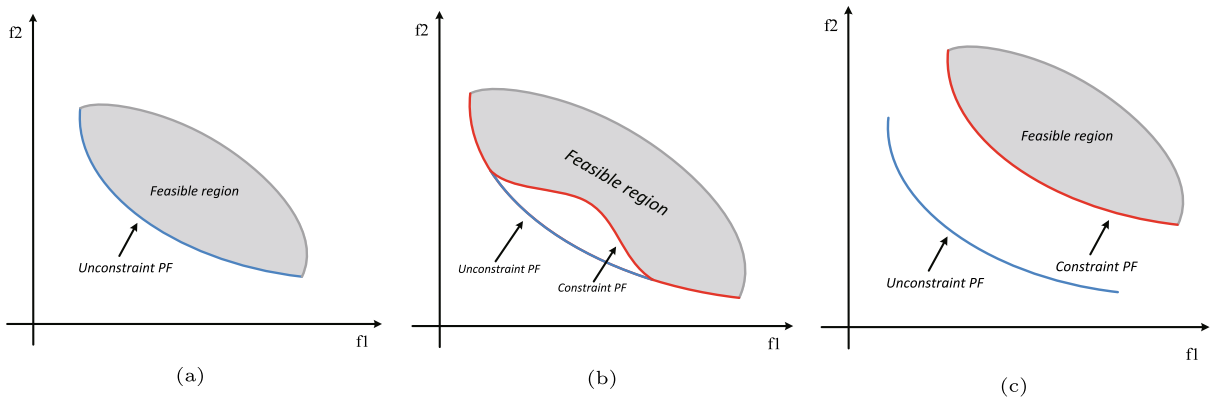

Fig. 1. Classification of CMOPs. (a) Type I. (b) Type II. (c) Type III.

As shown in Algorithm 1, at the beginning of CAEAD, *Pop*1 and *Pop*2 are initialized randomly, and then *Parent*1 and *Parent*2 are selected from *Pop*1 and *Pop*2, respectively, according to the nondominated sort and crowding distance, and *Parent*1 and *Parent*2*t* are used to generate *Off* 1 and *Off* 2, respectively. In this step, the two populations do not interact. Then we need to detect whether *Pop*2 has reached the unconstrained PF. If the difference between the sum of all individual objective values of the current generation and that of the previous generation is less than the set threshold value (1e-2 in this paper), there is no dominant solution in *Pop*2, and this is the evolution stage. *Pop*2 can be regarded as reaching the unconstrained PF since in the evolution stage the Epsilon value is always constant. When *Pop*2 reaches the unconstrained PF, the algorithm sets the Epsilon value as the constraint violation value of the individual with the largest constraint violation value in *Pop*2. When *Pop*2 has converged to the unconstrained PF, the algorithm will gradually reduce the Epsilon value. Through this operation, *Pop*2 will slowly degenerate. At the same time, we use *Parent*1 and *Parent*2 to jointly generate a *Off* 3. Since the *Off* 3 is produced by *Parent*1 and *Parent*2, most of the position of *Off* 3 will be between *Parent*1 and *Parent*2. The purpose of this procedure is to search for a small feasible region near *Pop*1.

In this paper, the environmental selection of the two populations is based on dominance. Generally, the environmental selection can be replaced by the decomposition-based and index-based environmental selection because only the individuals who meet the constraints and have good convergence and diversity are selected. *Pop*1 uses the environmental selection with the CDP. When selecting a new *Pop*1, the individuals with better convergence and diversity will be selected from all the individuals satisfying the constraint. If there are two individuals that do not meet the constraint, the individual with less constraint violation will be selected; *Pop*2 uses the environmental selection with the Epsilon method to select the new *Pop*2. When the constraint violation value is less than Epsilon, the individuals are always regarded as feasible solutions, and then the same rules as the CDP are used to select individuals.

Algorithm 2: UpdateEpsilon

Input: reduction ratio of ε : τ,ε value before updated: ε_k,the maximum ε value until now expect inf:ε_{max};
Output: *stage*, the updated ε value: *result*;
1: **if** $\varepsilon_k < \varepsilon_{threshold}$ **then**
2: *result* = $(1\text{-}\tau)^*\varepsilon_k$;
3: *stage* = *Degeneration*;
4: **else**
5: *result* = ε_{max};
6: *stage* = *Evolution*;
7: **end if**
8: return *stage*, *result*

Algorithm 2 describes the procedure of updating the Epsilon value and the *Pop*2 state. At the beginning of Algorithm 1, Algorithm 1 does not call Algorithm 2 to update Epsilon value and the *Pop*2 state. At this time, the Epsilon value is set to infinity, so *Pop*2 can ignore the constraint and evolution at the beginning so that *Pop*2 can quickly reach the unconstrained PF. When *Pop*2 reaches the unconstrained PF, Algorithm 1 uses Algorithm 2 to update the Epsilon value and *stage*. At this time, the CV value of the individual with the maximum CV value is set to the current Epsilon value; the state of *Pop*2 is set to degenerate, and all individuals of *Pop*2 are still regarded as feasible solutions. Then the Epsilon value gradually

decreases, and the whole *Pop*2 will slowly and gently degenerate from the unconstrained PF to the feasible region. When the decrease of the Epsilon value is smaller than the threshold value (in this paper 1e-4), *Pop*2 has considered all constraints. Additionally, Algorithm 2 will instantly increase the Epsilon value to the maximum Epsilon value until now (expect infinity). At this time, *Pop*2 returns to the evolutionary stage. When *Pop*2 reaches the unconstrained PF again, the Epsilon value is reduced again, and *Pop*2 enters the degradation stage again to the end of the whole algorithm.

3.2. Analysis of CAEAD

In this section, we show the effectiveness of our algorithm by comparing CAEAD with CTAEA and CCMO.

Fig. 2 shows the early generation of CTAEA, CCMO, and CAEAD on LIR-CMOP3. LIR-CMOP3 has ten discontinuous and small feasible regions far away from the unconstrained PF. It is challenging to deal with LIR-CMOP3 well. CTAEA has good distribution in the early generation. The feasible region of LIR-CMOP3 is very narrow, so the proportion of feasible solutions in early *CA* is far less than N. At this time, *CA* will prefer to select the individuals with smaller constraint violation values and better aggregation values from all infeasible individuals, that is, in this case, the constraint and object values are given the same priority. As a result, the new generated *CA* will still have many infeasible individuals, which will affect the convergence of the next generation's solution to the feasible region. For CCMO, because *Population*2 does not consider any constraints and interacts on environmental selection, *Population*2 has better or the same convergence than *Population*1. *Population*2 can quickly converge to the unconstrained PF, and *Population*1 also selects the offspring of *Population*2 in environmental selection, so *Population*2 can help *Population*1 reach the feasible region quickly. For CAEAD, since the initial epsilon value is infinity, *Pop*2 does not consider any constraints, so its early stage is the same as CCMO. *Pop*2 helps *Pop*1 reach the feasible region quickly and then converges to the unconstrained PF.

As shown in Fig. 3, considering both constraint violation and convergence, CTAEA converges to the part of the feasible region in mid-generation, and most of the solutions in *CA* are still not feasible. Due to the updating mechanism of *DA*, when there are more individuals in *CA* in a certain subregion, fewer individuals will choose the next generation *DA*. Therefore, *CA* and *DA* are in different subregions. Most of the parents in CTAEA are generated as *parent*1 from *CA* and as *parent*2 in *DA*. Therefore, most of the offspring are located between *CA* and *DA*. However, *CA* and *DA* are far away from the feasible region. The offspring of CTAEA do not have both good convergence and good diversity, nor good feasibility. The results affect the next generations of *CA* and *DA*. It can be seen from the Fig. 3 that in the medium generation, most of the *offspring*2 generated by *Population*2 in CCMO are near *Population*2, that is, near the unconstrained PF. However, the unconstrained PF is far away from the feasible region, which results in the generation of *population*2 with no effect and wasted evaluation times. However, for *Population*1 of CCMO, although *Population*1 has obtained some feasible regions, it has no way to obtain all ten feasible regions because *Population*1 falls into local optimum. For CAEAD, *Pop*1 falls into the local optimum too, but due to the gradual decrease of the Epsilon value, *Pop*2 will gradually degenerate from the unconstrained PF to the near feasible region. Since *Off*2 always surrounds *Pop*2, and *Off*3 generated based on *Pop*1 and *Pop*2 is between *Pop*1 and *Pop*2, *Off*2 and *Off*3 will help *Pop*1 break away from local optimum when *Pop*2 degenerates near the feasible region. Then a better diversity solution is obtained.

As shown in Fig. 4, in the end, CTAEA only obtains a few feasible solutions, and most of the individuals in *CA* are in the infeasible region. The reason for this result is that *DA* is on the unconstrained PF, while *CA* straddles between the unconstrained PF and the constrained PF, and its offspring are between *CA* and *DA*. These two populations and their offspring cannot provide effective information for the next generation *CA*, resulting in that the next generation *CA* cannot approach the constrained PF, and then cannot obtain the feasible solution. CCMO did not make any changes to the end of the generation. After the *Population*2 converged to the unconstrained PF, it was always on the unconstrained PF, which could not provide effective information for *Population*1. However, *Population*1 fell into the local optimum and could not jump out, resulting in *Population*1 not obtaining all ten feasible regions. At the end of CAEAD, we can see that it has obtained all the solutions of 10 feasible regions due to *Pop*2's degradation mechanism and interaction at appropriate times (i.e., *Off*3 is generated during degradation). It helps *Pop*1 to get better solutions in the feasible region and helps *Pop*1 jump out of local optimum, which provides effective diversity information for *Pop*1.

Fig. 5 shows the distribution of *Off*2 and *Off*3 in the type III test problem. We can see that at the end of the evolution stage, when *Pop*2 is in the unconstrained PF, most *Off*2 are in the unconstrained PF, and all *Off*2 are outside the feasible region. At this time, the generated *Off*2 is not helpful to *Pop*1. When *Pop*2 degenerates near the feasible region, we can see that most of *Off*2 and *Off*3 are near the feasible region, and the proportion of feasible solutions increases significantly, so we can prove that our idea is compelling.

4. Experimental studies

All the experiments in this paper are based on PlatEMO2.9 [37]. The default parameters in the platform are used where there is no particular explanation.

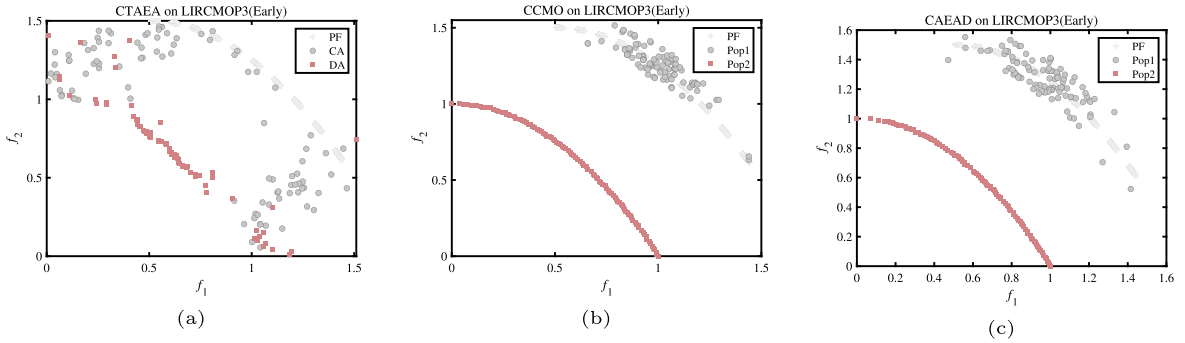

Fig. 2. Three dual-population CMOEAs on LIRCMOP3 in early generation. (a) CTAEA. (b) CCMO. (c) CAEAD.

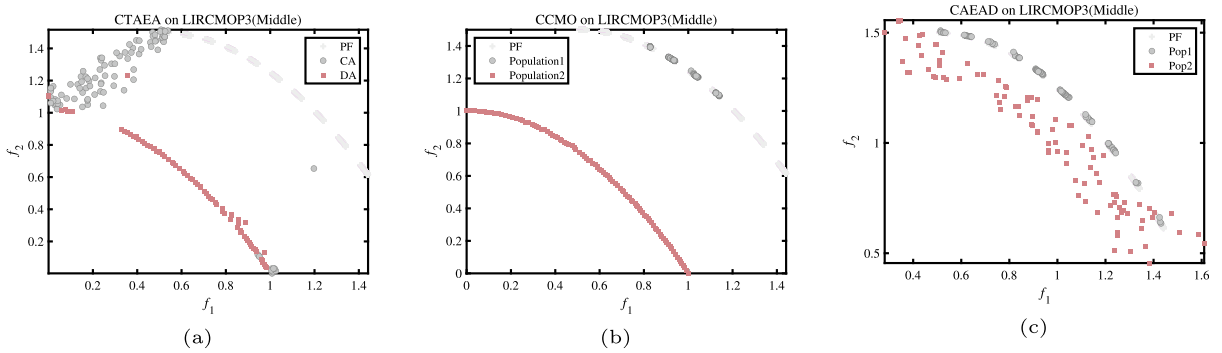

Fig. 3. Three dual-population CMOEAs on LIRCMOP3 in middle generation. (a) CTAEA. (b) CCMO. (c) CAEAD.

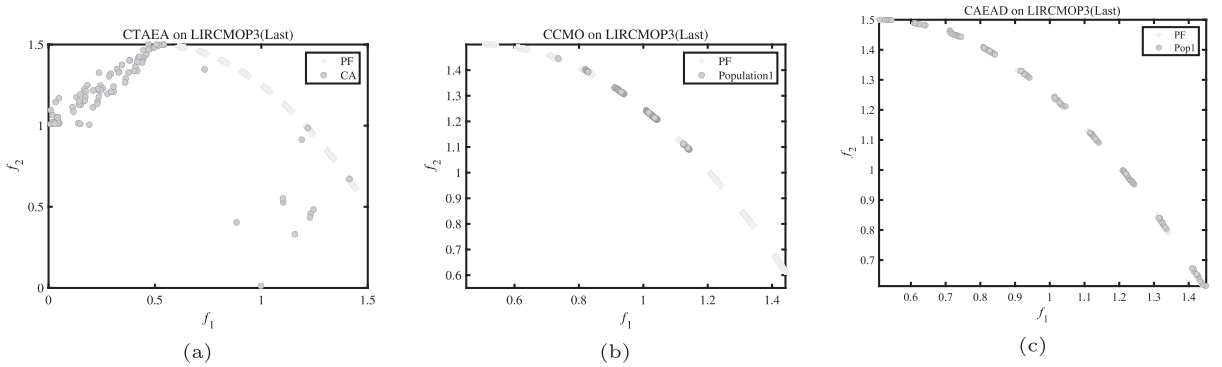

Fig. 4. Three dual-population CMOEAs on LIRCMOP3 in last generation. (a) CTAEA. (b) CCMO. (c) CAEAD.

4.1. Benchmark suites and real-world CMOPs

We used DASCMOP [38] test suite, DOC [39] test suite and LIRCMOP [28] test suite to test our algorithm. Their objective number M, decision vector D and evaluation times FEs are as follows: for all DASCMOP problems, $D = 30$, $FEs = 300{,}000$. For DASCMOP1-DASCMOP6, $M = 2$; for DASCMOP7-DASCMOP9, $M = 3$; for the nine DOC problems, $FEs = 300{,}000$; $M = 3$ for DOC8 and DOC9; $M = 2$ for the remaining problems, and D is fixed to different values for different problems according to [39]. For the 14 LIRCMOP problems, $FEs = 300{,}000$, $D = 10$; for LIRCMOP13 and LIRCMOP14 $M = 3$; for the remaining problems $M = 2$.

For the real-world problems, we adopted four problems to compare our algorithms. They are heat exchange network design problem [40], process synthesis problem [41], synchronous optimal pulse width modulation of 5-level and 3-level inverter problems [42,43]. We set 300,000 evaluations for all of the real-world CMOPs. The purpose of the heat exchange system network problem [40] is to design a system with better economic objectives such as the number and area of heat exchange units and noneconomic objectives such as environmental impact. The process synthesis problem [41] aims to

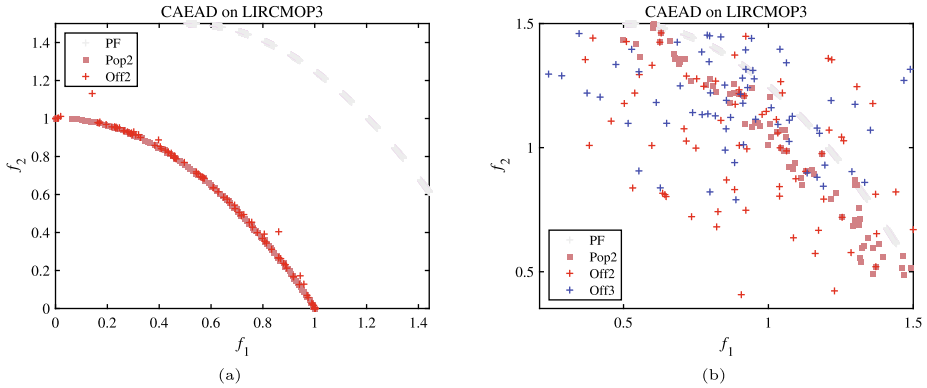

Fig. 5. The offspring distribution of CAEAD. (a) evolution stage. (b) degeneration stage.

reduce the computational effort required to solve the mixed-integer nonlinear programming optimization problem and reduce the effect that nonconvexities can have in cutting off the global optimum, which could be used in the synthesis of the HDA toluene process. Synchronous optimal pulse width modulation [42,43] is an emerging strategy for the control of medium-voltage drives. It allows a considerable reduction of switching frequency without increasing the distortion. The reduction in switching frequency reduces the switching losses and thus increases the efficiency of the inverter.

4.2. CMOEAs used for comparisons

We used two dual-population algorithms (i.e., CTAEA [31], CCMO [32]), two multistage algorithms (i.e., MSCMO [35], MOEAD-DAE [34]) and two classic CMOEAs (i.e., NSGAII-CDP [15], CMOEAD [44]) to compare our algorithm. Their parameter settings were the same as those proposed in their papers. For CCMO, MSCMO, and the algorithm proposed in this respective papers, differential evolution (DE) [45] was used to generate offspring, while NSGAII-CDP, CMOEAD, MOEAD-DAE, and CTAEA used simulated binary crossover [46] and polynomial mutation [21] to generate offspring. When the binary crossover of probability simulation was set to 1, the probability of polynomial mutation was set to $1/D$ (D represents the number of decision variables). The distribution index of crossover and mutation was set to 20; CR differential evolution and F parameter were set to 1 and 0.5, respectively. All population numbers were set to 100.

4.3. Performance metrics

Inverted General Distance (IGD) [47] is a comprehensive performance evaluation metric. It is improved from General Distance (GD) [48]. It can evaluate the convergence and diversity performance of the algorithm simultaneously by calculating the minimum distance sum between each individual on the real PF and the individual set obtained by the algorithm. The smaller the IGD, the better the comprehensive performance, including convergence and diversity. The IGD is calculated as follows:

$$IGD(P,Q)=\frac{\sum_{v\in P}d(v,Q)}{|P|}, \tag{7}$$

where P is the point set uniformly distributed on the real PF, and $|P|$ is the number of individuals of the point set distributed on the real PF. Q is the optimal Pareto optimal solution set obtained by the algorithm. $D(v,Q)$ is the minimum Euclidean distance from individual v to population Q in P. Therefore, IGD evaluates the comprehensive performance of the algorithm by calculating the average value of the minimum distance between the point set on the real PF and the obtained population.

Suppose that $R=(r_1,r_2,\ldots,r_m)$ is a vector (called reference point) in the objective that can dominate the constrained PF. The hypervolume(HV) [49] measures the super volume of the objective space bounded by R and dominated by P:

$$HV=vol(\bigcup_{i\in P}[f_1(i),r_1]\times\cdots[f_m(i),r_m]), \tag{8}$$

where $vol(\cdots)$ represents a Lebesgue metric of Supervolume. Similar to IGD, the HV index can also measure the diversity and convergence of a constrained PF approximation. The higher the value of HV, the higher the approximation degree of P.

For the benchmark problems, we use the IGD metric to assess the performance of all the CMOEAs since their true PFs are known. For the real-world CMOPs, we use the HV to assess the performance of all the CMOEAs since their true PFs are unknown.

4.4. Experimental results

Table 1 shows the comparison of IGD mean and standard deviation of 30 independent operations on LIRCMOP, DASCMOP, and DOC by our algorithm, two double population algorithms, and two classic algorithms. The IGD value of each problem was calculated according to the method recommended in [37], which is based on about 10,000 reference points sampled on the problem PF. We also used the Wilcoxon rank test [50] with a significance level of 0.05 to analyze the results. Among them '+', '−' and '≈' respectively showed that the results of CMOEA were significantly better, significantly worse, and statistically similar to the results of the proposed CAEAD.

4.4.1. Comparison on DASCMOPs

DASCMOP is a CMOP test suite that can define the difficulty of feasibility, diversity, and convergence. It uses η, ζ, and γ to measure the diversity difficulty, feasibility difficulty, and convergence difficulty of the test function, and finally expresses it with a triple. For example, the triplet on DASCMOP1 is [0.5, 0.5, 0.5], which means that there is a diversity difficulty, feasibility difficulty, and convergence difficulty of 0.5 on DASCMOP1, and the algorithm applied must have a good diversity preserving strategy to obtain good results. Our CAEAD algorithm performs best on seven DASCMOP test problems; the MOEAD-DAE performs best on two DASCMOP test problems, because most DASCMOP test problems belong to the type III test problems mentioned in Section 2.2. Our algorithm has a degradation mechanism in dealing with the type III test problems. *Pop*2 can provide *Pop*1 with more diverse solutions, so we finally get better results.

From Fig. 6, we can see that CAEAD obtains all ten discontinuous PF solutions, and the diversity is good. Although CCMO and MSCMO also obtain 10 discontinuous PF solutions, the diversity of solutions in the top discontinuous region is not good enough. In this result, *Population*2 does not provide solutions after reaching the unconstrained PF and wastes evaluation times. The CTAEA, NSGAII-CDP, MOEAD-DAE, and CMOEAD, only obtained partial discontinuous PF solutions.

Fig. 7 shows the performance of each algorithm on DASCMOP7. We can see that all the algorithms converge to the constrained PF, but only CAEAD and CMOEAD have good diversity. The reason for this result is that the *Pop*2 of CAEAD can repeatedly search between the unconstrained PF and constrained PF. When *Pop*2 returns to the constrained PF, it can provide better diversity for *Pop*1. CMOEAD also has good diversity because CMOEAD is a vector-based algorithm, but from the IGD value, CAEAD is better than CMOEAD.

4.4.2. Comparison on DOCs

The DOC test suite is challenging. Various decision constraints (such as inequality constraints, equality constraints, linear constraints, and nonlinear constraints) are constraints, linear constraints, and nonlinear constraints with different properties (such as nonlinear, minimal, and multimodal) in the decision space. At the same time, some controllable and straightforward object constraints are designed to reduce the feasible region in the objective space, and the PF has many characteristics, such as being continuous, discrete, mixed, or degenerate. As shown in Fig. 8, from the perspective of convergence and feasibility, only MSCMO, CCMO, and CAEAD converge to the constrained PF, while others do not. From the perspective of distribution, MSCMO and CAEAD are slightly better than CCMO.

4.4.3. Comparison on LIRCMOPs

The LIRCMOP test problem has many different characteristics. For LIRCMOP-4, the constrained PF is discontinuous and far away from the unconstrained PF, and the feasible region is only a discontinuous line segment. LIRCMOP-8 will encounter a sizeable infeasible region before converging to the constrained PF, which is very difficult to cross and easily causes local convergence. For LIRCMOP9-LIRCMOP12, the feasible regions are discontinuous, which requires the feasibility of the algorithm. LIRCMOP13-LIRCMOP14 have a large infeasible region in the three-dimensional objective space, which means that the algorithm has difficulty converging to the constrained PF. The results show, our CAEAD algorithm performed best on seven test problems in the LIRCMOP test set, in four test problems in MSCMO, and one in MOEAD-DAE. CMOEAD performed best on two test problems.

It can be seen from Fig. 9 that CAEAD obtained all ten discontinuous constrained PFs. MSCMO, CCMO, NSGAII-CDP, and CMOEAD of LIRCMOP4 fell into the local optimum, and only part of the constrained PF was obtained. Most of the solutions in the infeasible region, and almost no feasible solutions, were obtained by CTAEA since *CA* considers the convergence and feasibility simultaneously. For the MOEAD-DAE, no feasible solution was obtained on LIRCMOP4.

4.4.4. Comparison on real-world CMOPs

Finally, we compare our algorithm with the contrast algorithm on real-world problems. The heat exchange network design problem [40] has three optimization objectives, eight equality constraints, and nine decision variables. The process synthesis problem [41] has two objectives, nine inequality constraints and seven decision variables; The synchronous optimal pulse width modulation of 3-level and 5-level inverter problem [42,43] has two optimization objectives, 24 inequality constraints, and 25 decision variables. The definition of optimization objectives and constraints can be found in their references. Table 2 shows the HV results of our algorithm and the comparison algorithm. We can see that our algorithm achieved the largest HV on these four real-world problems, representing the best performance of our algorithm on these four problems.

Table 1
Mean and standard deviation of IGD values on DASCMOP, DOC and LIRCMOP problems. 'N/A' indicates that no feasible solution was found. '+', '−', and '≈' indicate that the result is significantly better, significantly worse, and statistically similar to that obtained by CAEAD, respectively.

Problem	AEAD	MSCMO	CCMO	NSGAII-CDP	MOEAD-DAE	CTAEA	CMOEAD
DASCMOP1	2.2753e-3 (7.75e-4)	2.6588e-3 (8.02e-4) −	2.3708e-3 (4.01e-4) ≈	7.2194e-1 (3.91e-2) −	6.3378e-1 (1.47e-1) −	2.6126e-1 (1.97e-2) −	7.0520e-1 (4.01e-2) −
DASCMOP2	2.9566e-3 (9.11e-5)	3.0049e-3 (5.92e-5) −	3.2495e-3 (1.52e-4) −	2.7335e-1 (3.22e-2) −	2.0817e-1 (3.45e-2) −	1.0197e-1 (2.47e-2) −	2.5216e-1 (3.64e-2) −
DASCMOP3	1.8899e-2 (1.74e-3)	1.8940e-2 (1.78e-3) −	1.9839e-2 (8.07e-4) −	3.5787e-1 (3.48e-2) −	3.0030e-1 (1.39e-1) −	1.3699e-1 (2.74e-2) −	3.4465e-1 (2.38e-2) −
DASCMOP4	1.8749e-3 (6.59e-4)	2.3106e-2 (6.65e-2) −	2.6035e-2 (8.94e-2) ≈	1.5102e-3 (7.03e-5) ≈	1.3186e-3 (3.34e-5) +	9.9648e-3 (2.00e-3) −	4.1111e-2 (9.23e-2) −
DASCMOP5	3.3920e-3 (2.13e-4)	3.4094e-3 (3.50e-4) ≈	3.5070e-3 (4.07e-4) ≈	3.5281e-3 (1.29e-4) −	3.0901e-3 (8.60e-5) +	7.2451e-3 (4.88e-4) −	4.9095e-3 (8.20e-4) −
DASCMOP6	1.8843e-2 (2.11e-3)	3.3263e-2 (6.72e-2) −	7.7827e-2 (1.21e-1) −	3.1107e-1 (1.52e-1) −	4.1276e-2 (4.17e-2) −	2.2902e-2 (2.82e-3) −	1.1472e-1 (1.82e-1) −
DASCMOP7	3.7299e-2 (2.84e-3)	4.5224e-2 (2.41e-2) −	6.5038e-2 (1.28e-1) −	4.9760e-2 (2.41e-3) −	3.7334e-2 (1.70e-3) ≈	3.8375e-2 (8.42e-4) −	4.4446e-2 (3.60e-3) −
DASCMOP8	4.3723e-2 (2.83e-3)	5.1814e-2 (2.05e-2) ≈	5.3746e-2 (3.33e-2) −	6.1685e-2 (2.86e-3) −	5.1650e-2 (5.04e-3) −	6.0277e-2 (1.44e-2) −	6.3146e-2 (2.80e-3) −
DASCMOP9	4.0538e-2 (9.01e-4)	4.0703e-2 (1.09e-3) ≈	4.0840e-2 (1.01e-3) ≈	3.5664e-1 (7.14e-2) −	2.8690e-1 (2.16e-1) −	1.9723e-1 (4.91e-2) −	2.4529e-1 (1.27e-1) −
DOC1	5.4539e-3 (4.00e-4)	5.2062e-3 (5.59e-4) +	5.9304e-3 (8.28e-4) −	2.7269e+0 (1.99e+0) −	2.6245e+1 (1.43e+2) −	5.0144e+2 (2.16e+2) −	2.3411e+0 (1.65e+0) −
DOC2	9.5912e-3 (7.76e-3)	6.9461e-3 (6.02e-3) +	1.5395e-2 (4.65e-2) −	NaN (NaN)	NaN (NaN)	NaN (NaN)	NaN (NaN)
DOC3	2.5246e+2 (3.94e+2)	2.7487e+2 (3.23e+2) ≈	5.8574e+2 (4.57e+2) −	7.2857e+2 (2.42e+2) −	3.1870e+2 (1.58e+2) ≈	NaN (NaN)	6.8007e+2 (1.88e+2) −
DOC4	2.4582e-2 (4.05e-3)	2.5126e-2 (7.01e-3) ≈	2.3176e-2 (3.40e-3) ≈	7.4707e-1 (6.45e-1) −	4.1550e-1 (3.57e-1) −	2.1493e+2 (2.09e+2) −	6.0025e-1 (3.28e-1) −
DOC5	4.5690e+0 (2.42e+1)	2.3138e+1 (5.13e+1) ≈	1.8704e+1 (4.74e+1) ≈	NaN (NaN)	NaN (NaN)	NaN (NaN)	NaN (NaN)
DOC6	2.9699e-3 (1.62e-4)	2.9827e-3 (2.95e-4) ≈	4.8017e-3 (3.65e-3) −	1.7800e+0 (2.67e+0) −	5.6362e-1 (1.02e-1) −	5.8028e+1 (7.55e+1) −	9.8236e-1 (8.58e-1) −
DOC7	2.7905e-3 (2.90e-4)	4.8606e-3 (1.35e-2) −	2.5867e-3 (3.19e-4) +	5.2378e+0 (1.50e+0) −	7.9930e-1 (4.33e-1) −	NaN (NaN)	5.4942e+0 (2.67e+0) −
DOC8	7.1707e-2 (4.29e-3)	6.3104e-2 (3.31e-3) +	7.0950e-2 (5.12e-3) ≈	6.3183e+1 (5.01e+1) −	4.6863e+1 (2.93e+1) −	3.9502e+2 (1.44e+2) −	4.6520e+1 (4.58e+1) −
DOC9	8.5504e-2 (1.32e-2)	8.1200e-2 (1.48e-2) ≈	8.1353e-2 (1.10e-2) ≈	1.7241e-1 (9.61e-2) −	2.1809e-1 (1.09e-1) −	8.6158e-1 (3.40e-1) −	4.1651e-1 (5.61e-2) −
LIRCMOP1	9.7529e-3 (2.67e-3)	1.8753e-2 (1.34e-2) −	4.3078e-2 (3.43e-2) −	2.2564e-1 (8.81e-2) −	3.5916e-1 (1.86e-2) −	2.3352e-1 (1.46e-1) −	1.5648e-1 (5.74e-2) −
LIRCMOP2	7.6012e-3 (1.88e-3)	1.4327e-2 (1.76e-2) ≈	8.7446e-2 (3.40e-2) −	1.6045e-1 (4.20e-2) −	4.0485e-1 (4.46e-2) −	5.2875e-2 (1.33e-2) −	1.0911e-1 (2.35e-2) −
LIRCMOP3	4.0735e-3 (1.54e-3)	5.9028e-2 (4.89e-2) −	1.4596e-1 (5.71e-2) −	2.2498e-1 (7.59e-2) −	NaN (NaN)	2.2495e-1 (1.45e-1) −	1.6840e-1 (5.63e-2) −
LIRCMOP4	4.0419e-3 (2.09e-3)	5.1568e-2 (3.80e-2) −	1.1751e-1 (4.20e-2) −	2.2801e-1 (6.82e-2) −	4.1589e-1 (0.00e+0) ≈	1.1934e-1 (6.36e-2) −	1.7097e-1 (4.76e-2) −
LIRCMOP5	4.9852e-3 (1.88e-4)	5.0076e-3 (1.97e-4) ≈	5.1331e-3 (1.52e-4) −	6.4457e-1 (5.20e-1) −	4.9615e-3 (1.84e-4) ≈	6.9463e-2 (1.88e-2) −	6.1162e-1 (5.33e-1) −
LIRCMOP6	5.1106e-3 (1.52e-4)	5.0571e-3 (1.77e-4) ≈	5.2256e-3 (1.72e-4) −	4.3834e-1 (4.94e-1) −	5.0866e-3 (8.75e-4) +	9.4717e-2 (8.69e-2) −	5.7342e-1 (6.01e-1) −
LIRCMOP7	7.1288e-3 (1.66e-4)	7.1395e-3 (1.67e-4) ≈	7.2113e-3 (2.19e-4) ≈	9.6272e-3 (3.05e-3) −	9.9848e-3 (4.94e-4) −	1.9643e-2 (5.16e-3) −	7.5133e-2 (2.67e-2) −
LIRCMOP8	7.1823e-3 (2.35e-4)	7.1996e-3 (1.65e-4) ≈	7.2161e-3 (2.26e-4) ≈	1.1954e-2 (1.16e-2) −	9.1566e-3 (5.57e-4) −	1.8181e-2 (8.20e-3) −	8.6470e-2 (4.51e-2) −
LIRCMOP9	2.8101e-3 (9.30e-5)	2.4818e-3 (7.40e-5) +	2.6561e-3 (8.95e-5) +	4.0492e-1 (1.18e-1) −	2.6939e-3 (9.96e-5) +	5.4395e-2 (1.65e-2) −	3.8132e-1 (1.06e-1) −
LIRCMOP10	4.8416e-3 (1.37e-4)	4.1985e-3 (1.20e-4) +	4.6068e-3 (1.27e-4) +	2.6756e-1 (1.08e-1) −	4.3442e-3 (1.38e-4) +	7.6510e-2 (7.69e-2) −	1.9369e-1 (6.68e-2) −
LIRCMOP11	2.3986e-3 (4.73e-5)	2.3650e-3 (5.41e-5) +	2.3778e-3 (4.58e-5) ≈	1.7254e-1 (1.17e-1) −	2.5716e-3 (1.40e-4) −	1.1969e-1 (4.01e-2) −	1.9536e-1 (8.68e-2) −
LIRCMOP12	2.9713e-3 (1.50e-4)	3.0179e-3 (1.24e-4) ≈	3.1024e-3 (1.84e-4) −	1.2193e-1 (6.24e-2) −	3.1831e-3 (1.26e-4) −	1.6076e-2 (4.01e-3) −	1.5760e-1 (8.53e-2) −
LIRCMOP13	1.0706e-1 (1.97e-3)	1.0678e-1 (2.20e-3) ≈	1.0750e-1 (1.57e-3) ≈	1.1854e-1 (4.15e-3) −	9.7629e-2 (9.02e-4) +	1.0926e-1 (1.92e-3) −	9.2899e-2 (1.41e-7) +
LIRCMOP14	1.0062e-1 (1.46e-3)	1.0003e-1 (1.04e-3) ≈	9.9649e-2 (8.90e-4) +	1.2183e-1 (3.66e-3) −	1.0067e-1 (9.07e-4) ≈	1.1119e-1 (9.73e-4) −	9.5313e-2 (6.41e-7) +
+/−/≈		6/10/16	4/16/12	0/29/1	6/18/5	0/28/0	2/28/0

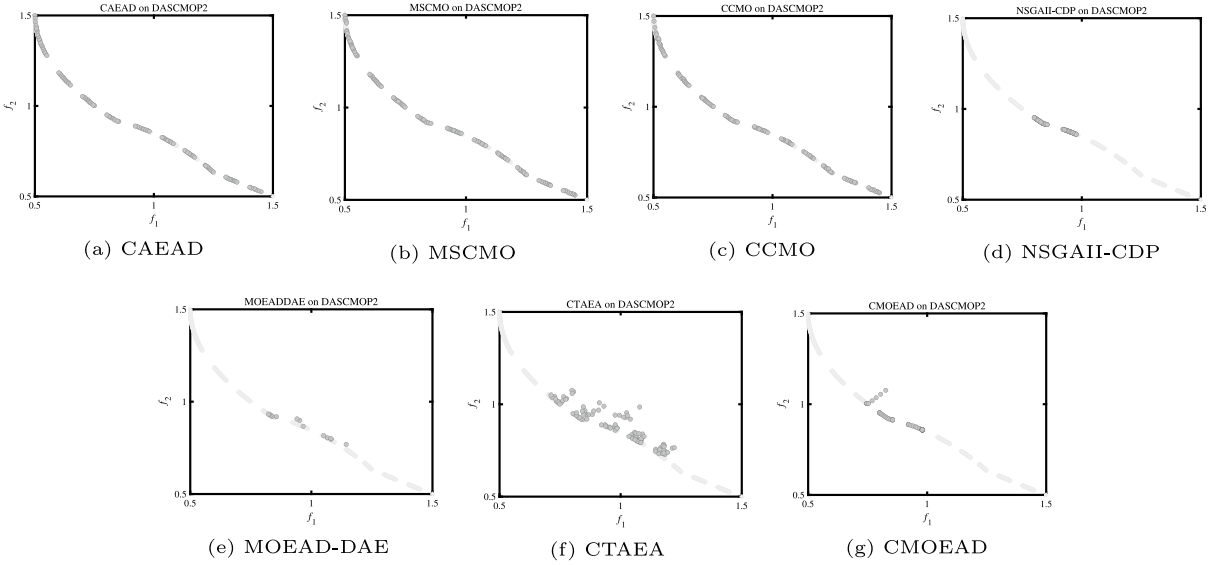

Fig. 6. Solutions with median IGD value among 30 runs obtained by CAEAD, MSCMO, CCMO, NSGAII-CDP, MOEAD-DAE, CTAEA and CMOEAD on DASCMOP2.

4.5. Further investigations of CAEAD

In this subsection, we discuss the effect of the offspring generated by *Parent*1 and *Parent*2 jointly, where CAEAD is compared to its two variants on the DASCMOP benchmark suite. The first variant of CAEAD always generates *Off*3 no matter when, which means two populations did not interact at the right time. The second variant always does not generate *Off*3 no matter when; these two variants verify whether the interaction needs to be effective at the appropriate time.

Table 3 shows the performance of CAEAD and its two variants on the DASCMOP suite. We can see that CAEAD obtained six best average values on DASCMOP. Although variant 1 obtained two best average values, it does not show a significant difference. Variant 2 did not obtain any optimal average value. CAEAD has two and five results significantly better than variant 1 and variant 2, respectively, proving that the population must interact appropriately.

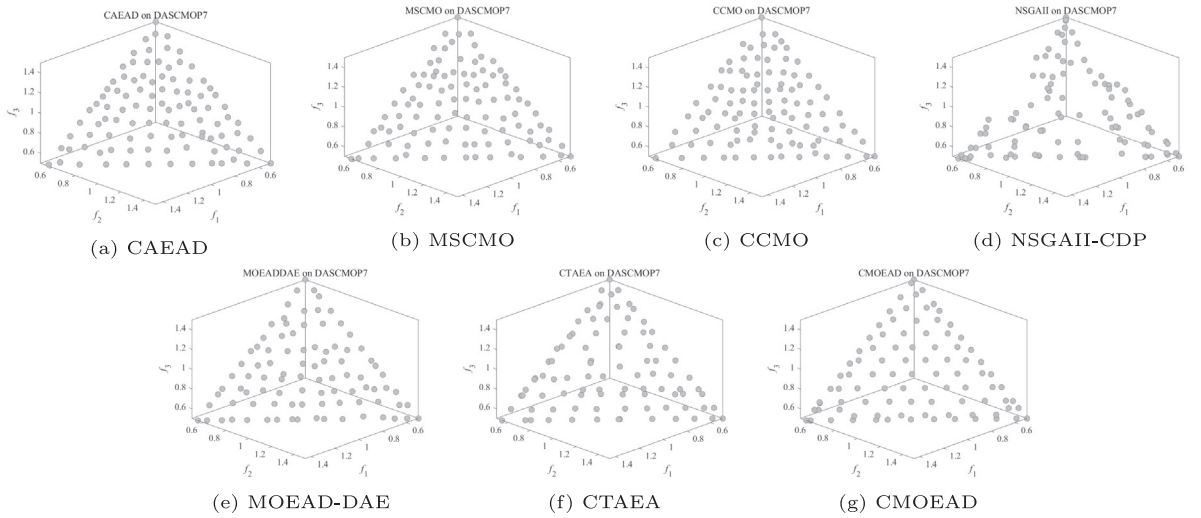

Fig. 7. Solutions with median IGD value among 30 runs obtained by CAEAD, MSCMO, CCMO, NSGAII-CDP, MOEAD-DAE, CTAEA and CMOEAD on DASCMOP7.

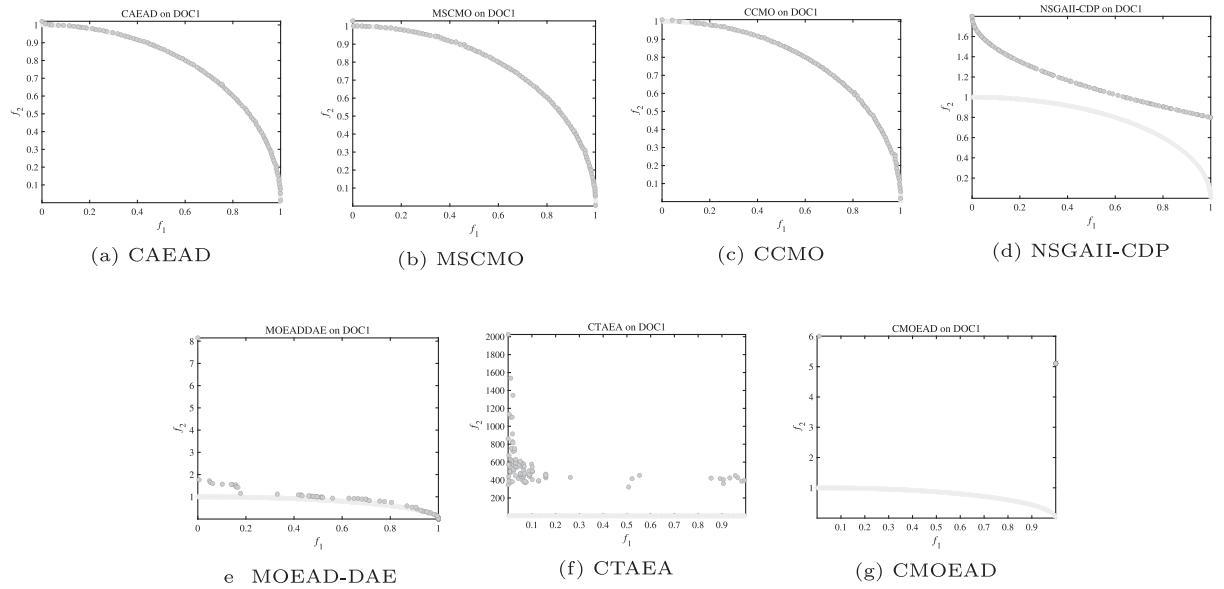

Fig. 8. Solutions with median IGD value among 30 runs obtained by CAEAD, MSCMO, CCMO, NSGAII-CDP, MOEAD-DAE, CTAEA and CMOEAD on DOC1.

5. Summary

In this paper, we have proposed a dual-population constraint optimization algorithm based on alternative evolution and degeneration. The secondary population will gradually degenerate to the feasible region after reaching the PF and then return to the PF again to provide better information for the main population. From the experimental results, we know the CAEAD is very competitive. The reasons can be summarized as follows: first, the secondary population of CAEAD has a degradation mechanism that can provide better diversity solutions for the main population and help the main population jump out of the local optimum. Secondly, the population interacts at the right time to improve the searchability of the objective space and find better solutions.

It can be seen from this paper that how to balance the feasibility, convergence, and diversity is very important. It is desirable to extend the proposed evolution and degeneration strategy and adopt more effective environmental selection strategies for solving other more challenging CMOPs. Further more, we will also explore the parameters of our algorithm in more detail so that our algorithm can have better performance on different CMOPs.

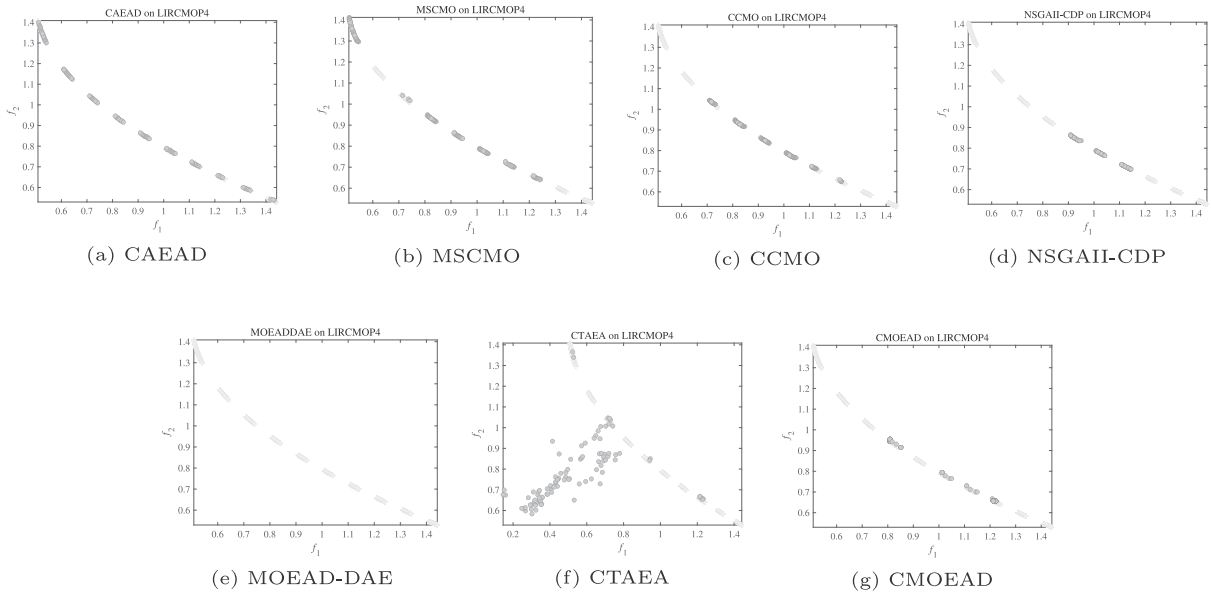

Fig. 9. Solutions with median IGD value among 30 runs obtained by CAEAD, MSCMO, CCMO, NSGAII-CDP, MOEAD-DAE, CTAEA and CMOEAD on LIRCMOP4.

Table 2
Mean and standard deviation of HV values on real-world problems. 'N/A' indicates that no feasible solution was found. '+', '−', and '≈' indicate that the result is significantly better, significantly worse, and statistically similar to that obtained by CAEAD, respectively.

Problem	CAEAD	MSCMO	CCMO	NSGAII-CDP	MOEAD-DAE	CTAEA	CMOEAD
Heat exchange network design problem (9/3/8)	7.3906e-1 (4.07e-1)	NaN (NaN)	6.2542e-1 (4.37e-1) ≈	NaN (NaN)	NaN (NaN)	NaN (NaN)	NaN (NaN)
Process synthesis problem (7/2/9)	7.8716e-1 (3.95e-4)	7.8626e-1 (6.14e-4) −	7.8669e-1 (7.51e-4) −	7.7722e-1 (1.54e-2) −	7.7015e-1 (4.68e-2) −	7.2094e-1 (3.19e-2) −	7.8104e-1 (1.12e-2) −
Synchronous optimal pulse width modulation of 3-level inverters probelm (25/2/24)	7.6737e-1 (6.27e-2)	NaN (NaN)	7.6630e-1 (8.79e-2) ≈	6.3995e-1 (7.06e-2) −	6.3098e-1 (9.06e-2) −	5.7615e-1 (1.81e-1) −	6.9376e-1 (6.51e-2) −
Synchronous optimal pulse width modulation of 5-level inverters probelm (25/2/24)	6.6567e-1 (2.39e-1)	NaN (NaN)	5.4768e-1 (2.61e-1) ≈	3.9850e-1 (3.30e-1) −	4.8232e-1 (2.66e-1) −	8.5333e-2 (1.53e-1) −	4.3214e-1 (2.96e-1) −
+/ − / ≈		0/1/0	0/1/3	0/3/0	0/3/0	0/3/0	0/3/0

Table 3
Mean and standard deviation of IGD values obtained by CAEAD and its two variants on the DASCMOP benchmark suite. Best result in each row is highlighted.

Problem	CAEAD	Variant1	Variant2
DASCMOP1	2.2753e-3 (7.75e-4)	1.9837e-3 (3.46e-4) ≈	2.1983e-3 (4.67e-4) ≈
DASCMOP2	2.9566e-3 (9.11e-5)	2.9412e-3 (7.91e-5) ≈	3.0690e-3 (1.00e-4) −
DASCMOP3	1.8899e-2 (1.74e-3)	1.9354e-2 (7.47e-5) ≈	1.9133e-2 (1.29e-3) ≈
DASCMOP4	1.8749e-3 (6.59e-4)	4.1347e-2 (1.29e-1) −	3.2512e-2 (9.17e-2) ≈
DASCMOP5	3.3920e-3 (2.13e-4)	3.3054e-3 (1.40e-4) ≈	2.8677e-2 (1.12e-1) ≈
DASCMOP6	1.8843e-2 (2.11e-3)	6.5424e-2 (1.28e-1) ≈	2.1245e-2 (5.44e-3) −
DASCMOP7	3.7299e-2 (2.84e-3)	5.5060e-2 (8.89e-2) ≈	1.2229e-1 (2.28e-1) −
DASCMOP8	4.3723e-2 (2.83e-3)	1.0155e-1 (2.23e-1) ≈	8.7870e-2 (1.53e-1) −
DASCMOP9	4.0538e-2 (9.01e-4)	4.0994e-2 (9.21e-4) −	4.1252e-2 (1.19e-3) −
+/ − / ≈		0/2/7	0/5/4

CRediT authorship contribution statement

Juan Zou: Methodology, Software. **Ruiqing Sun:** Conceptualization, Validation, Formal analysis, Writing - original draft, Writing - review & editing. **Shengxiang Yang:** Investigation, Resources. **Jinhua Zheng:** Supervision, Project administration.

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.

Acknowledgements

The authors wish to thank the support of the National Natural Science Foundation of China (Grant No. 61876164, 61772178), the Natural Science Foundation of Hunan Province (Grant No. 2020JJ4590), the Education Department Major Project of Hunan Province (Grant No. 17A212), the MOEA Key Laboratory of Intelligent Computing and Information Processing, the Science and Technology Plan Project of Hunan Province (Grant No. 2016TP1020), the Provinces and Cities Joint Foundation Project (Grant No. 2017JJ4001), the Hunan province science and technology project funds (2018TP1036).

References

[1] N. Young, Blended ranking to cross infeasible regions in constrainedmultiobjective problems (2005) 191–196.
[2] R. Yenugula, A. Ojha, Solving multiobjective optimization problems using hybrid cooperative invasive weed optimization with multiple populations, IEEE Trans. Syst. Man Cybern.: Syst. (2016) 1–12..
[3] C.-F. Juang, Y.-T. Yeh, Multiobjective evolution of biped robot gaits using advanced continuous ant-colony optimized recurrent neural networks, IEEE Trans. Cybern. (2017) 1–13..
[4] B. Tan, H. Ma, Y. Mei, M. Zhang, Evolutionary multi-objective optimization for web service location allocation problem, IEEE Trans. Services Comput. (2018) 1–1..
[5] X.-L. Zheng, L. Wang, A collaborative multiobjective fruit fly optimization algorithm for the resource constrained unrelated parallel machine green scheduling problem, IEEE Trans. Syst. Man Cybern.: Syst. (2016) 1–11..
[6] G. Sun, T. Pang, J. Fang, G. Li, Q. Li, Parameterization of criss-cross configurations for multiobjective crashworthiness optimization, Int. J. Mech. Sci. 124..
[7] K. Deb, Constrained Multi-objective Evolutionary Algorithm (2019) 85–118.
[8] K. Deb, Multi-Objective Optimization Using Evolutionary Algorithms, vol. 16, 2001..
[9] Z. Cai, Y. Wang, A multiobjective optimization-based evolutionary algorithm for constrained optimization, Evol. Comput. IEEE Trans. 10 (6) (2007) 658–675.
[10] Q. Zhang, A. Zhou, Y. Jin, Rm-meda: a regularity model-based multiobjective estimation of distribution algorithm, Evol. Comput. IEEE Trans. 12 (2008) 41–63.
[11] H. Ishibuchi, T. Fukase, N. Masuyama, Y. Nojima, Dual-grid model of moea/d for evolutionary constrained multiobjective optimization (2018) 665–672.
[12] A. Eiben, J. Smith, From evolutionary computation to the evolution of things, Nature 521 (2015) 476–482.
[13] J. he, X. Yao, Towards an analytic framework for analysing the computation time of evolutionary algorithms, Artif. Intell. 145 (2003) 59–97..
[14] Y. Tian, Y. Zhang, Y. Su, X. Zhang, K. Tan, Y. Jin, Balancing objective optimization and constraint satisfaction in constrained evolutionary multi-objective optimization, IEEE Trans. Cybern. doi:10.1109/TCYB.2020.3021138..
[15] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: Nsga-ii, ieee trans. on evol, IEEE Trans. Evol. Comput. 6..
[16] Q. Zhang, H. Li, Moea/d: a multiobjective evolutionary algorithm based on decomposition, Evol. Comput. IEEE Trans. 11 (2008) 712–731.
[17] N. Hochstrate, B. Naujoks, M. Emmerich, Sms-emoa: multiobjective selection based on dominated hypervolume, Eur. J. Oper. Res. 181 (2007) 1653–1669.
[18] Z. Fan, Y. Fang, W. Li, J. Lu, X. Cai, C. Wei, A comparative study of constrained multi-objective evolutionary algorithms on constrained multi-objective optimization problems (2017) 209–216.
[19] C. Fonseca, P. Fleming, Multiobjective optimization and multiple constraint handling with evolutionary algorithms. i. a unified formulation, IEEE Trans. Syst. Man Cybern. Part A 28 (1) (1998) 26–37.
[20] C. Coello, A. Christiansen, Moses: a multiobjective optimization tool for engineering design, Eng. Optim. 31 (1999) 337–368.
[21] K. Deb, An efficient constraint handling method for genetic algorithms, Comput. Method Appl. Math. 186 (2000) 311–338.
[22] J. Wang, G. Liang, J. Zhang, Cooperative differential evolution framework for constrained multiobjective optimization, IEEE Trans. Cybern. (2018) 1–13..
[23] S. Zeng, L. Kang, L. Ding, An orthogonal multi-objective evolutionary algorithm for multi-objective optimization problems with constraints, Evol. Comput. 12 (2004) 77–98.
[24] M. Asafuddoula, T. Ray, R. Sarker, A decomposition based evolutionary algorithm for many objective optimization, IEEE Trans. Evol. Comput. 19..
[25] Z. Fan, Y. Fang, W. Li, X. Cai, C. Wei, E. Goodman, Moea/d with angle-based constrained dominance principle for constrained multi-objective optimization problems, Appl. Soft Comput. 74..
[26] J. Yuan, A constraint handling technique using compound distance for solving constrained multi-objective optimization problems, AIMS Math. 6 (2021) 6220–6241.
[27] T. Takahama, S. Sakai, Constrained optimization by the epsilon constrained differential evolution with gradient-based mutation and feasible elites (2006) 1–8.
[28] Z. Fan, W. Li, X. Cai, H. Huang, Y. Fang, Y. Yugen, J. Mo, C. Wei, E. Goodman, An improved epsilon constraint-handling method in moea/d for cmops with large infeasible regions, Soft Comput. 23..
[29] W.-F. Gao, G. Yen, S.-Y. Liu, A dual-population differential evolution with coevolution for constrained optimization, IEEE Trans. Cybern. 45..
[30] E. Kieffer, G. Danoy, P. Bouvry, A. Nagih, A new co-evolutionary algorithm based on constraint decomposition (2017) 492–500.
[31] K. Li, R. Chen, G. Fu, X. Yao, Two-archive evolutionary algorithm for constrained multiobjective optimization, IEEE Trans. Evol. Comput. 23 (2) (2019) 303–315.
[32] Y. Tian, T. Zhang, J. Xiao, X. Zhang, Y. Jin, A coevolutionary framework for constrained multi-objective optimization problems, IEEE Trans. Evol. Comput..

[33] Z. Liu, Y. Wang, Handling constrained multiobjective optimization problems with constraints in both the decision and objective spaces, IEEE Trans. Evol. Comput. (2019) 1–1..
[34] Q. Zhu, Q. Zhang, Q. Lin, A constrained multiobjective evolutionary algorithm with detect-and-escape strategy, IEEE Trans. Evol. Comput. 24 (5) (2020) 938–947.
[35] H. Ma, H. Wei, Y. Tian, R. Cheng, X. Zhang, A multi-stage evolutionary algorithm for multi-objective optimization with complex constraints, Inf. Sci. 560..
[36] 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..
[37] Y. Tian, R. Cheng, X. Zhang, Y. Jin, Platemo: a matlab platform for evolutionary multi-objective optimization, IEEE Comput. Intell. Mag. 12 (2017) 73–87.
[38] Z. Fan, W. Li, X. Cai, H. Li, Q. Zhang, K. Deb, E. Goodman, Difficulty adjustable and scalable constrained multi-objective test problem toolkit, Evol. Comput. 28 (2019) 1–28.
[39] Z. Liu, Y. Wang, Handling constrained multiobjective optimization problems with constraints in both the decision and objective spaces, IEEE Trans. Evol. Comput. (2019) 1–1..
[40] G. Guillén-Gosálbez, A novel milp-based objective reduction method for multi-objective optimization: application to environmental problems, Comput. Chem. Eng. 35 (2011) 1469–1477.
[41] G. Kocis, I. Grossmann, A modeling and decomposition strategy for minlp optimization of process flowsheets, Comput. Chem. Eng. 13 (1989) 797–819.
[42] A. Rathore, J. Holtz, T. Boller, Synchronous optimal pulsewidth modulation for low-switching-frequency control of medium-voltage multilevel inverters, Ind. Electron. IEEE Trans. 57 (2010) 2374–2381.
[43] A. Rathore, J. Holtz, T. Boller, Optimal pulsewidth modulation of multilevel inverters for low switching frequency control of medium voltage high power industrial ac drives. doi:10.1109/ECCE.2010.5618413..
[44] 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, Evol. Comput. IEEE Trans. 18 (2014) 602–622.
[45] S. Das, P. Suganthan, Differential evolution: a survey of the state-of-the-art, Evol. Comput. IEEE Trans. 15 (2011) 4–31.
[46] R. Agrawal, K. Deb, R. Agrawal, Simulated binary crossover for continuous search space, Complex Syst. 9..
[47] P. Bosman, D. Thierens, The balance between proximity and diversity in multiobjective evolutionary algorithms, IEEE Trans. Evol. Comput. 7 (2) (2003) 174–188.
[48] D. Veldhuizen, G. Lamont, Evolutionary computation and convergence to a pareto front, Late Breaking Papers at the Genetic Programming 1998 Conference..
[49] L. While, P. Hingston, L. Barone, S. Huband, A faster algorithm for calculating hypervolume, Evol. Comput. IEEE Trans. 10 (2006) 29–38.
[50] E. Zitzler, J. Knowles, L. Thiele, Quality assessment of pareto set approximations, Multiobjective Optimization, Springer (2008) 373–404.