Complex & Intelligent Systems (2023) 9:5907–5922
https://doi.org/10.1007/s40747-023-01042-2

ORIGINAL ARTICLE

A dual-population constrained multi-objective evolutionary algorithm with variable auxiliary population size

Jing Liang[1] · Zhaolin Chen[1] · Yaonan Wang[2] · Xuanxuan Ban[1] · Kangjia Qiao[1] · Kunjie Yu[1]

Received: 21 November 2022 / Accepted: 25 February 2023 / Published online: 12 April 2023

Abstract

Constrained multi-objective optimization problems (CMOPs) exist widely in the real world, which simultaneously contain multiple constraints to be satisfied and multiple conflicting objectives to be optimized. Therefore, the challage in addressing CMOPs is how to better balance constraints and objectives. To remedy this issue, this paper proposes a novel dual-population based constrained multi-objective evolutionary algorithm to solve CMOPs, in which two populations with different functions are employed. Specifically, the main population considers both objectives and constraints for solving the original CMOPs, while the auxiliary population is used only for optimization of objectives without considering constraints. In addition, a dynamic population size reducing mechanism is proposed, which is used to adjust the size of the auxiliary population, so as to reduce the consumption of computing resoruces in the later stage. Moreover, an independent external archive is set to store feasible solutions found by the auxiliary population, so as to provide high-quality feasible solutions for the main population. The experimental results on 55 benchmark functions show that the proposed algorithm exhibits superior or at least competitive performance compared to other state-of-the-art algorithms.

Keywords Constrained multi-objective optimization problems · Dual-population · Dynamic population size reducing mechanism · External archive

Introduction

Many optimization problems in the real world usually contain multiple objective functions and complex constraints, which can be collectively referred to as constrained multi-objective optimization problems (CMOPs) [1–3]. Generally, CMOPs can be defined by the following formula [4]:

$$\begin{aligned} &\min\ F(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x})) \\ &\text{s.t. } \mathbf{x} \in \Omega \\ &\quad g_j(\mathbf{x}) \le 0, \quad j = 1, \ldots, p \\ &\quad h_j(\mathbf{x}) = 0, \quad j = p+1, \ldots, q \end{aligned} \tag{1}$$

where $F(\mathbf{x})$ is the objective vector needed to optimize which consists of m objectives, $\mathbf{x}$ is the decision vector with D-dimension in the decision space Ω, $g_j(\mathbf{x})$ stands for the jth inequality constraint, and $h_j(\mathbf{x})$ represents the $(j-p)^{\text{th}}$ equality constraint. The constraint violation of $\mathbf{x}$ on the jth constraint is usually defined as:

$$cv_j(\mathbf{x}) = \begin{cases} \max\left\{0, g_j(\mathbf{x})\right\}, & j = 1, \ldots, q \\ \max\left\{0, |h_j(\mathbf{x}) - \theta|\right\}, & j = q+1, \ldots, p \end{cases} \tag{2}$$

Jing Liang, Zhaolin Chen, Yaonan Wang, Kangjia Qiao and Kunjie Yu contributed equally to this work.

✉ Xuanxuan Ban
18438622381@163.com

Jing Liang
liangjing@zzu.edu.cn

Zhaolin Chen
zzuczl2019@163.com

Yaonan Wang
yaonan@hnu.edu.cn

Kangjia Qiao
qiaokangjia@yeah.net

Kunjie Yu
yukunjie@zzu.edu.cn

[1] School of Electrical and Information Engineering, Zhengzhou University, Zhengzhou 450001, China

[2] School of Electrical and Information Engineering, Hunan University, Changsha 410082, China

where θ is a very small constant used to relax the boundary of equality constraints. The total constraint violation degree of $\mathbf{x}$ needs to consider all equality constraints and inequality constraints, so it is defined as:

$$CV(\mathbf{x}) = \sum_{j=1}^{p} cv_j(\mathbf{x}) \tag{3}$$

Based on the above concepts, when $CV(\mathbf{x})$ is equal to zero, $\mathbf{x}$ is called a feasible solution; otherwise, $\mathbf{x}$ is called an infeasible solution.

For two feasible solutions $\mathbf{x_1}$ and $\mathbf{x_2}$, $\mathbf{x_2}$ is dominated by $\mathbf{x_1}$ when the following conditions are met: $F(\mathbf{x_1})$ is not worse than $F(\mathbf{x_2})$ in any objective, and $F(\mathbf{x_1})$ is better than $F(\mathbf{x_2})$ in at least one objective [5]. Furthermore, if a solution $\mathbf{x}^*$ is not dominated by any solution, then $\mathbf{x}^*$ is called a Pareto-optimal solution. All Pareto-optimal solutions constitute the feasible Pareto set (PS), and the mapping of PS in objective space forms the feasible/constrained Pareto front (CPF). The purpose of solving CMOP is to find a well-distributed CPF. Obviously, this is not a simple task due to the existence of conflicting objectives and multiple complex constraints.

In order to deal with the relationship among several conflicting objectives, substantial multi-objective evolutionary algorithms (MOEAs) have been proposed over the past few decades and shown excellent performance in solving unconstrained multi-objective optimization problems. These MOEAs can be classified into three categories based on their selection mechanisms: the method of dominance-based [6], the method of decomposition-based [7], and the method of indicator-based [8], in which the first method have received a lot of attention due to their simplicity and ease of implementation. However, none of these methods can be directly used to solve CMOPs since they cannot effectively handle constraints. In order to rescue this issue, constraint handing techniques (CHTs) are designed to compare feasible and infeasible solutions, which are generally combined with MOEAs to form constrained MOEAs (CMOEAs) [9].

Indisputably, the combination methods of CHTs and MOEAs are very important, different combination methods will produce different effects, which will greatly affect the performance of the algorithms. If the algorithm favors the satisfaction of constraints, the population will easily find feasible solutions, but it will also fall into the local feasible region; if the algorithm gives priority to the optimization of objectives, the diversity of the population will be enhanced, but it may lead to the failure to find feasible solutions for the population. Therefore, in order to achieve the balance between objectives and constraints, researchers began to attempt dual-population and two-stage mechanisms [10,11]. Evidence suggests that these two types of mechanisms can well balance between the objectives and constraints. In the two-stage methods, the population generally searches for unconstrained PF (UPF) in the first stage to utilize the information of the objectives, and gradually converges to CPF from UPF in the second stage. But the UPF information is ignored in the second stage, so that the population diversity cannot be maintained well. And if too much computing resources are used in the first phase, the computing resources in the second phase will not be sufficient, as a result, the population will not converge to the real CPF. For the dual-population mechanisms, the first population as main population considers both objectives and constraints, which used to search CPF and ensure the feasibility of the output population. The second population as auxiliary population is employed to explore UPF without considering constraints, which can help the first population improve diversity. However, if UPF and CPF do not overlap, finding UPF will provide less help for the main population in the later stage. Therefore, the computing resources consumed by the second population in the later stage will be wasted. So, it is necessary to consider the resource allocation for the the auxiliary population.

Based on the above discussion, this paper designs a novel dual-population algorithm, named DPVAPS. The main contributions of this paper are as follows:

1. A dynamic population size reducing mechanism is proposed, so that the size of the auxiliary population will gradually decrease during the evolution. When UPF and CPF do not completely overlap, the proposed mechanism can reduce the resource consumption of the auxiliary population in the later stage. Therefore, the main population will be allocated more computing resources to search for CPF.
2. An external archive is used to save the feasible solutions found by the auxiliary population, which can provide more different feasible solutions for the main population, thereby increasing the diversity of the main population.
3. Systematic and comprehensive experiments are carried out on five test suits sets to verify the effectiveness of the proposed algorithm.

The rest of this paper is organized as follows. the description of the existing CMOEAs are introduced in "Literature review". In "The proposed algorithm" describes the proposed DPVAPS algorithm. The experimental results are given in "Experimental study". The conclusion of the paper and future work are presented in "Conclusion".

Literature review

In this section, the related work on CMOEAs will be briefly introduced. Generally speaking, CHTs and the internal mechanism used by CMOEAs play a decisive role in

the performance of the algorithms. From this perspective, CMOEAs can be divided into the following categories: (1) The penalty function method [12–14]; (2) The methods that consider constraints and objectives separately [15–17]; (3) the method of using two-stage [18–20]; and (4) the method based on dual-population [21–23].

The penalty function method is one of the simplest CHTs, which constructs a penalty coefficient for the constraint violation degree of the individual, thus transforming the constrained optimization problem into an unconstrained optimization problem, and then MOEAs are employed to solve the transformed problems. Obviously, the penalty coefficient has a significant impact on the performance of the penalty function method. Penalty methods are classified into static and dynamic methods according to whether the penalty coefficient will change with the evolutionary generation. The weight of objectives and constraints is constant in the static method, which is very detrimental to the balance of them, and the performance of the static method will drop dramatically once the problem becomes complex. Therefore, the dynamic method is very popular in solving CMOPs. In [24], an adaptive dynamic penalty function method based on threshold is proposed by Jan et al., then it is embedded into the multiobjective evolutionary algorithm based on decomposition (MOEA/D) framework to solve CMOPs. Jiao et al. [25] designed a feasible-guiding strategy, in which a new fitness value is obtained by using the objective functions modified by the constraint violation degree. In addition, Ma et al. [26] assigned two rankings to each individual, one based on the objective function value and the other based on the constraint domination principle (CDP). The two rankings biased toward the objectives and constraints, respectively, and then the proportion of feasible solutions in the population is used to weight the them. Although the effect of the dynamic method will be significantly improved compared with the static method, it is very difficult to set appropriate change rules.

The methods that consider constraints and objectives separately mainly include CDP [27], stochastic ranking (SR) [28], and ϵ constrained method [29]. For CDP, the dominant relationship between individual $\mathbf{x}$ and individual $\mathbf{y}$ are as follows:

(a) $\mathbf{x}$ is feasible, but $\mathbf{y}$ is infeasible, then individual $\mathbf{x}$ dominates $\mathbf{y}$.
(b) Both $\mathbf{x}$ and $\mathbf{y}$ are feasible, but $\mathbf{x}$ has the better objective function values than $\mathbf{y}$, then $\mathbf{x}$ dominates $\mathbf{y}$.
(c) Neither $\mathbf{x}$ nor $\mathbf{y}$ is feasible, but $CV(\mathbf{x}) < CV(\mathbf{y})$, then individual $\mathbf{x}$ dominates $\mathbf{y}$.

CDP is widely used by researchers since it is easy to implement and free-parameter. The most typical algorithm is NSGA-II-CDP [27], which can help the population to find the feasible region quickly and improve the convergence of the population. However, CDP pays too much attention to the constraints, as a result, the population is easy to fall into the local optimum especially in some complex problems. In an attempt to surmount the shortcomings, SR and ϵ constrained method are designed to consider the information of objectives. For the former, a probability parameter is used to determine whether comparisons between individuals are based on the CDP or the objective function. In this way, the information of objectives can be used to some extent and the diversity of population will be enhanced. However, improper parameter settings will cause the population to encounter difficulties in convergence. ϵ constrained method uses the parameter ϵ to relax the constraints, and all individuals in the population whose constraint violation degree is less than ϵ are regarded as feasible solutions. ϵ will gradually decrease with the evolution process, and the information of the objectives will also be used to guide the population evolution, but it is difficult to grasp the appropriate change rule, and the setting of the initial ϵ value is very challenging.

For the method of using two-stage, it divides the evolution process of the population into two stages, and different stages perform different functions. Fan et al. [18] put forward a push and pull (PPS) framework whose first phase ignored all constraints and focused on exploring UPF. In the second stage, the ϵ constrained method was employed to promote the population recovery to CPF from UPF. However, if the UPF is difficult to find for the population, too much computing resources will be consumed by the first stage, resulting in insufficient computing resources in the second phase, so that the complete CPF could not be found. In [30], a two-phase framework, TOP, was proposed by Liu and Wang. The purpose of the first stage is to transform CMOP into a constrained single-objective optimization problem for finding feasible regions by weighting all objectives. In the second phase, the CMOEA was employed to explore the complete CPF. Nevertheless, the performance of the first stage is not satisfactory for the problems with small or discrete feasible regions. Tian et al. [31] designed a two-stage CMOEA, including stage A and stage B. Stage A assigns the same priority to the objectives and the constraints, with the aim of finding all feasible domains. The constraints have a higher priority than the objectives in stage B, so as to promote the population converge to Pareto front. However, some excellent objectives information will be lost since the priority of the objectives is not greater than the constraints. Liang et al. [32] explored the relationship between UPF and CPF at the learning stage, and then the relationship was used to guide the evolution of the population at the evolutionary stage. In general, the design of the two-stage mechanism is very prominent, and the transition conditions between stages should be reasonable, which will greatly affect the performance of the algorithm in solving CMOPs. Moreover, for the current two-stage algorithm,

the utilization of the objectives information is ignored in the second stage.

To solve the above issues, some dual-population algorithms have been proposed. In [22], a dual-population algorithm named CTAEA was proposed. It used two archive populations, one archive considered both constraints and objectives, while the other only considered objectives. The exchange of information between the two archives occured during the offspring generation and environmental selection. However, this strong correlation between two populations leads to the slow search speed of the algorithm. To circumvent this disadvantage, Tian et al. [21] developed a co-evolutionary framework (CCMO) using the weak correlation, in which the first population was mainly used to solve the original CMOPs, and the second population converged to the UPF by ignoring constraints directly. These two populations independently produced offspring and only exchanged information during the environmental selection. Qiao et al. [33] suggested a evolutionary multitasking-based CMO framework (EMCMO). Compared with CCMO, EMCMO considers the types of problems and analyzes the information of different individuals. However, the performance of the algorithm is not satisfactory when solving the problem whose UPF is far from the CPF. In [23], a dual-population algorithm named BiCo was proposed, which used two populations to approximate the CPF from different directions. The above dual-population algorithms all used two populations to consider constraints and objectives respectively, and used information exchange to balance constraints and objectives. However, when the UPF and CPF do not completely overlap, the second population has less auxiliary effect for the first population in the later stage. At this time, if the auxiliary population and the main population enjoy the same computing resources, the resource wasted by the auxiliary population will affect the performance of the algorithm. Therefore, it is necessary to reduce the resource consumption of the second population, so that the main population can use more resources to find the CPF.

In order to solve the above issues in dual-population algorithms, this paper proposes a novel dual-population algorithm. In the proposed algorithm, a dynamic population size reducing mechanism is proposed, which is used to save the computing resources occupied by the auxiliary population in the later stage. More detailed description will be presented in the next section.

The proposed algorithm

The procedure of the proposed algorithm

The overall framework and main procedure of the proposed algorithm are shown in Fig. 1 and Alogrithm 1, respectively. First, the two populations are randomly initialized respectively. POP_1 is the main population and mainly responsible for searching for CPF. Meanwhile, the CDP method which has a strong ability to find feasible solutions is employed by the main population to improve the feasibility. The second population POP_2 is the auxiliary population, which does not consider any constraint and adopts the fast non-dominated sorting method to push the population to approach the UPF, so as to utilize infeasible solutions to expand search. As shown in lines 6–9, the two populations use genetic algorithm (GA) to generate offspring, respectively. More specifically, the tournament selection method is used to select mating parents from the population, and the simulated binary crossover (SBX) [34] and the polynomial mutation (PM) [35] are employed to produce offspring based on the selected mating parents. The information exchange is achieved between the two populations mainly by sharing offspring during the environmental selection as shown in lines 14–15. Since the auxiliary population ignores all constraints, its convergence and diversity are better than the main population. Therefore, the auxiliary population will reach the feasible region earlier than the main population, which can provide more search directions to the main population. By this way, the diversity of the main population will be improved.

As shown in line 15, the dynamic adjustment of the auxiliary population size is mainly realized by a dynamic coefficient δ, which will decrease as the number of evolutionary generation increases. For a CMOP, if the UPF and CPF do not completely overlap, the information of UPF in the later stage is not very helpful for the main population to search for CPF. As the size of the auxiliary population decreases, the computing resources occupied in the later stage will also decrease. In this way, more computing resources can be allocated to the main population. As shown in lines 16–19, the external archive A is mainly used to store feasible solutions found by the auxiliary population, so as to provide more information to the main population during the environmental selection. When the size of A exceeds NP, A will be truncated [21].

Dynamic population size reducing mechanism

The dynamic population size reducing mechanism is designed with the purpose of avoiding the waste of computing resources in the later evolution stage. During the evolution of the two populations, the infeasible regions will not hinder the evolution of the auxiliary population since it does not consider any constraints, so it will converge faster than the main population. At this point, the information of the auxiliary population will help the main population to cross those large infeasible regions and reach the optimal feasible region. Moreover, the auxiliary population will provide more evolutionary directions for the main population to increase

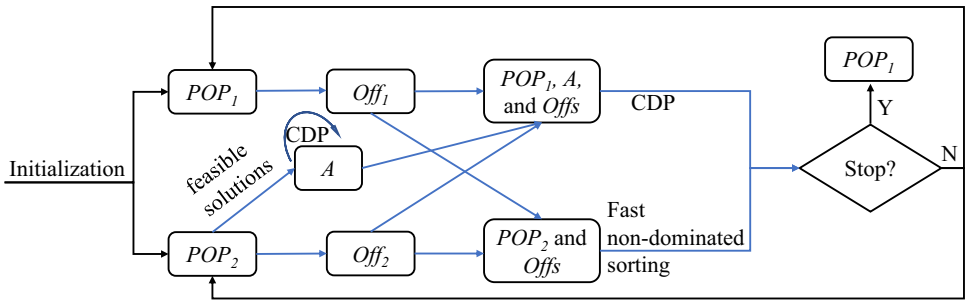

Fig. 1 The framework of DPVAPS

Dynamic population size reducing mechanism

Algorithm 1 The procedure of DPVAPS

Input: NP (population size)
$totalFES$(maximum number of function evaluations)
A (external archive)
δ (dynamic coefficient)
Output: POP_1 (the main population)
1: $POP_1 \leftarrow RandomInitialization(NP)$;
2: $POP_2 \leftarrow RandomInitialization(NP)$;
3: POP_1 and POP_2 are evaluated by CDP and fast non-dominated sorting, respectively;
4: $FES = NP * (1 + \delta)$;
5: **While** $FES \leq totalFES$
6: $Parent_1 \leftarrow$ Select NP parents from POP_1 by the tournament method;
7: $Parent_2 \leftarrow$ Select $round(NP * \delta)$ parents from POP_2 by the tournament method;
8: $Off_1 \leftarrow$ Generate $N/2$ offsprings based on $Parent_1$ by the operators of GA;
9: $Off_2 \leftarrow$ Generate $round(NP * \delta/2)$ offsprings based on $Parent_2$ by the operators of GA;
10: $POP_1 \leftarrow POP_1 \cup Off_1 \cup Off_2 \cup A$;
11: $POP_2 \leftarrow POP_2 \cup Off_1 \cup Off_2$;
12: Evaluate POP_1 and POP_2;
13: $FES = FES + NP/2 + round(NP * \delta/2)$;
14: $POP_1 \leftarrow$ Select NP solutions from POP_1 by the CDP method;
15: $POP_2 \leftarrow$ Select $round(NP * \delta)$ solutions from POP_2 by the fast non-dominated sorting method;
16: $A \leftarrow A \cup$ The feasible solutions selected from POP_2;
17: **If** $|A| > NP$
18: $A \leftarrow$ Select NP individuals by CDP;
19: **End If**
20: $\delta \leftarrow$ Update δ by formula (4);
21: **End While**

its diversity. However, after the auxiliary population reaches the UPF, its offspring will also be generated in the vicinity of the UPF and no more beneficial information will be produced. Even if more computing resources are given to the auxiliary population at this time, the help to the main population will be very limited. In contrast, the main population requires more computational resources to explore the CPF. Therefore, a dynamic population size reducing mechanism is designed to avoid the waste of computing resources by the auxiliary population and to allocate more resources to the main population.

To illustrate the dynamic population size reducing mechanism, an example is provided in Fig. 2.

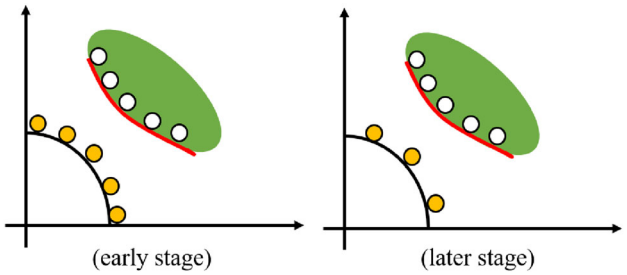

Fig. 2 Schematic diagram of dynamically adjusting population size, where the black and red curves represent the UPF and CPF, respectively. The green area represents the feasible region. The white and yellow dots represent the main population and auxiliary population, respectively

In this case, the auxiliary population can only provide very limited assistance to the main population in the later stage, because the information of the auxiliary population cannot help the main population to improve the diversity on CPF. At this time, if the same computing resources is still allocated to the two populations, that is, the auxiliary population is kept the same population size as the main population, limited computing resources will be wasted by the auxiliary population. Therefore, a dynamic coefficient δ develpoed to change the size of the auxiliary population is proposed as follows:

$$\delta = \frac{|Gen|}{|Gen_{max}|} * (\delta_{min} - 1) + 1 \tag{4}$$

where $|Gen|$ and $|Gen_{max}|$ represent the current generation and the maximum generation, respectively. δ_{min} represents the minimum of δ. Figure 3 shows the changing rule of dynamic coefficient δ with the continuous increase of evolutionary generation. Given that the initial size of the auxiliary population is 100 and the δ_{min} is 0.1, then as δ decreases from 1 to 0.1, the size of the auxiliary population will gradually decrease and become 10 in the last generation. In this way, more computing resources can be used by the main population in the later stage, thereby improving diversity of the main population on the CPF. It should be noted that we have verified that the algorithm can achieve the best performance when δ_{min} is set to 0.1.

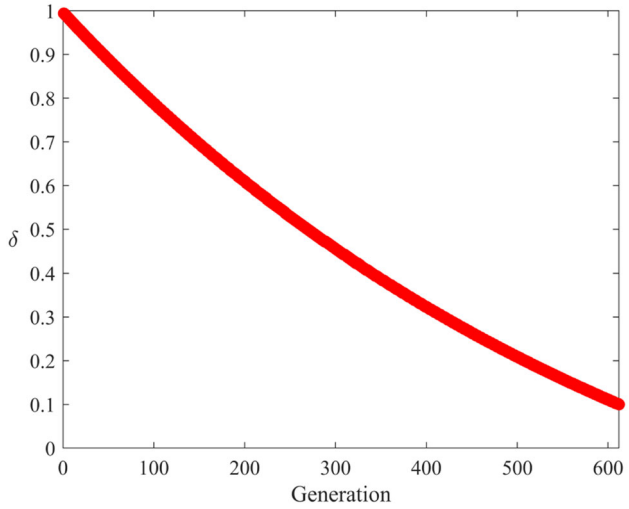

Fig. 3 The changing trend of δ

External archive

In an attempt to increase the diversity of main population, the external archive A is created. Figure 4 illustrates the importance of setting an archive.

Archive A saves the feasible solutions encountered by $POP2$ during the evolutionary process. Because the auxiliary population does not consider constraints, there is a high probability that the high-quality feasible solutions in the auxiliary population will be dominated by those infeasible solutions, as a result, these feasible solutions are eliminated in the environmental selection stage. In addition, since these feasible solutions in the auxiliary population are not exactly the same as the feasible solutions searched by the main population, if these feasible solutions are preserved, they will form a complementary effect with the main population, searching the areas not searched by the main population, which will be extremely helpful to the diversity of the main population. Most importantly, it is very helpful to improve the distribution ability of the main population on the CPF.

Analysis for different evolution stages of DPVAPS

In this paper, the dual-population mechanism is used, and the size of the second population will decrease with the increase of evolution algebra. CCMO [21] also uses the dual-population mechanism, moreover, they both use NSGA-II as the basic optimizer. Here, their differences are compared:

Similarly, the dual-population mechanism is used in [21]. The purpose of the first population in CCMO is to search CPF, while the second population mainly focuses on the exploration of UPF. The size of the second population is always the same as that of the first population, so these two

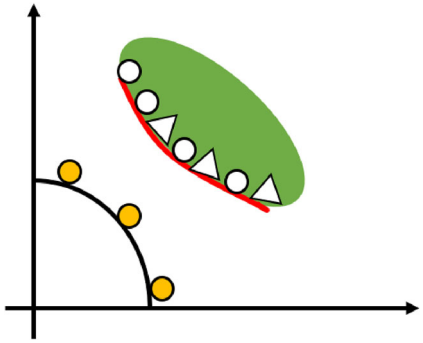

Fig. 4 Schematic diagram of the reason for setting external archive, where the white triangles represent the external archive

populations consume the same computing resources in the process of evolution. In addition, the second population lacks the mechanism to save feasible solutions, so the feasible solutions searched by the second population are not fully utilized by the first population, resulting in a waste of knowledge. However, the second population of DPVAPS decreases dynamically during evolution, so more computing resources will be used by the main population. In addition, the feasible solutions found by the second population will be saved to provide effective information for the main population and help the main population search CPF more completely. These are the two main differences between DPVAPS and CCMO.

In addition, in order to further study the efficiency of the proposed algorithm, the benchmark problem LIRCMOP8 is selected for comparative experiments. LIRCMOP8 has many large infeasible regions, these large infeasible blocks will block the evolution of the population and its UPF is located in the infeasible region, while the CPF is on the boundary of feasible region. The populations distribution of NSGA-II, CCMO, and DPVAPS in the objective space in the early, middle, and later stages on the problem LIRCMOP8 are listed in Fig. 5. It can be seen that in the early stage of evolution, the main populations of the three algorithms begins to converge towards CPF, and some individuals have reached the vicinity of CPF. In addition, the distribution of auxiliary populations of CCMO and DPVAPS is more dispersed due to all constraints are not considered. In the middle stage of evolution, some individuals of NSGA-II reach the CPF, while the other individuals are blocked by large infeasible blocks and stayed in the region far away from the CPF. The main populations of CCMO and DPVAPS are almost all distributed around the CPF, and their auxiliary populations are all near the UPF. The main reason is that their auxiliary populations do not consider constraints, which can help the main populations cross large infeasible blocks to reach the CPF. In the later stage of evolution, the population of NSGA-II is similar to that in the middle stage. Some of individuals are distributed on the CPF, while those blocked by the infeasible region are still stagnant

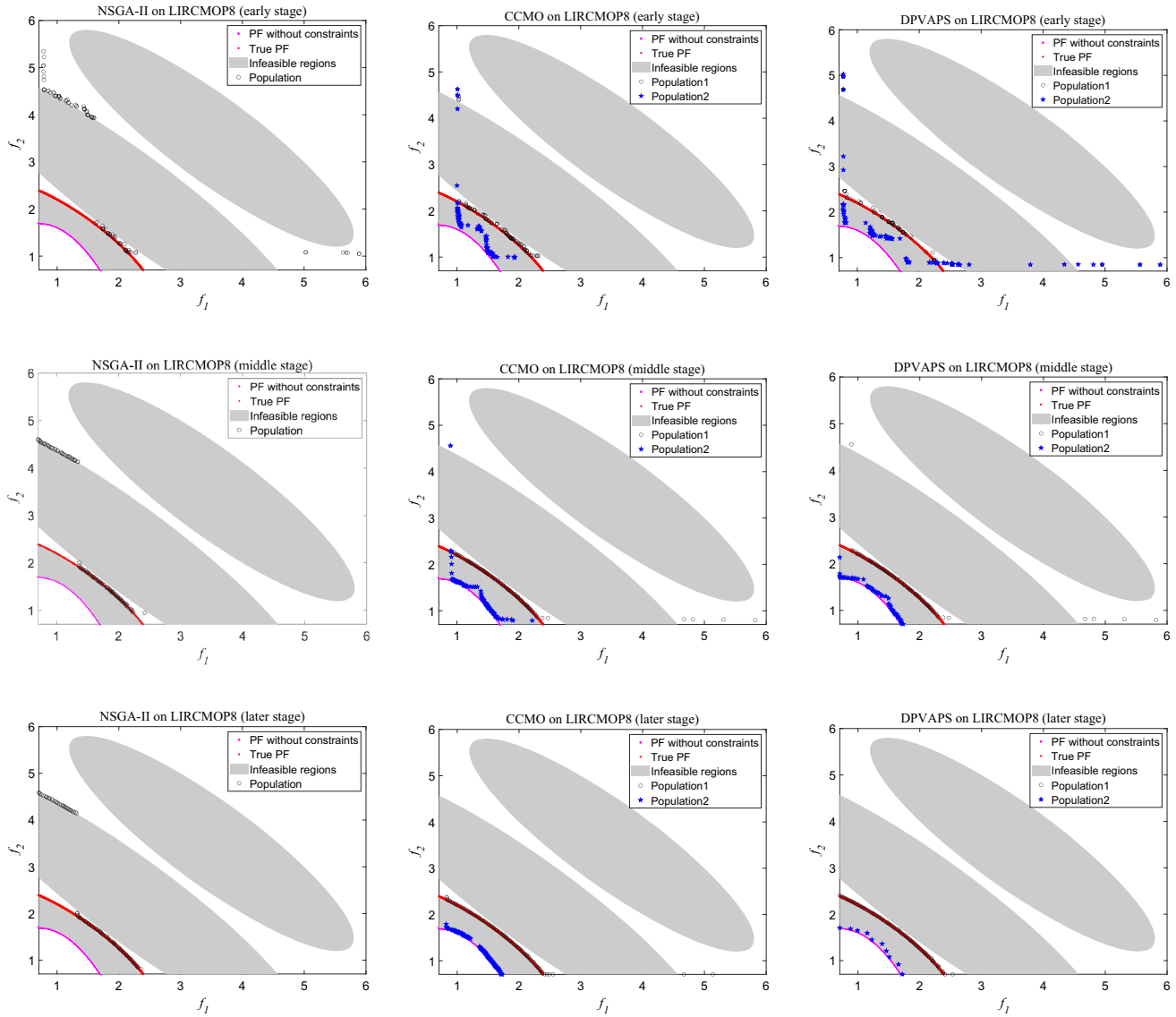

Fig. 5 Populations in the early, middle, and later stages of NSGA-II, CCMO, and DPVAPS on benchmark problem LIRCMOP8

away from the CPF. This is because NSGA-II lacks the mechanism to help the population cross the infeasible regions, so the individuals are easily hindered by the infeasible regions and cannot reach the optimal feasible region. In contrast, all individuals in the main populations of CCMO and DPVAPS can reach CPF. The CPF covered by CMMO is incomplete, the upper left corner of the CPF is not found. While the main population of DPVAPS is more evenly and completely distributed on the CPF. For two auxiliary populations, they have reached UPF. However, the number of individuals in auxiliary populations of DPVAPS is reduced to 10, while the size of auxiliary population of CCMO is still consistent with the main population. In this way, more computing resources can be used by the main population of DPVAPS. Moreover, the external archive in DPVAPS will save the feasible solutions in the auxiliary population and make up the area not searched by the main population. That is why the convergence and uniformity of the main population of DPVAPS is better than that of CCMO.

Experimental study

In this section, in an attempt to test the performance of the proposed algorithm, it will be compared with five state-of-the-art CMOEAs: NSGA-II [27], CCMO [21], BiCo[23], PPS [18], and ToP [30], in which the NSGA-II is one of the most classical CMOEAs and it is often chosen as the optimizer of the main population in the dual-population mechanism; CCMO and BiCo are classical dual-population

Table 1 The specific parameter settings of the test sets

Test suits	The objective number (m)	Dimension (D)
MW	$m = 3$: MW4, MW8, and MW14; $m = 2$: other problems	$D = 15$
LIRCMOP	$m = 3$: LIRCMOP13-14; $m = 2$: the rest problems	$D = 10$
DTLZ	$m = 3$	$D = 7$: C1_DTLZ1, DC1_DTLZ1, DC2_DTLZ1, and DC3_DTLZ1; $D = 12$: the remaining problems
CTP	$m = 2$	$D = 4$
DASCMOP	$m = 2$: DASCMOP1-6; $m = 3$: DASCMOP7-9	$D = 30$

algorithms; and PPS and ToP are the representatives of two-stage algorithms.

The experiments are carried out on five commonly used test sets, i.e., MW [36], LIRCMOP [37], DTLZ [22], CTP, and DASCMOP [38]. The specific settings of the number of objectives and dimensions are shown in Table 1. In addition, the population size NP is set to 100, the maximum number of evaluations is set to 60000, and each algorithm independently runs 30 times on each benchmark function. For the sake of fairness, the evolutionary strategies of all comparison algorithms are GA, and other parameters are the same as their original papers. The inverted generational distance (IGD) [39] and feasible rate (FR) are employed as indicators to measure the performance of the algorithms. IGD reflects the convergence and diversity of the algorithm. The smaller the IGD value of the algorithm is, the better its performance will be. FR highlights the ability of the algorithm to search for feasible solutions. The larger the value, the stronger the algorithm's ability. All comparative experiments are performed on the PlatEMO [40] platform.

Verification of the proposed mechanisms

The dynamic population size reducing mechanism and the external archive are the two main contributions of this paper. In order to verify the effectiveness of these two mechanisms, two variants, DPVAPS_1 and DPVAPS_2, are created as control groups for comparative experiments in this section, in which DPVAPS_1 means that only the dynamic population size reducing mechanism is employed without using external archive, and DPVAPS_2 means that only external archive is used. The specific average and standard deviation of IGD values are shown in Table 2, in which the best result is marked in bold. "+", "−", and "=" indicate that the variant is significantly better than, worse than, or comparable to the proposed algorithm, respectively.

As can be seen from Table 2, DPVAPS achieves the best IGD average values on 19 out of the 24 functions compared with DPVAPS_1. What's more, the performance of DPVAPS is significantly better than that of DPVAPS_1 on 14 benchmark problems, while it is outperformed by it on only 2 benchmark problem. For DPVAPS_1, the external archive is not employed, thus the feasible solutions in the main population are preserved. The disadvantage is that the information of the promising feasible solutions in the auxiliary population is ignored. As a result, the diversity of the main population is lost. Because the auxiliary population does not consider any constraint, the probability of feasible solutions being dominated by infeasible solutions is large, so these solutions will be eliminated in the environment selection stage. While these feasible solutions may be high-quality information for the main population. By using the feasible solutions in the auxiliary population, the evolution direction of the main population can be increased, more feasible regions can be searched, and some CPF fragments that are not easy to search can also be explored. Figure 6 shows the populations in the early, middle, and later stages of DPVAPS_1 on benchmark problem LIRCMOP8. Compared with the main population distribution of DPVAPS, DPVAPS_1 is not completely distributed on CPF, which is caused by the loss of diversity, while the external archive can remedy this defect. Therefore, the performance of DPVAPS is superior to its variant without external archive.

From the Table 2, DPVAPS achieves the better IGD average values on 22 out of the 24 problems. In addition, according to the Wilcoxon rank-sum test, DPVAPS performs significantly better than its variant DPVAPS_1 on 15 functions, but significantly worse than it on only 1 test problems. Compared with DPVAPS_2, the computational resources consumed by the auxiliary population of DPVAPS gradually decrease with the evolutionary process, so that more computational resources can be used by the main population. For some complex problems, if only half of the computational resources are employed by the main population, there is a situation that the computational resources are exhausted before the complete CPF is searched. While if the main population has enough computing resources, it will have the ability to continue to approach the real CPF after reaching the optimal feasible region. Therefore, DPVAPS shows more excellent performance than DPVAPS_2.

In summary, according to the above results and analysis, the two mechanisms in the proposed algorithm are very effective.

Table 2 Average IGD values of DPVAPS_1, DPVAPS_2, and DPVAPS on LIRCMOP and DTLZ tests

Problem	DPVAPS_1	DPVAPS_2	DPVAPS
LIRCMOP1	**2.3707e−1 (8.35e−2) +**	3.1181e−1 (1.30e−1) =	2.96798e−1 (9.20e−2)
LIRCMOP2	**1.6755e−1 (9.51e−2) +**	1.6554e−1 (5.38e−2) =	2.1026e−1 (9.22e−2)
LIRCMOP3	**2.6450e−1 (6.93e−2) =**	2.8104e−1 (1.23e−1) =	2.8250e−1 (9.08e−2)
LIRCMOP4	**2.4162e−1 (9.66e−2) =**	2.4403e−1 (7.90e−2) =	2.6383e−1 (7.84e−2)
LIRCMOP5	1.8162e−2 (7.00e−3) −	2.8261e−2 (1.80e−2) −	**1.6034e−2 (9.38e−3)**
LIRCMOP6	1.8189e−2 (1.08e−2) −	2.1968e−2 (1.63e−2) −	**1.2559e−2 (3.71e−3)**
LIRCMOP7	1.2483e−2 (4.61e−3) −	1.1806e−2 (4.03e−3) −	**9.8116e−3 (2.24e−3)**
LIRCMOP8	9.4475e−3 (3.16e−3) =	1.1143e−2 (4.82e−3) −	**8.6200e−3 (1.35e−3)**
LIRCMOP9	8.4287e−2 (5.80e−2) =	8.0360e−2 (4.67e−2) =	**5.3668e−2 (3.46e−2)**
LIRCMOP10	9.5758e−3 (4.89e−3) −	7.2114e−3 (9.11e−4) −	**6.8317e−3 (1.24e−3)**
LIRCMOP11	2.7502e−3 (4.32e−4) −	2.9374e−3 (4.01e−4) −	**2.5734e**−3 **(2.08e−4)**
LIRCMOP12	5.6977e−3 (3.36e−3) −	8.8732e−3 (7.06e−3) −	**4.2910e−3 (2.48e−3)**
LIRCMOP13	9.2905e−2 (1.03e−3) −	9.3265e−2 (9.22e−4) −	**9.1558e−2 (8.63e−4)**
LIRCMOP14	9.5250e−2 (8.52e−4) −	9.5581e−2 (7.54e−4) −	**9.4227e−2 (9.42e−4)**
C1_DTLZ1	1.9922e−2 (1.41e−4) −	1.9971e−2 (1.62e−4) −	**1.9782e−2 (1.58e−4)**
C1_DTLZ3	5.3812e−2 (6.44e−4) −	5.3593e−2 (6.45e−4) =	**5.3303e−2 (4.75e−4)**
C2_DTLZ2	4.2614e−2 (6.20e−4) =	4.2858e−2 (4.51e−4) −	**4.2303e−2 (5.68e−4)**
C3_DTLZ4	**1.2023e−1 (1.56e−1) =**	1.2086e−1 (1.36e−1) =	1.4459e−1 (1.89e−1)
DC1_DTLZ1	1.1442e−2 (8.49e−5) −	1.1508e−2 (1.39e−4) −	**1.1393e−2 (7.58e−5)**
DC1_DTLZ3	1.1387e−1 (1.09e−3) =	1.1429e−1 (7.01e−4) −	**1.1364e−1 (9.02e−4)**
DC2_DTLZ1	2.0108e−2 (1.61e−4) −	2.0258e−2 (2.24e−4) −	**1.9973e−2 (1.29e−4)**
DC2_DTLZ3	1.0656e−1 (1.56e−1) −	**5.3762e−2 (2.47e−3) +**	7.2399e−2 (9.33e−2)
DC3_DTLZ1	6.9160e−3 (1.36e−4) −	6.9174e−3 (1.33e−4) −	**6.8349e−3 (6.56e−5)**
DC3_DTLZ3	2.0530e−1 (1.35e−1) =	**1.5999e−1 (2.63e−3) =**	1.7800e−1 (6.84e−2)
+/−/=	2/14/8	1/15/8	

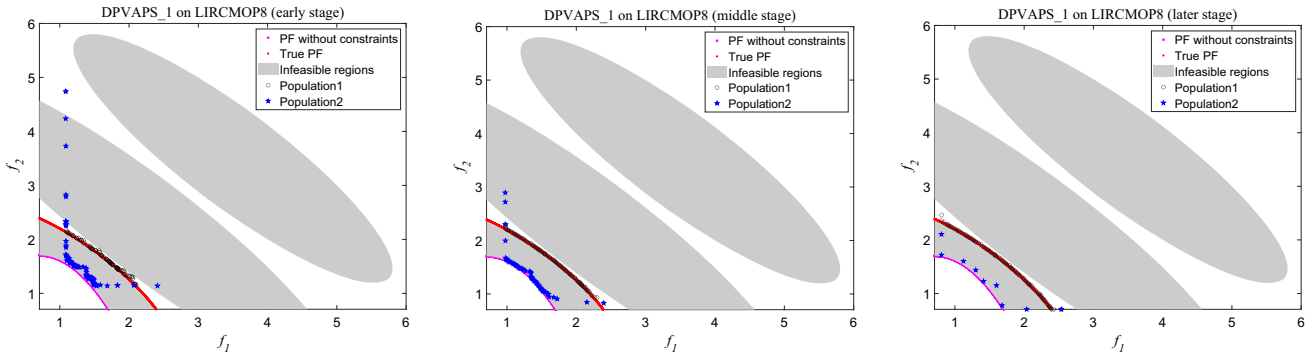

Fig. 6 Populations in the early, middle, and later stages of DPVAPS_1 on benchmark problem LIRCMOP8

Parameter analysis

In the proposed algorithm, the size of the auxiliary population is controlled by parameter δ_{min}. The value of parameter δ_{min} in the dynamic population size reducing mechanism is 0.1, indicating that the size of the auxiliary population gradually decreases to 10 with the increase of evolution. To verify the suitability of this parameter, several variants are created, namely DPVAPS_10, DPVAPS_20, DPVAPS_30, and DPVAPS_40, in which δ_{min} is taken as 0.2, 0.3, 0.4, and 0.5 respectively, indicating that the size of the auxiliary population gradually decreases to 20, 30, 40, and 50. The experiments are conducted on two test sets, LIRCMOP and DTLZ, and the experimental results are shown in Table 3.

It can be seen from the experimental results that DPVAPS achieves the best IGD average on 15 out of 24 problems. And based on the Wilcoxon rank-sum test results, DPVAPS is significantly better than its variants on 9, 9, 7, and 11 problems

Table 3 The validation experiments of parameter δ_{min}

Problem	DPVAPS_20	DPVAPS_30	DPVAPS_40	DPVAPS_50	DPVAPS
LIRCMOP1	3.5838e−1 (1.07e−1) −	**2.9097e−1 (1.15e−1) =**	2.9635e−1 (1.08e−1) =	2.9314e−1 (1.02e−1) =	2.96798e−1 (9.20e−2)
LIRCMOP2	**2.0823e−1 (6.99e−2) =**	2.0963e−1 (9.44e−2) =	2.0872e−1 (8.29e−2) =	2.0935e−1 (8.65e−2) =	2.1026e−1 (9.22e−2)
LIRCMOP3	3.1073e−1 (7.24e−2) −	3.2142e−1 (9.57e−2) −	2.9065e−1 (1.08e−1) =	**2.7638e−1 (8.93e−2) =**	2.8250e−1 (9.08e−2)
LIRCMOP4	2.6374e−1 (8.77e−2) =	2.7543e−1 (6.88e−2) =	**2.3959e−1 (7.97e−2) =**	2.7081e−1 (1.07e−1) =	2.6383e−1 (7.84e−2)
LIRCMOP5	2.3923e−2 (1.44e−2) −	1.7690e−2 (5.50e−3) =	2.3783e−2 (1.72e−2) −	2.5845e−2 (2.11e−2) −	**1.6034e−2 (9.38e−3)**
LIRCMOP6	1.4438e−2 (3.83e−3) −	1.8056e−2 (1.06e−2) −	1.7165e−2 (1.29e−2) −	2.0038e−2 (1.01e−2) −	**1.2559e−2 (3.71e−3)**
LIRCMOP7	1.2378e−2 (5.19e−3) =	1.2193e−2 (4.95e−3) =	1.1991e−2 (4.04e−3) =	1.3812e−2 (5.38e−3) −	**9.8116e−3 (2.24e−3)**
LIRCMOP8	9.3777e−3 (2.56e−3) −	9.9412e−3 (4.57e−3) −	9.6097e−3 (2.96e−3) −	1.0109e−2 (4.08e−3) −	**8.6200e−3 (1.35e−3)**
LIRCMOP9	5.9915e−2 (4.76e−2) =	5.5194e−2 (4.99e−2) =	6.8503e−2 (5.14e−2) =	7.1875e−2 (7.09e−2) =	**5.3668e−2 (3.46e−2)**
LIRCMOP10	9.4669e−1 (9.26e−2) =	9.3139e−1 (8.52e−2) =	9.2988e−1 (8.24e−2) =	9.2096e−1 (1.18e−1) =	**6.8317e−3 (1.24e−3)**
LIRCMOP11	2.9300e−3 (8.97e−4) =	2.7151e−3 (3.44e−4) =	2.7498e−3 (3.64e−4) =	2.9376e−3 (5.08e−4) −	**2.5734e−3 (2.08e−4)**
LIRCMOP12	6.4582e−3 (5.44e−3) =	5.0822e−3 (2.77e−3) =	6.8953e−3 (5.77e−3) =	6.4573e−3 (4.55e−3) =	**4.2910e−3 (2.48e−3)**
LIRCMOP13	2.2645e−1 (2.84e−2) =	2.2194e−1 (3.07e−2) =	2.2322e−1 (2.95e−2) =	2.1617e−1 (3.74e−2) =	**9.1558e−2 (8.63e−4)**
LIRCMOP14	9.4514e−2 (1.01e−3) =	9.4651e−2 (7.65e−4) −	9.5294e−2 (1.02e−3) −	9.5281e−2 (1.13e−3) −	**9.4227e−2 (9.42e−4)**
C1_DTLZ1	1.9862e−2 (1.58e−4) −	1.9861e−2 (1.88e−4) −	1.9818e−2 (1.55e−4) =	1.9842e−2 (1.57e−4) =	**1.9782e−2 (1.58e−4)**
C1_DTLZ3	5.3481e−2 (5.67e−4) =	5.3935e−2 (7.94e−4) −	5.5729e−2 (1.17e−2) =	5.3666e−2 (6.04e−4) =	**5.3303e−2 (4.75e−4)**
C2_DTLZ2	**4.2290e−2 (3.91e−4) =**	4.2648e−2 (4.65e−4) −	4.2582e−2 (5.47e−4) −	4.2813e−2 (5.21e−4) −	4.2303e−2 (5.68e−4)
C3_DTLZ4	9.5879e−2 (1.51e−3) =	9.5784e−2 (1.38e−3) =	**9.5733e−2 (1.46e−3) =**	1.2080e−1 (1.36e−1) =	1.4459e−1 (1.89e−1)
DC1_DTLZ1	1.1448e−2 (1.19e−4) −	1.1427e−2 (8.35e−5) =	1.1443e−2 (1.37e−4) =	1.1456e−2 (1.34e−4) −	**1.1393e−2 (7.58e−5)**
DC1_DTLZ3	**1.1339e−1 (6.29e−4) =**	1.1390e−1 (1.31e−3) −	1.1370e−1 (1.15e−3) =	1.1385e−1 (8.83e−4) −	1.1364e−1 (9.02e−4)
DC2_DTLZ1	2.0085e−2 (1.63e−4) −	2.0092e−2 (1.36e−4) −	2.0193e−2 (1.69e−4) −	2.0201e−2 (1.52e−4) −	**1.9973e−2 (1.29e−4)**
DC2_DTLZ3	5.3422e−2 (1.85e−3) =	**5.3181e−2 (8.48e−4) =**	5.3209e−2 (4.20e−4) +	5.3845e−2 (2.26e−3) +	7.2399e−2 (9.33e−2)
DC3_DTLZ1	6.9041e−3 (1.43e−4) −	6.8684e−3 (1.15e−4) =	6.9352e−3 (1.66e−4) −	6.9709e−3 (2.22e−4) −	**6.8349e−3 (6.56e−5)**
DC3_DTLZ3	1.7339e−1 (8.10e−2) +	**1.6142e−1 (8.59e−3) =**	1.7438e−1 (8.05e−2) =	1.7442e−1 (8.10e−2) =	1.7800e−1 (6.84e−2)
+/−/=	1/9/14	0/9/15	1/7/16	1/11/12	

respectively, while it is only surpassed by its variants on 1, 0, 1, and 1 test peoblems. The reason is that when the size of the auxiliary population gradually decreases to 10, the main population will enjoy more computing resources to search CPF. While the auxiliary population will still give some help to the main population in the later stage, so it still needs to enjoy some resources. To sum up, it is quite appropriate when δ_{min} is 0.1.

Experimental results on MW and LIRCMOP problems

Table 4 shows the average and standard deviation of IGD values of the comparison algorithms on the five test sets, where the best result is marked in bold. Please note that if the algorithm cannot continuously find the feasible solutions on a problem, then only the feasibility rate is given in the form of "NAN (FR)". "+", "−", and "=" indicate that the compared algorithm is significantly better than, worse than, or comparable to the proposed algorithm, respectively. As can be seen from the Table 4, DPVAPS outperforms NSGA-II, CCMO, BiCo, PPS, and ToP on 24, 14, 16, 28, and 23 functions, respectively. In contrast, these compared algorithms outperform the proposed algorithm only on 0, 2, 3, 0, and 2 functions, respectively. From the perspective of FR, only CCMO and DPVAPS achieve 100% feasibility rate on all problems, while other algorithms cannot consistently find feasible solutions in 30 runs on some problems.

MW and LIRCMOP test suits have small and the discrete feasible regions, which requires a strong diversity of algorithms. NSGA-II does not outperform DPVAPS on any problem in these two test set, this is because NAGS-II uses only CDP to handle the constraints, and the population will quickly find one of the feasible regions, but this will cause the population to converge to this feasible domain instead of spreading to other regions, so its diversity cannot be guaranteed. CCMO and BiCo exhibit the similar performance with DPVAPS on some benchmark problems, the reason is that they all use the dual-population mechanism, whose auxiliary populations can provide beneficial information to the main population and help the main population find more feasible areas. This also proves the advantages of the dual-population mechanism. The auxiliary population of CCMO always consumes equal computational resources with the main population, so it does not have the superior performance exhibited by DPVAPS on most problems. The auxiliary population in BiCo uses an angle-based selection mechanism to ensure the diversity of the auxiliary population, so it performs well on some problems with small and multiple feasible regions, such as MW2, LIRCMOP1, and LIRCMOP3. But on the whole, DPVAPS performs better than BiCo. PPS and ToP are two-stage algorithms, the purpose of population is to find UPF in the first stage of PPS, while the second stage is to find CPF. But in these complex problems, there is a risk of the population falling into a local optimum when finding the UPF, which greatly affects the effect of the second stage. The first stage of ToP is to find feasible regions, and the second stage is to search CPF. However, in the first stage, there is a great probability to search the local feasible region rather than the optimal feasible region, so the optimal Pareto front cannot be found. For DPVAPS, the main population can be allocated more computing resources to search the CPF since the resources consumed by the auxiliary population gradually decreases. In addition, external archiving can increase the diversity of main population, therefore, the main population can be more evenly distributed over the CPF.

To sum up, the experimental results on the MW and LIRCMOP benchmark function sets can prove that the proposed algorithm has relatively better performance than other algorithms.

Experimental results on DTLZ, CTP, and DASCMOP problems

The specific IGD values of DPVAPS and five comparison algorithms on the DTLZ, CTP, and DASCMOP test sets are shown in Table 5. The results are blacked out once the algorithm achieves the optimal average IGD value on this problem. It can be seen that DPVAPS achieves the best average IGD values on 22 out of 27 test problems. In addition, DPVAPS performes significantly better than NSGA-II, CCMO, BiCo, PPS, and ToP on 23, 14, 22, 26, and 26 problems respectively based on the rank-sum test, while they only significantly outperform DPVAPS on 2, 1, 0, 0, and 1 problems, respectively.

The DTLZ test set has different constraint properties that can make the UPF become feasible, partially feasible, or completely infeasible. This makes the problems have different degrees of difficulty. Only CCMO and DPVAPS can consistently find feasible solutions on all peoblems in 30 runs. While the main population of DPVAPS can use more computing resources, so it can cover the CPF more completely. The dimensions of CTP test problems are relatively small, and their feasible regions are large, so they are relatively easy to solve. All algorithms can achieve the 100% feasible rates on these problems. The DASCMOP test set has many constraints, resulting in the large infeasible area blocks in the search space. This requires that the algorithm has the ability to cross the infeasible region. The auxiliary population of CCMO can ignore the constraints and reach the UPF. Furthermore, the information exchange in the environmental selection stage can help the main population cross the infeasible regions to reach the optimal feasible region, so it shows the superior effect. The auxiliary population of BiCo transforms the constraints into the objective, which still has the risk of falling into a local feasible domain, and thus its results are worse than those of CCMO and DPVAPS. PPS and ToP

Table 4 Mean (standard deviation) of IGD values of DPVAPS and five comparison algorithms on MW and LIRCMOP test sets

Problem	NSGA-II	CCMO	BiCo	PPS	ToP	DPVAPS
MW1	NaN (9.00e−1) −	1.6345e−3 (1.98e−5) −	5.4644e−3 (8.08e−3) −	NaN (8.67e−1) −	NaN (0.00e+0) −	**1.5909e−3(1.02e−5)**
MW2	3.2389e−2 (1.98e−2) −	2.3556e−2 (1.25e−2) =	**1.0036e−2 (6.02e−3) +**	2.9634e−2 (1.64e−2) −	NaN (8.00e−1) −	1.9144e−2 (9.60e−3)
MW3	9.7022e−3 (1.85e−2) −	5.2889e−3 (4.20e−4) =	5.1620e−3 (3.09e−4) =	1.6230e−2 (2.40e−2) −	NaN (5.00e−1) −	**5.1062e−3 (3.80e−4)**
MW4	5.5662e−2 (2.97e−3) −	4.1998e−2 (4.17e−3) −	4.1263e−2 (3.85e−4) −	5.6689e−2 (5.69e−3) −	NaN (0.00e+0) −	**4.0690e−2 (3.31e−4)**
MW5	2.1168e−1 (2.89e−1) −	2.8097e−3 (3.10e−3) −	3.1191e−2 (1.35e−1) −	2.7392e−1 (2.74e−1) −	NaN (0.00e+0) −	**2.3811e−3 (4.16e−3)**
MW6	1.2567e−1 (1.75e−1) −	4.0485e−2 (8.65e−2) −	3.9132e−2 (1.02e−1) =	1.0457e−1 (1.66e−1) −	NaN (9.33e−1) −	**1.9275e−2 (1.16e−2)**
MW7	5.4553e−2 (1.35e−1) −	**5.1119e−3 (7.56e−4) =**	6.8335e−3 (8.57e−3) =	2.7368e−2 (2.07e−2) −	2.1530e−1 (1.40e−1) −	5.1121e−3 (5.91e−4)
MW8	6.4071e−2 (2.33e−2) −	4.7680e−2 (5.72e−3) =	**4.4963e−2 (1.64e−3) =**	8.2722e−2 (5.54e−2) −	NaN (8.00e−1) −	4.5729e−2 (3.67e−3)
MW9	1.2872e−1 (2.48e−1) −	5.6063e−3 (2.16e−3) −	2.1822e−2 (9.37e−3) −	2.1671e−1 (3.04e−1) −	NaN (0.00e+0) −	**4.8624e−3 (7.03e−4)**
MW10	1.5152e−1 (1.20e−1) −	4.2421e−2 (7.25e−2) =	8.0248e−2 (1.18e−1) −	1.0685e−1 (1.40e−1) −	NaN (0.00e+0) −	**3.0728e−2 (1.88e−2)**
MW11	5.1295e−1 (3.10e−1) −	**6.3798e−3 (7.66e−4) =**	5.6502e−1 (2.57e−1) −	1.9956e−1 (2.95e−1) −	8.2763e−1 (9.35e−2) −	6.4509e−3 (5.35e−4)
MW12	7.0298e−2 (2.02e−1) −	**5.0827e−3 (3.79e−4) =**	1.0467e−2 (2.08e−2) =	2.7754e−1 (3.11e−1) −	NaN (6.67e−2) −	3.0639e−2 (1.40e−1)
MW13	4.7423e−1 (5.60e−1) −	7.8660e−2 (4.41e−2) −	1.4577e−1 (2.79e−1) =	1.6666e−1 (6.75e−2) −	NaN (8.67e−1) −	5.5424e−2 (3.59e−2)
MW14	1.2616e−1 (1.12e−2) −	9.9965e−2 (5.17e−3) =	**9.7694e−2 (2.61e−3) =**	2.0171e−1 (1.03e−1) −	5.8204e−1 (6.85e−1) −	9.8251e−2 (1.95e−3)
LIRCMOP1	2.6753e−1 (7.96e−2) =	2.4959e−1 (7.85e−2) +	**2.2566e−1 (6.18e−2) +**	NaN (7.00e−1) −	2.3774e−1 (8.30e−2) +	2.96798e−1 (9.20e−2)
LIRCMOP2	1.9821e−1 (6.61e−2) =	**1.6457e−1 (7.11e−2) +**	1.8656e−1 (6.09e−2) =	NaN (8.67e−1) −	1.9898e−1 (9.60e−2) =	2.1026e−1 (9.22e−2)
LIRCMOP3	2.4241e−1 (6.26e−2) =	2.4333e−1 (1.00e−1) =	**2.2775e−1 (7.45e−2) +**	NaN (9.67e−1) −	3.5986e−1 (9.43e−2) −	2.8250e−1 (9.08e−2)
LIRCMOP4	2.3880e−1 (6.00e−2) =	**2.2941e−1 (8.38e−2) =**	2.3851e−1 (6.74e−2) =	NaN (9.33e−1) −	3.2108e−1 (5.12e−2) −	2.6383e−1 (7.84e−2)
LIRCMOP5	8.2109e−1 (4.55e−1) −	2.9172e−2 (1.88e−2) −	9.8731e−1 (3.78e−1) −	1.8544e−1 (4.10e−2) −	4.3511e−1 (5.60e−1) =	**1.6034e−2 (9.38e−3)**
LIRCMOP6	7.5765e−1 (4.69e−1) −	2.0945e−2 (9.06e−3) −	7.9227e−1 (4.62e−1) −	2.4504e−1 (8.26e−2) −	3.5773e−1 (3.84e−1) −	**1.2559e−2 (3.71e−3)**
LIRCMOP7	2.2893e−2 (1.75e−2) −	1.6340e−2 (9.40e−3) −	2.1298e−2 (1.59e−2) −	7.8677e−2 (2.95e−2) −	**9.1205e−3 (3.98e−4) =**	9.8116e−3 (2.24e−3)
LIRCMOP8	1.1189e−1 (1.26e−1) −	1.5571e−2 (2.86e−2) =	4.4523e−2 (5.22e−2) −	1.0631e−1 (6.18e−2) −	2.3703e−1 (5.14e−1) −	**8.6200e−3 (1.35e−3)**
LIRCMOP9	5.6413e−1 (1.36e−1) −	1.0853e−1 (6.16e−2) −	5.9649e−1 (1.40e−1) −	2.9092e−1 (1.02e−1) −	3.6448e−1 (8.32e−2) −	**5.3668e−2 (3.46e−2)**
LIRCMOP10	4.0386e−1 (1.02e−1) −	7.2524e−3 (1.70e−3) =	5.6445e−1 (2.08e−1) −	1.1422e−1 (5.87e−2) −	**6.0888e−3 (4.55e−4) +**	6.8317e−3 (1.24e−3)
LIRCMOP11	2.6928e−1 (1.94e−1) −	3.4184e−3 (2.24e−3) −	3.3687e−1 (1.79e−1) −	4.3229e−2 (2.99e−2) −	1.7235e−1 (7.64e−2) −	**2.5734e−3 (2.08e−4)**
LIRCMOP12	1.9108e−1 (1.12e−1) −	1.2831e−2 (1.00e−2) −	2.0327e−1 (1.26e−1) −	7.5539e−2 (3.94e−2) −	5.1247e−2 (5.96e−2) −	**4.2910e−3 (2.48e−3)**
LIRCMOP13	1.5813e−1 (2.21e−1) −	9.3406e−2 (8.24e−4) −	9.3749e−2 (1.17e−3) −	1.1839e−1 (3.67e−3) −	1.3294e−1 (2.99e−2) −	**9.1558e−2 (8.63e−4)**
LIRCMOP14	1.2140e−1 (4.93e−3) −	9.6071e−2 (8.59e−4) −	9.5930e−2 (1.25e−3) −	1.1963e−1 (4.50e−3) −	1.1884e−1 (4.11e−3) −	**9.4227e−2 (9.42e−4)**
+/−/=	0/24/4	2/14/12	3/16/9	0/28/0	2/23/3	

Table 5 Mean (standard deviation) of IGD values of DPVAPS and five comparison algorithms on DTLZ, CTP, and DASCMOP test sets

Problem	NSGA-II	CCMO	BiCo	PPS	ToP	DPVAPS
C1_DTLZ1	2.7608e−2 (1.19e−3) −	1.9941e−2 (1.53e−4) −	2.0108e−2 (1.94e−4) −	2.6748e−2 (1.03e−3) −	NaN (0.00e+0) −	**1.9782e−2 (1.58e−4)**
C1_DTLZ3	5.7257e+0 (3.60e+0) −	5.3802e−2 (8.28e−4) −	5.1431e+0 (3.85e+0) −	1.0619e−1 (3.93e−2) −	1.6327e+0 (2.94e+0) −	**5.3303e−2 (4.75e−4)**
C2_DTLZ2	5.6929e−2 (2.87e−3) −	4.2811e−2 (5.54e−4) −	4.2688e−2 (5.30e−4) −	5.7604e−2 (2.35e−3) −	6.0993e−2 (1.19e−2) −	**4.2303e−2 (5.68e−4)**
C3_DTLZ4	1.2661e−1 (4.95e−3) +	**9.5454e−2 (1.28e−3) =**	1.2062e−1 (1.36e−1) =	2.2546e−1 (2.42e−1) −	1.4272e−1 (6.79e−3) +	1.4459e−1 (1.89e−1)
DC1_DTLZ1	1.4593e−2 (4.91e−4) −	1.1495e−2 (1.34e−4) −	1.1574e−2 (1.69e−4) −	1.8186e−2 (8.22e−4) −	6.0919e−2 (9.70e−2) −	**1.1393e−2 (7.58e−5)**
DC1_DTLZ3	1.2926e−1 (3.85e−3) −	1.1435e−1 (8.13e−4) −	1.1542e−1 (1.16e−3) −	1.5471e−1 (4.83e−2) −	2.0702e+0 (2.60e+0) −	**1.1364e−1 (9.02e−4)**
DC2_DTLZ1	NaN (0.00e+0) −	2.0264e−2 (1.56e−4) −	NaN (6.67e−2) −	6.8148e−2 (6.53e−2) −	NaN (0.00e+0) −	**1.9973e−2 (1.29e−4)**
DC2_DTLZ3	NaN (0.00e+0) −	**5.3945e−2 (1.47e−3) +**	NaN (0.00e+0) −	NaN (7.00e−1) −	NaN (0.00e+0) −	7.2399e−2 (9.33e−2)
DC3_DTLZ1	1.6972e−1 (1.10e−1) −	6.9285e−3 (1.44e−4) −	1.4038e−1 (1.21e−1) −	5.1759e−2 (1.00e−1) −	4.9939e+0 (4.80e+0) −	**6.8349e−3 (6.56e−5)**
DC3_DTLZ3	1.5971e+0 (4.78e−1) −	**1.7752e−1 (8.02e−2) =**	1.3010e+0 (4.22e−1) −	4.2474e−1 (2.20e−1) −	9.5632e+0 (3.96e+0) −	1.7800e−1 (6.84e−2)
CTP1	7.6593e−2 (7.64e−2) −	4.2415e−3 (1.05e−3) −	3.6673e−2 (4.14e−2) −	1.2634e−1 (6.60e−2) −	4.0236e−3 (1.42e−4) −	**3.6891e−3 (3.40e−4)**
CTP2	2.0994e−3 (1.62e−4) −	**1.5335e−3 (7.97e−5) =**	1.6853e−3 (1.10e−4) −	4.1494e−3 (1.77e−3) −	2.2607e−3 (1.26e−4) −	1.5413e−3 (6.93e−5)
CTP3	2.3269e−2 (2.95e−2) −	2.0439e−2 (2.61e−3) −	2.1555e−2 (2.13e−2) −	3.7644e−2 (1.54e−2) −	2.9856e−2 (5.15e−3) −	**1.8742e−2 (2.24e−3)**
CTP4	2.1992e−1 (1.59e−1) =	1.2317e−1 (1.57e−2) =	1.9076e−1 (1.23e−1) =	1.5782e−1 (5.24e−2) −	1.7366e−1 (3.35e−2) −	**1.2264e−1 (1.63e−2)**
CTP5	6.4738e−3 (3.09e−3) +	**7.5222e−3 (2.49e−3) =**	9.7007e−3 (5.08e−3) =	1.9185e−2 (7.75e−3) −	9.4284e−3 (2.80e−3) −	7.5597e−3 (2.33e−3)
CTP6	1.1411e−2 (3.78e−4) −	9.7973e−3 (3.71e−4) =	9.9246e−3 (1.26e−3) =	1.6823e−2 (1.72e−3) −	1.5334e−2 (1.41e−2) −	**9.7110e−3 (2.31e−4)**
CTP7	1.4625e−3 (5.99e−5) −	1.1503e−3 (9.33e−6) −	1.1535e−3 (6.43e−6) −	2.5645e−3 (2.19e−3) −	1.5178e−3 (6.24e−5) −	**1.1343e−3 (6.49e−6)**
CTP8	1.5320e−1 (1.50e−1) −	4.9141e−3 (3.98e−4) −	1.5761e−1 (2.11e−1) −	1.1616e−2 (2.31e−3) −	7.7339e−2 (1.96e−1) −	**4.6305e−3 (2.80e−4)**
DASCMOP1	7.4096e−1 (2.35e−2) −	7.1328e−1 (3.15e−2) =	7.3126e−1 (3.57e−2) −	7.2192e−1 (5.92e−2) =	7.8767e−1 (2.94e−2) −	**7.1314e−1 (3.50e−2)**
DASCMOP2	2.9807e−1 (3.29e−2) −	2.5012e−1 (2.90e−2) =	2.6567e−1 (3.14e−2) −	2.6701e−1 (3.65e−2) −	7.8374e−1 (3.97e−2) −	**2.5025e−1 (2.99e−2)**
DASCMOP3	3.5571e−1 (3.04e−2) =	3.5059e−1 (4.60e−2) =	3.5790e−1 (3.49e−2) =	3.7221e−1 (8.20e−2) −	7.4219e−1 (5.02e−2) −	**3.2437e−1 (4.11e−2)**
DASCMOP4	3.5698e−1 (1.04e−1) −	3.4093e−3 (2.93e−3) −	4.7293e−1 (1.94e−1) −	3.4550e−1 (2.25e−1) −	NaN (0.00e+0) −	**2.0739e−3 (1.27e−3)**
DASCMOP5	NaN (7.67e−1) −	3.5786e−3 (2.51e−3) −	2.4221e−1 (2.70e−1) −	1.0537e−1 (7.29e−2) −	NaN (0.00e+0) −	**2.9491e−3 (1.36e−4)**
DASCMOP6	NaN (7.33e−1) −	3.6882e−2 (2.78e−2) =	NaN (9.67e−1) −	2.9304e−1 (1.04e−1) −	NaN (0.00e+0) −	**3.3805e−2 (2.43e−2)**
DASCMOP7	5.4190e−2 (1.42e−2) −	3.0782e−2 (8.32e−4) =	5.3822e−2 (4.32e−2) −	1.5699e−1 (8.42e−2) −	NaN (0.00e+0) −	**3.0475e−2 (7.56e−4)**
DASCMOP8	7.8050e−2 (4.41e−2) −	4.0040e−2 (1.62e−3) =	7.4886e−2 (6.36e−2) −	4.9501e−1 (3.30e−1) −	NaN (0.00e+0) −	**3.9919e−2 (1.67e−3)**
DASCMOP9	4.1390e−1 (6.83e−2) −	3.8551e−1 (6.98e−2) −	4.0472e−1 (6.02e−2) −	6.5148e−1 (1.30e−1) −	7.1182e−1 (1.38e−1) −	**3.4350e−1 (7.13e−2)**
+/−/=	2/23/2	1/14/12	0/22/5	0/26/1	1/26/0	

Table 6 Results obtained by the Wilcoxon test

DPVAPS VS	R^+	R^-	P value	$\alpha = 0.05$
NSGA-II	1227.0	95.0	0	YES
CCMO	1059.5	381.5	0.00109	YES
BiCo	1164.5	210.5	0.000003	YES
PPS	1225.0	6.0	0	YES
ToP	624.0	23.0	0	YES

are two-stage algorithms. Since the search engine of PPS is replaced by GA, resulting in the diversity of the population is decreased, the population uses too much computational resources to search UPF in the first stage, and thus the population does not have enough computational resources to search the true CPF in the second stage. The auxiliary population of DPVAPS can reach the UPF quickly without considering any constraint, so it can help the main population to reach the optimal feasible region across the infeasible region. What's more, the main population can use more computing resources, and external archiving can help the main population search more feasible regions, so it shows excellent performance.

In conclusion, according to the results of the algorithms on DTLZ, CTP, and DASCMOP test suits, the performance of DPVAPS in solving simple or complex problems is better than that of the comparison algorithms.

Statistical results

In this section, all algorithms have carried out two statistical experiments on all 55 problems: Wilcoxon test and Friedman test. The experiments are conducted on KEEL software [41]. Table 6 shows the Wilcoxon test results. It can be seen that all R^+ are greater than R^-, indicating that DPVAPS is superior to the comparison algorithms. Furthermore, there are significant differences between the comparison algorithms and DPVAPS at the significance level $\alpha = 0.05$. Friedman test results are exhibited in the Fig. 7, the smaller the ranking value, the better the performance of the algorithm. It can be seen that the Rankings of the proposed algorithm is the smallest, which proves that its performance is superior to other algorithms. In a word, from the statistical results, the performance of DPVAPS is also the most superior compared with other comparison algorithms.

Figure 8 shows the convergence graphs of the average IGD values of DPVAPS and comparison algorithm for 30 runs on test questions LIRCMOP5 and LIRCMOP12. Among them, the LIRCMOP5 has the large infeasible regions, which block the evolution of the population. Similarly, the LIRCMOP12 also has the large infeasible regions, and its CPF is discrete. It can be seen from Fig. 8 that CCMO and DPVAPS achieve similar performance on the test problem LIRCMOP5, while

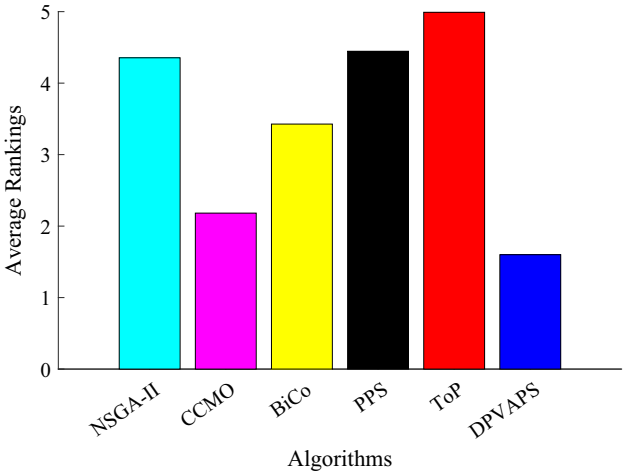

Fig. 7 Average rankings of all six methods obtained by the Friedman test on all 55 functions

DPVAPS achieves the fastest convergence speed on the test problem LIRCMOP12. In a word, DPVAPS has faster convergence speed compared with other algorithms.

Conclusion

This paper proposes a novel dual-population algorithm to solve CMOPs. The main population is responsible for searching for CPF, and the auxiliary population is responsible for searching for UPF. In addition, In order to reduce the waste of computing resources of the auxiliary population in the later stage, a dynamic population size reducing mechanism is designed to change the size of the auxiliary population. Furthermore, an external archive is used to store feasible solutions found by the auxiliary population, which can provide more new feasible solutions for the main population, thereby increasing the diversity of the main population on CPF. In the experimental stage, the effectiveness of the two mechanisms, the dynamic population size reducing mechanism and external archive, are verified. The rationality of parameter δ_{min} used to control population size is also investigated. Moreover, the experimental results on 55 test problems prove that the proposed algorithm is promsing in solving CMOPs. However, the developed algorithm still has some limitations: (1) the size of auxiliary population decreases according to the same change rule on different problems. Therefore, a population size reduction mechanism based on the type of problems should be designed to deal with different problems; (2) the auxiliary population movement mechanism can be designed so that it can approach the CPF from the infeasible region side and explore the CPF with the main population. In the future, DPVAPS will be further improved to solve these problems, and can be applied to solve practical problems.

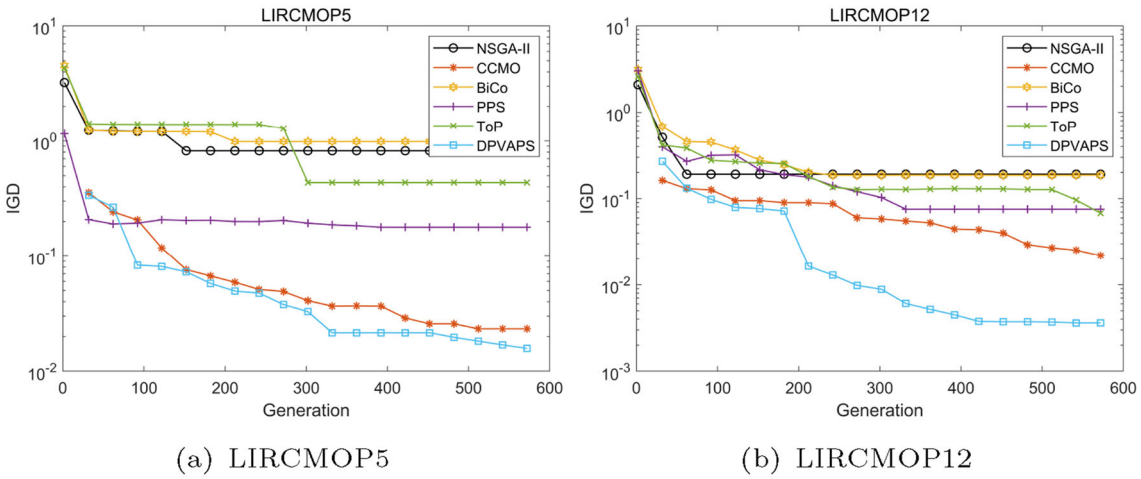

Fig. 8 Convergence graphs of average IGD values obtained by 30 runs of DPVAPS and other comparison algorithms on two representative functions

Acknowledgements This work was supported in part by the National Natural Science Fund for Outstanding Young Scholars of China (61922072), National Natural Science Foundation of China (62176238, 61806179, 61876169, 62106230, and 61976237), China Postdoctoral Science Foundation (2020M682347, 2021T140616, 2021M692920), Key Research and Development and Promotion Projects in Henan Province (192102210098), and Training Program of Young Backbone Teachers in Colleges and Universities in Henan Province (2020GGJS 006).

References

1. He C et al (2022) A self-organizing map approach for constrained multi-objective optimization problems. Complex Intell Syst 8:5355–5375
2. Qiao K et al (2022) Feature extraction for recommendation of constrained multi-objective evolutionary algorithms. IEEE Trans Evol Comput. https://doi.org/10.1109/TEVC.2022.3186667
3. Long W et al (2022) A constrained multi-objective optimization algorithm using an efficient global diversity strategy. Complex Intell Syst. https://doi.org/10.1007/s40747-022-00851-1
4. Chen Y et al (2022) Constraint multi-objective optimal design of hybrid renewable energy system considering load characteristics. Complex Intell Syst 8(2):803–817
5. Liang J et al (2022) A survey on evolutionary constrained multi-objective optimization. IEEE Trans Evol Comput. https://doi.org/10.1109/TEVC.2022.3155533
6. Du K-J, Li J-Y, Wang H, Zhang J (2022) Multi-objective multi-criteria evolutionary algorithm for multi-objective multi-task optimization. Complex Intell Syst. https://doi.org/10.1007/s40747-022-00650-8
7. Ishibuchi H, Nojima Y et al (2017) On the effect of normalization in moea/d for multi-objective and many-objective optimization. Complex Intell Syst 3(4):279–294
8. Sun Y, Yen GG, Yi Z (2018) Igd indicator-based evolutionary algorithm for many-objective optimization problems. IEEE Trans Evol Comput 23(2):173–187
9. Liang J, Ban X, Yu K, Qiao K, Qu B (2022) Constrained multiobjective differential evolution algorithm with infeasible-proportion control mechanism. Knowl-Based Syst 250:109105
10. Qiao K et al (2022) Dynamic auxiliary task-based evolutionary multitasking for constrained multi-objective optimization. IEEE Trans Evol Comput. https://doi.org/10.1109/TEVC.2022.3175065
11. Yu K, Liang J, Qu B, Yue C (2021) Purpose-directed two-phase multiobjective differential evolution for constrained multiobjective optimization. Swarm Evol Comput 60:100799
12. Lin C-H (2013) A rough penalty genetic algorithm for constrained optimization. Inf Sci 241:119–137
13. Tessema B, Yen GG (2009) An adaptive penalty formulation for constrained evolutionary optimization. IEEE Trans Syst Man Cybern Part A Syst Hum 39(3):565–578
14. Yu K, Liang J, Qu B, Luo Y, Yue C (2021) Dynamic selection preference-assisted constrained multiobjective differential evolution. IEEE Trans Syst Man Cybern Syst 52(5):2954–2965
15. Ning W et al (2017) Constrained multi-objective optimization using constrained non-dominated sorting combined with an improved hybrid multi-objective evolutionary algorithm. Eng Optim 49(10):1645–1664
16. Yang Y, Liu J, Tan S, Wang H (2019) A multi-objective differential evolutionary algorithm for constrained multi-objective optimization problems with low feasible ratio. Appl Soft Comput 80:42–56
17. Liu Z-Z, Wang Y, Wang B-C (2019) Indicator-based constrained multiobjective evolutionary algorithms. IEEE Trans Syst Man Cybern Syst 51(9):5414–5426
18. Fan Z et al (2019) Push and pull search for solving constrained multi-objective optimization problems. Swarm Evol Comput 44:665–679
19. Yu X, Lu Y (2018) A corner point-based algorithm to solve constrained multi-objective optimization problems. Appl Intell 48(9):3019–3037

20. Ming F et al (2021) A simple two-stage evolutionary algorithm for constrained multi-objective optimization. Knowl-Based Syst 228:107263
21. Tian Y, Zhang T, Xiao J, Zhang X, Jin Y (2020) A coevolutionary framework for constrained multiobjective optimization problems. IEEE Trans Evol Comput 25(1):102–116
22. Li K, Chen R, Fu G, Yao X (2018) Two-archive evolutionary algorithm for constrained multiobjective optimization. IEEE Trans Evol Comput 23(2):303–315
23. Liu Z-Z, Wang B-C, Tang K (2021) Handling constrained multiobjective optimization problems via bidirectional coevolution. IEEE Trans Cybern
24. Jan MA, Tairan N, Khanum RA (2013) Threshold based dynamic and adaptive penalty functions for constrained multiobjective optimization. In: IEEE, pp 49–54
25. Jiao L, Luo J, Shang R, Liu F (2014) A modified objective function method with feasible-guiding strategy to solve constrained multiobjective optimization problems. Appl Soft Comput 14:363–380
26. Ma Z, Wang Y, Song W (2019) A new fitness function with two rankings for evolutionary constrained multiobjective optimization. IEEE Trans Syst Man Cybern Syst
27. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197
28. Runarsson TP, Yao X (2000) Stochastic ranking for constrained evolutionary optimization. IEEE Trans Evol Comput 4(3):284–294
29. Takahama T, Sakai S (2010) Efficient constrained optimization by the ε constrained adaptive differential evolution. In: IEEE, pp 1–8
30. Liu Z-Z, Wang Y (2019) Handling constrained multiobjective optimization problems with constraints in both the decision and objective spaces. IEEE Trans Evol Comput 23(5):870–884
31. Tian Y et al (2021) Balancing objective optimization and constraint satisfaction in constrained evolutionary multiobjective optimization. IEEE Trans Cybern
32. Liang J et al (2022) Utilizing the relationship between unconstrained and constrained pareto fronts for constrained multiobjective optimization. IEEE Trans Cybern. https://doi.org/10.1109/TCYB.2022.3163759
33. Qiao K et al (2022) An evolutionary multitasking optimization framework for constrained multiobjective optimization problems. IEEE Trans Evol Comput 26(2):263–277
34. Deb K, Agrawal RB et al (1995) Simulated binary crossover for continuous search space. Complex Syst 9(2):115–148
35. Deb K, Goyal M et al (1996) A combined genetic adaptive search (geneas) for engineering design. Comput Sci Informat 26:30–45
36. Ma Z, Wang Y (2019) Evolutionary constrained multiobjective optimization: test suite construction and performance comparisons. IEEE Trans Evol Comput 23(6):972–986
37. Fan Z et al (2019) An improved epsilon constraint-handling method in moea/d for cmops with large infeasible regions. Soft Comput 23(23):12491–12510
38. Fan Z et al (2020) Difficulty adjustable and scalable constrained multiobjective test problem toolkit. Evol Comput 28(3):339–378
39. Bosman PA, Thierens D (2003) The balance between proximity and diversity in multiobjective evolutionary algorithms. IEEE Trans Evol Comput 7(2):174–188
40. Tian Y, Cheng R, Zhang X, Jin Y (2017) Platemo: a matlab platform for evolutionary multi-objective optimization [educational forum]. IEEE Comput Intell Mag 12(4):73–87
41. Alcalá-Fdez J et al (2009) Keel: a software tool to assess evolutionary algorithms for data mining problems. Soft Comput 13(3):307–318