Learning-Aided Evolutionary Search and Selection for Scaling-up Constrained Multiobjective Optimization

Songbai Liu, Zeyi Wang, Qiuzhen Lin, *Member, IEEE*, Jianqiang Li, Kay Chen Tan, *Fellow, IEEE*

***Abstract*—The existing constrained multiobjective evolutionary algorithms (CMOEAs) still have great room for improvement in balancing population's convergence, diversity and feasibility on complex constrained multiobjective optimization problems (CMOPs). Besides, their effectiveness deteriorates dramatically when facing the CMOPs with scaling-up objective space or search space. We are thus motivated to design a learning-aided CMOEA with promising problem-solving ability and scalability for various CMOPs. In the proposed solver, two learning models are respectively trained online on constrained-ignored task and feasibility-first task, which are then used to learn the two improvement-based vectors for enhancing the search by differential evolution. In addition, the union population of parent and child solutions is divided into multiple subsets with a hierarchical clustering based on cosine similarity. A comprehensive indicator, considering objective-based performance and constraint violation degree of a solution, is developed to select the representative solution from each cluster. The effectiveness of the proposed optimizer is verified by solving the CMOPs with various irregular Pareto fronts, the number of objectives ranging from 2 to 15, and the dimensionality of search space scaling up to 1000.**

***Index Terms*—Evolutionary algorithm, Constrained Multiobjective Optimization, Learning-aided Search and Selection.**

I. Introduction

Most multiobjective optimization problems (MOPs) in real-world applications, e.g., path planning [1], urban bus scheduling [2], and engineering design [3], often subject to various constraints. Mathematically, a constrained MOP (CMOP) can be formulated as:

$$\begin{aligned} &\text{Minimize } F(x) = (f_1(x), f_2(x), ..., f_m(x)) \\ &\text{s.t. } \begin{cases} g_i(x) \le 0, i = 1, ..., l \\ h_j(x) = 0, j = 1, ..., k \end{cases}, \end{aligned} \tag{1}$$

where $x = (x_1, x_2, \ldots, x_n)$ is a solution vector to define the m objective functions $f_1(x), \ldots, f_m(x)$, the l inequality constraints $g_{i=1:l}(x) \le 0$, and the k equality constraints $h_{j=1:k}(x) = 0$. In solving CMOPs, the feasible solutions must satisfy all constraints in addition to optimizing the often mutually conflicted objectives [4]. Their optimal solutions in the search space are represented by a trade-off mirror curve known as constrained Pareto front (CPF) in the objective space. This CPF is composed of those feasible solutions that are not dominated by any other feasible solutions. In contrast, unconstrained Pareto front (UPF) is a counterpart mirror curve obtained without considering constraints. Therefore, solving CMOPs is generally more challenging than addressing unconstrained MOPs (UMOPs) due to the additional complexity of satisfying constraints.

Multiobjective evolutionary algorithms (MOEAs) are specialized tools for solving MOPs. They employ population-based heuristic search to explore a diverse set of candidate solutions, avoiding premature convergence to a single corner of the Pareto front [5]. MOEA-optimizers, categorized as decomposition-based, Pareto-based, or indicator-based solvers [6], excel at addressing UMOPs, covering the UPFs. However, when applied to CMOPs, they encounter challenges. In navigating CPFs, they risk being misled or obstructed by infeasible solutions [7]. Addressing CMOPs requires a nuanced equilibrium between achieving convergence, maintaining diversity, and ensuring feasibility within the evolving population throughout the evolutionary process. This highlights the need for constrained MOEAs (CMOEAs) to effectively address CMOPs. Existing CMOEAs are categorized based on their constraint handling techniques (CHTs), which typically fall into four main categories:

The first type of CHTs is feasibility-guided discriminant criteria (FDC) for fitness (or quality) comparison of solutions [8]-[22]. This type of CHTs usually takes solution's feasibility (e.g., constraint violation (CV) degree) into account on the basis of the existing criteria, such as constrained domination relationship, feasibility-driven scalarization functions, performance indicators that incorporate feasibility, etc. However, the strong focus on feasibility may inadvertently lead the evolutionary population to persist on the deceptive Pareto front (DPF) or get trapped in easily discoverable part of the CPF. In this paper, the deceptive feasible regions (bounded by the DPF) are defined as separate from the target feasible regions (bounded by the CPF), where each solution in the former is dominated by at least one solution on the CPF.

The second type of CHTs is constraint-pruned problem transformation (CPT) from a CMOP to a UMOP [23]-[30]. This type of CHTs aims to tailor constraints as additional objectives or as penalty terms of objectives. In general, the CV degree and feasible rate of solutions are the components involved in the pruning process. A sophisticated MOEA is then selected to solve the retailored UMOP. However, ensuring that the constructed UMOP aligns with the target CMOP's optima

This work is partially supported by the National Natural Science Foundation of China (NSFC) under Grants 62306180, 62173236, and U21A20512, the National Natural Science Founds for Distinguished Young Scholar under Grant 62325307, the Natural Science Foundation of Guangdong Province under Grant 2023A1515011238, the Shenzhen Science and Technology Program under Grant JCYJ20220531101411027, the Research Grants Council of the Hong Kong SAR under Grants PolyU11211521, PolyU15218622 and PolyU15215623, and the Hong Kong Polytechnic University (Project IDs: P0039734 and P0035379). (*Corresponding authors*: *Q. Lin and J. Li*).

S. Liu, Z. Wang, Q. Lin, and J. Li are with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China (Email: songbai@szu.edu.cn).

K.C. Tan is with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong SAR (Email: kctan@polyu.edu.hk).

poses challenges. Therefore, they often require a cumbersome parameter tuning to determine an appropriate penalty coefficient or degree of constraint relaxation. Besides, the introduction of less-understood additional objectives presents new and tricky challenges (e.g., many-objective optimization problem with irregular Pareto fronts) to the corresponding optimizer.

The third type of CHTs is collaborative optimization (CO) of objectives and constraints with multiple populations or archives [31]-[44]. In general, one or more auxiliary optimization tasks are firstly constructed to assist the optimization of the target CMOP (i.e., the main task), followed by evolving a personalized population on each task. The most typical auxiliary task is the constraint-ignored counterpart of the target CMOP. Another common way to create auxiliary task is the constraint-relaxed formulation. After that, these parallel populations interact with each other to improve the effectiveness of evolutionary search and selection. This type of CHTs is relatively popular in designing CMOEAs, especially with the rise of multitask or transfer optimization recently [45]-[46]. By driving multiple population, they can maintain a superior diversity and feasibility of candidate solutions. Nevertheless, their performance strongly depends on the relationship between the auxiliary task and the main task. The auxiliary task may mislead the evolutionary search for the main task when the obtained optimal solution sets between them differ greatly. In addition, the optimization of helpless auxiliary tasks results in a waste of computing resources.

The fourth type of CHTs is alternate preference (AP) between constraints and objectives at multiple stages [47]-[61]. This type of CHTs divides the evolutionary process into two or more stages with different purposes, e.g., to make paired trade-offs among diversity, convergence, and feasibility. It achieves this by alternately favoring specific optimization tasks, like constraint-ignored and constraint-relaxed tasks, at different stages. Properly executed stage switches enable this CHT to cross the obstacles of the deceptive feasible regions and infeasible regions. However, it may lead to a wasteful use of computing resources when staying too long in the constraint-ignored stage to push the population to the UPF that is far away from CPF. Conversely, inadequate resource allocation to this stage may pull the population back into the deceptive feasible region, hindering its progress towards the target feasible region. Hence, the proper switching of stages and the rational allocation of computing resources for each stage still require further studies.

To sum up, these four types of CHTs have their own limitations, more details of which will be provided in Section II. In addition, most existing CHTs rarely take their scalability into account, which leads to a corresponding deterioration of their performance when the objective space or search space keeps scaling up [62]. To alleviate these issues, we attempt to develop a learning-aided CMOEA (LCMOEA) with an enhanced ability in balancing population's diversity, convergence, and feasibility, meanwhile, with a considerable scalability in solving high-dimensional CMOPs. The proposed LCMOEA has the following three main contributions or advantages:

(1) A learnable differential evolution operator is developed to search for promising solutions with the guidance of two improvement vectors. They are learned by two well-trained multilayer perceptron models respectively for constraint-ignored task and feasibility-first task.

(2) A clustering-aided environmental selection is customized to get a population with a trade-off (convergence, diversity, feasibility). A comprehensive indicator considering both convergence and feasibility is proposed to guide the representative selection in each cluster.

(3) The collaboration of learnable reproduction and clustering-aided selection makes the optimizer has an outstanding scalability in solving CMOPs with scaling up objective space and search space.

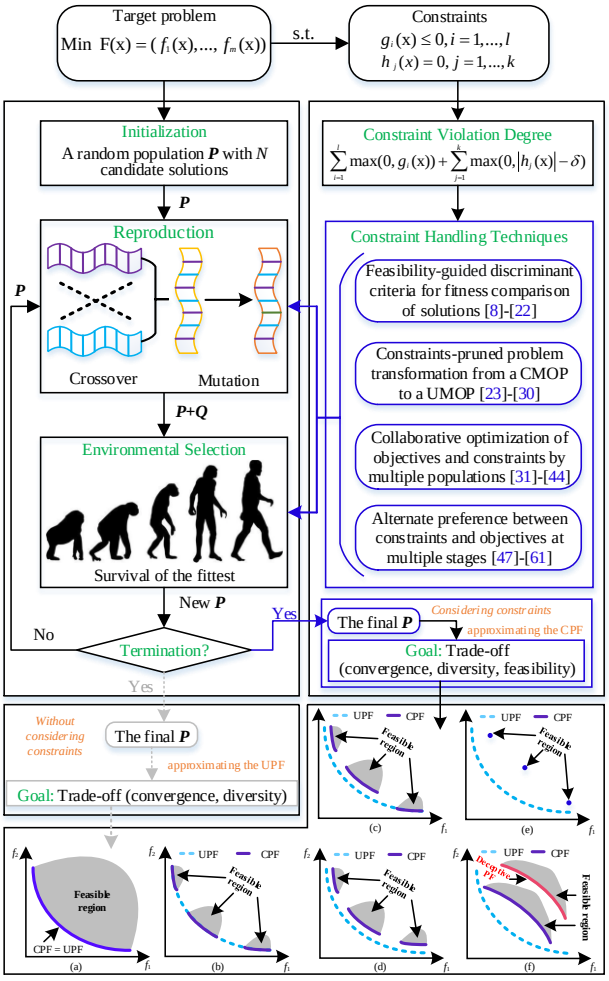


Fig. 1 Illustration of the general process of CMOEAs.

The remainder of this paper is organized as follows. Section II gives a brief review of existing CMOEAs and discusses their pitfalls that motivate our research. Section III provides the general framework and algorithmic components of LCMOEA. Section IV implements experimental studies on LCMOEA in solving various CMOPs. Section V summarizes this paper and analyzes the possible future extension work.

II. Related Work and Motivations

In general, the flow of a CMOEA can be depicted in Fig. 1. Given a target CMOP, it starts with an initial parent population

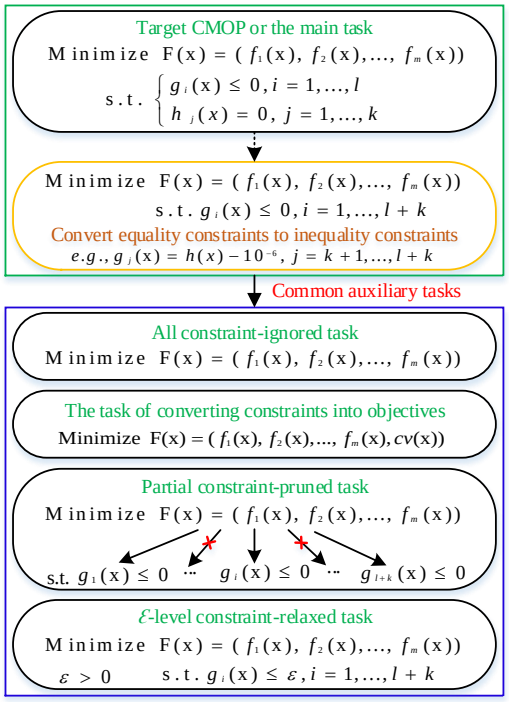


Fig. 2 Illustration of the common auxiliary tasks for the target CMOP.

P. Then, an offspring population *Q* is generated by inputting *P* into a reproduction model to search with evolutionary operators, e.g., crossover and mutation. Subsequently, both *P* and *Q* undergo evaluation in an environmental selection model. The model assesses the quality of solutions, updating *P* by retaining the superior half of solutions from the combined *P* and *Q*. This iterative process, akin to that of an MOEA, continues until the specified termination condition is met. The distinctive feature of a CMOEA lies in its requirement for the target problem to conform to a set of constraints, and thus corresponding CHTs should be designed to cope with the challenges caused by the violation of constraints. In contrast to MOEAs, the goal of a CMOEA is to obtain a promising population that strikes a balance among convergence, diversity, and feasibility in approximating the CPF. Next, we will start with a brief review of existing CMOEAs based on their used CHTs. We then analyze the various challenges caused by satisfying constraints from the six CPF cases as depicted in Fig. 1. Finally, we will elucidate the motivation of this work.

A. Brief Review of Existing CMOEAs

We start by reviewing the CMOEAs based on the first type of CHTs, i.e., FDC-based CMOEAs. which always follow a feasibility-first principle to redefine existing selection criteria. For example, the CV degree of solutions is given priority in redefining their domination relationship [8], forming the well-known constraint domination principle (CDP). Specifically, a solution x is said to dominate another solution y following the ε-level CDP [9], denoted as $\mathrm{x} \prec_{\varepsilon}^{CDP} \mathrm{y}$, can be defined as:

$$\mathrm{x} \prec_{\varepsilon}^{CDP} \mathrm{y} \Leftarrow \begin{cases} [cv(\mathrm{x}) \le \varepsilon \cap cv(\mathrm{y}) \le \varepsilon \cap \mathrm{x} \prec \mathrm{y}] \\ \cup \quad [cv(\mathrm{x}) = cv(\mathrm{y}) \cap \mathrm{x} \prec \mathrm{y}], \\ \cup \quad [cv(\mathrm{x}) < cv(\mathrm{y})] \end{cases} \tag{2}$$

where ε is a tolerated level to relax the CV degree of solutions, and its value is gradually decreased as the evolutionary process goes on. $\mathrm{x} \prec \mathrm{y}$ indicates x dominate y following the traditional unconstrained domination principle [8]. Besides, the feasibility of x can be reflected by its CV degree $cv(\mathrm{x})$, which is defined as follows,

$$cv(\mathrm{x}) = \sum_{i=1}^{l} \max(0, g_i(\mathrm{x})) + \sum_{j=1}^{k} \max(0, |h_j(\mathrm{x})| - \delta), \tag{3}$$

where δ is a very small positive number used to convert equality constraints to inequality constraints. So, x is a feasible solution only if its $cv(\mathrm{x})$ equal to 0. Due to its simplicity, CDP is widely embedded into various MOEAs (e.g., MOEA/D [10], NSGA [11], RVEA [12], and HypE [13]) for solving CMOPs. Following efforts include angle-based CDP [14], strictly constrained dominance [15], constrained θ-dominance principle [16], and adaptive CDP [17]. Besides, the fitness comparison of x and y in stochastic ranking (SR) [18] is controlled by a probability parameter $\rho \in [0,1]$. In particular, x is said to dominate y in SR, termed $\mathrm{x} \prec_{SR} \mathrm{y}$, can be defined as:

$$\mathrm{x} \prec_{SR} \mathrm{y} \Leftarrow \begin{cases} [\mathrm{x} \prec_{\varepsilon}^{CDP} \mathrm{y} \cap r \le \rho] \\ \cup \ [\mathrm{x} \prec \mathrm{y} \cap r > \rho] \end{cases}, \tag{4}$$

where r is a random number within [0,1]. Moreover, new performance indicators guided by constraints are proposed to handle CMOPs, including constrained crowding distance, constrained shift-based density estimation [19], and cost value -based distance [20]. A new indicator concurrently considering the convergence, diversity, and feasibility of a population is proposed in [21]. The tchebycheff aggregation in [22] is redefined by considering the CV degree of solutions.

We then turn to review the CMOEAs using the second type of CHTs, i.e., CPT-based CMOEAs, in which constraints are tailored as additional objectives or as penalty terms of objectives. For instance, the overall CV function is directly treated as an additional objective in [23]-[25]. In this way, a new MOP is defined as follow,

$$\text{Minimize } \mathrm{F}(\mathrm{x}) = (f_1(\mathrm{x}), f_2(\mathrm{x}), ..., f_m(\mathrm{x}), cv(\mathrm{x})), \tag{5}$$

Moreover, the target CMOP in [26] and [27] is transformed into a new UMOP with three special objectives (or goals), which are elaborately customized to quantify the feasibility, diversity, and convergence of a solution, respectively. In [28] and [29], each objective $f(\mathrm{x}) \in \mathrm{F}(\mathrm{x})$ is redefined by considering the CV and feasible rate. The reformulated CMOP in [30] is a problem with the same objectives as in (5) but with a dynamic constraint boundary ε, which can be defined as follow,

$$\begin{aligned} &\text{Minimize } \mathrm{F}(\mathrm{x}) = (f_1(\mathrm{x}), f_2(\mathrm{x}), ..., f_m(\mathrm{x}), cv(\mathrm{x})) \\ &\text{s.t. } g_i(\mathrm{x}) \le \varepsilon_i, i = 1, ..., l \end{aligned}, \tag{6}$$

where ε updates adaptively as evolution progresses.

Our review is moved to the CMOEAs with the third type of CHTs, i.e., CO-based CMOEAs, which use multiple personalized populations collaboratively to solve different tasks. The common auxiliary tasks of the target CMOP are outlined in Fig. 2. The all constraint-ignored task is considered in CTAEA [31], DBC-CMOEA [32], CCMO [33], URCMO [34], and DBEMTO [35]. In these optimizers, one population considers constraints, while the other disregards them, evolving jointly to enhance the efficiency of reproduction and environmental

selection. In particular, TPEA [36] employs two auxiliary populations composed of the innermost and outermost infeasible nondominated solutions, essentially addressing two extreme all constraint-ignored tasks. Besides, EMCMO [37] and MCCMO [38] introduce partial constraint-pruned tasks, facilitating adaptive knowledge transfer between different tasks. In addition, ε-level constraint-relaxed tasks are considered in CMOCSO [39], MFO [40], dp-ACS [41], and MTCMO [42], where specific knowledge transfer and update strategies are designed for solving the respective tasks. Here, task generation involves adjusting the value of ε. TriP [43] and CMOEMT [44] customize both constraint-ignored and constraint-relaxed tasks to enhance diversity and leverage knowledge acquired from different task types.

We finally dive into the CMOEAs that involve multiple stages to alternatively switch their optimization preferences, i.e., AP-based CMOEAs. A prevalent instance is the dual-stage pattern, often referred to as the push-and-pull search (PPS) pattern [47]. Without considering constraints, the initial stage prioritizes objectives to push the population towards UPF. Subsequently, in the second stage, the focus shifts to constraints, pulling the population back towards the CPF. Representative CMOEAs based on this idea include CMOES [48], C-TSEA [49], DD-CMOEA [50], PDTP-MDE [51], NSGA-II -P [52], PPTA [53], ARCMO [54], and TS-NSGA-II [55]. To achieve the corresponding purpose, the evolutionary operators taken in the two stages have their own characteristics. Another pattern is to divide the evolutionary process into multiple stages based on the number of constraints (or the level of constraint relaxation). The optimization task starts with zero constraints in the initial stage, incrementally adding one constraint upon each transition to the subsequent stage until incorporating all constraints. Representative CMOEAs following this idea include MSCOM [56] and C3M [57]. In addition, the evolutionary process can be divided into multiple stages based on the population state detection, e.g., determining whether the population is experiencing stagnation or converging into a feasible subregion. This allows for adaptive alternation of preferred tasks upon transitioning to a new stage. Representative solvers include MOEA/D-DAE [58], PKAEO [59], CMOEA-DPMS [60], and DSPCMDE [61].

B. Challenges Caused by Satisfying Constraints

Examining the interplay between the UPF and CPF, we discern six distinctive cases, each presenting unique challenges when transitioning from an MOP to a CMOP, as illustrated in Fig. 1.

Case 1: Identical UPF and CPF. In this scenario, solving the CMOP may not inherently be more challenging than its counterpart MOP. The constraints, providing valuable information such as a feasibility-driven search direction, can expedite population convergence.

Case 2: CPF comprising few separate segments on UPF. Challenges emerge in preserving population diversity when the CPF consists of isolated segments. FDC-based CMOEAs may struggle, tending to be confined to one segment due to their lesser emphasis on diversity. The search might be easily stuck at the relatively easy-to-find segments of the CPF.

Algorithm 1 General Framework of LCMOEA

Input: the target CMOP(m, n) and the $FE_{\max}$
Output: the final population P

```
initialize P with N random solutions of CMOP, FE = 0
while FE ≤ FE_max do
    (M_1, M_2) ← Training(P)                          //Algorithm 2
    Q ← LearnableReproduction(M_1, M_2, P)            //Algorithm 3
    P ← ClusteringAidedSelection(P, Q)                //Algorithm 4
    FE ← FE + N
end while
return the P
```

Case 3: Part of CPF on UPF, part separated. In this case, irregular shapes form on the CPF, necessitating increased efforts to maintain population diversity.

Case 4: Completely separated CPF and UPF. The infeasible region between CPF and UPF (especially with a large size) will pose a challenge to balance convergence and feasibility. The constraint-ignored tasks in CO-based MOEAs may not only provide negative aid to the target task, but also cause a waste of computing resources.

Case 5: Extremely small size of CPF (or feasible space). Locating a feasible solution becomes notably challenging, presenting a significant hurdle to the evolutionary search and selection process.

Case 6: CPF is sandwiched between UPF and DPF. The population may encounter hindrances from both deceptive feasible and infeasible regions. In AP-based CMOEAs, the population will be likely converged to DPF once the solver cannot push it pass the infeasible or deceptive regions at the stage of preferring objectives.

Understanding these cases provides valuable insights into the distinctive challenges posed by CMOPs and informs the strategic design of effective CHTs.

C. Motivations.

The discussions above highlight the challenges posed by CMOPs to existing MOEAs. Initially, constraints lead to infeasibility across the majority of the search space, acting as substantial barriers impeding the evolutionary population's progression towards the CPF. This complicates the task of guiding MOEAs to identify small feasible regions within the vast expanse of infeasible space. In addition, constraints introduce deceptive feasible regions, presenting a significant obstacle for the optimizer in sustaining population convergence. Concretely, the solver may only push the population to converge to the DPF, or it may push too far to pull the population from the UPF back to the CPF. Finally, constraints contribute to irregularities in the CPF compared to its UPF counterpart, making it challenging for the solver to maintain population diversity. Essentially, the optimizer may be limited to search for solutions in the easiest-to-find feasible region.

To address these challenges, existing CMOEAs have made considerable efforts to balance the convergence, diversity, and feasibility. Nevertheless, there remains significant room for improvement in CMOEA design, particularly when solving CMOPs featuring scaling up objectives and search spaces. As the decision variables or objectives scale up, the effectiveness

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2024.3380366

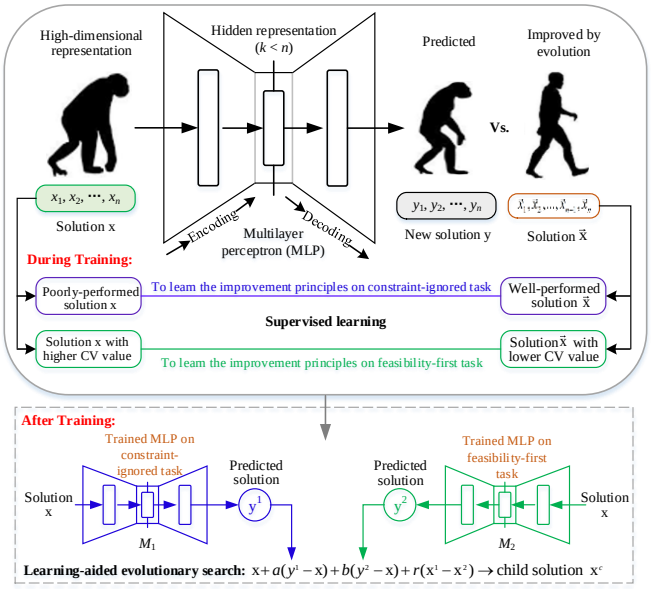


Fig. 3 Illustration of the basic idea of learnable differential evolution.

of existing CMOEAs may degenerate dramatically due to the "curse of dimensionality". For example, the selection pressure considerably decreases as the scaling up of the objective space, especially pronounced in the presence of infeasible and deceptive obstacles due to constraints. Maintaining diversity becomes increasingly challenging in CMOPs with highly irregular shapes as the objective space scales up, and the efficiency of existing evolutionary search strategies declines sharply in larger search spaces.

Therefore, we are motivated to improve the scalability of CMOEAs in solving scaling up CMOPs by improving their evolutionary search and environmental selection. To improve the efficiency of evolutionary optimization, researchers have investigated combining MOEAs with machine learning models to create adaptable evolutionary frameworks [75]. They use machine learning to identify key features that differentiate subpar solutions from desirable ones. This gained insight then guides the creation of new solutions tailored to meet desired criteria, ultimately enhancing the overall quality of the population. When facing CMOPs with scaling up search space, we expect to train a supervised learning model to steer evolutionary search in the promising direction, so that the population can quickly converge to the target feasible regions. When solving CMOPs with scaling up objective space, we expect to design a clustering-aided CHTs for guiding the environmental selection with balanceable convergency, diversity, and feasibility. The collaboration between learnable search and selection in the proposed LCMOEA equips it with robust problem-solving ability to handle various CMOPs. Further details on LCMOEA are provided in the following section.

III. The Proposed Algorithm

A. The General Framework of LCMOEA

Algorithm 1 delineates the core framework of LCMOEA, mirroring the structure commonly found in traditional MOEAs as illustrated in Fig. 1. Tailored for a target CMOP with m objectives and n variables, it starts with an initial population P in line 1, composed of N randomly sampled solutions. The primary loop, encapsulating procedures from line 2 to line 7, embodies the iterative evolutionary process. Within this loop, it first gets two trained models M_1 and M_2 in line 3, which are used in the reproduction to aid the evolutionary search. Here, both M_1 and M_2 designed as multi-layer perceptron (MLPs), share a consistent architecture. Their input/output layers have n neurons and only one hidden layer containing K neurons is considered to make the cost of model training affordable and worth the benefits (K=10). As depicted in Fig.3, the training data of M_1 and M_2 involve pairing solutions from P according to the constraint-ignored task and feasibility-first task, respectively. Further details on model training are elaborated in the subsequent subsection. Empowered by M_1 and M_2, the LCMOEA generates an offspring population Q in line 4 through a learning-aided search on P, which can speed up the population's progress towards CPF. Employing a clustering-aided and constraint-involved environmental selection, half of the less adaptable solutions from P+Q are filtered out to get a new P in line 5. The above training, search, and selection procedures iterate until the termination condition is met, i.e., $FE = FE_{\max}$. Here, FE is the counter of function evaluations and $FE_{\max}$ is the preassigned maximum number of function evaluations. The final P, representing an approximation of the CPF for the target CMOP, is outputted in line 8.

Algorithm 2 Training

Input: the current population P with N solutions
Output: the trained M_1 and M_2
1:initialize two random MLP models M_1 and M_2
2:initialize N uniform reference vectors $v^1 \ldots v^N$
3:normalize all solutions in P by (9) and (13)
4:**for** e = 1 to *epoch* **do**
5: **for** i = 1 to N **do**
6: find $(x^1, x^2) \in P$ that are closest to v^i based on (8)
7: get $(x, \vec{x})$ from (x^1, x^2) for constraint-ignored task
8: compute the loss L_x of M_1 based on MSE
9: update the parameters of M_1 by $\partial L_x / \partial \mathbf{w}_1$
10: get $(x, \vec{x})$ from (x^1, x^2) for feasibility-first task
11: compute the loss L_x of M_2 based on MSE
12: update the parameters of M_2 by $\partial L_x / \partial \mathbf{w}_2$
13: **end for**
14:**end for**
15:**return** M_1 and M_2

B. Learning-aided Reproduction

The aim of reproduction is to generate new solutions with enhanced quality in the search space. In this paper, we propose a learnable differential evolution (LDE) operator that leverages two MLP models (i.e., M_1 and M_2) to search for new solutions, as depicted in Fig. 3. Concretely, for each parent solution $x \in P$, the LDE can generate a child solution x^c by the following search formula,

$$x^c = x + \alpha(y^1 - x) + (0.5 - \alpha)(y^2 - x) + r(x^1 - x^2), \quad (7)$$

where y^1 and y^2 are respectively the output of M_1 and M_2 with input of x. Besides, x^1 and x^2 are two random solutions selected from P, $x \neq x^1 \neq x^2$. Moreover, the parameter $\alpha \in [0, 0.5]$ is used to control the LDE search and r is a random value

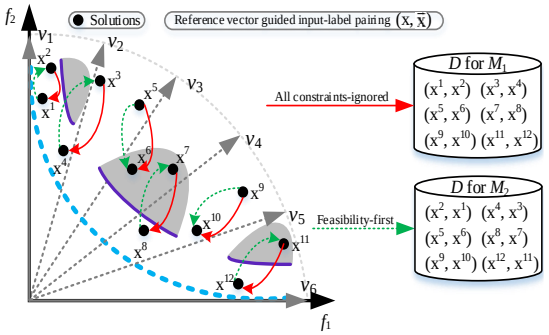


Fig. 4 Illustration of the reference vector-guided input-label pairing process to get the training data for M_1 and M_2, respectively.

sampled within [0, 1]. As can be seen from (7), LDE is distinct in that it uses two MLP models to learn two vectors y^1 and y^2 for guiding the evolutionary search. Specifically, LDE in (7) consists of three terms, each serving a distinct purpose: the first term $(y^1 - \mathrm{x})$ facilitates rapid convergence toward UPF, the second term $(y^2 - \mathrm{x})$ expedites convergence toward CPF, and the last term $(\mathrm{x}^1 - \mathrm{x}^2)$ enhances search diversity to prevent the population from being trapped in local optima. We will conduct in-depth experimental analysis on (7) in Section IV, including sensitivity analysis of parameter α and validity analysis of (y^1, y^2, x^1, x^2). Next, we will further explain the advantages of LDE by looking at why and how we learn the two vectors y^1 and y^2 for achieving specific purpose.

1) The expectations for MLP models

The search direction of traditional evolutionary operators (e.g., differential evolution) is often random. Such blind search will make their efficiency drop sharply when the search space becomes larger and constrained. The common coping strategy is to guide the search according to the historical optimal solutions. However, this search behavior may easily lead to local optima or premature convergence of the population. The main reason behind this is that once these guiding solutions fall into local optima, their potential for further improvement is significantly hampered. Our motivation stems from the belief that each solution, even those already identified as optimal, should possess the ability to learn a promising improvement direction at its specific position. To address this, we propose training two MLP models (M_1 and M_2) to achieve this purpose. The goal of M_1 is to learn the improvement principles of solutions for the constraint-ignored task. This model aims to push the population toward the UPF with an accelerated speed, guiding the search in the direction of objective-based performance improvement. Concurrently, M_2 is designed to learn the improvement principles for the feasibility-first task. The goal of this model is to rapidly guide the population toward the CPF by searching in the direction of feasibility improvement.

Thus, the use of MLP models allows for a more informed and directed search. Instead of relying solely on historical optimal solutions, these models provide each solution, including discovered best-performed solutions, the capability to learn a promising improvement direction at its specific position. Each solution becomes an active contributor to the improvement process. Solutions are empowered to adapt and guide the search based on their individual characteristics. This adaptability is crucial for avoiding stagnation, local optima, or premature convergence, common challenges in CMOEAs.

Algorithm 3 LearnableReproduction

Input: the trained M_1, M_2, and the parent population P
Output: the offspring population Q

```
initialize the offspring population Q = ∅
for each x ∈ P do
    compute the y^1 by inputting x to the M_1
    compute the y^2 by inputting x to the M_2
    select two different random solutions (x^{d_1}, x^{d_2}) ∈ P
    get offspring x^c by LDE on x, y^1, y^2, x^{d_1}, x^{d_2} with (7)
    update x^c by the polynomial mutation
    add x^c into the offspring population Q
end for
return Q
```

2) The training of MLP models

The training is a process to iteratively update the weights of M_1 and M_2 by backpropagation with gradient descent. It involves calculating and minimizing the loss by adjusting parameters along the steepest descent direction. For a detailed training procedures of MLP models, refer to the appendix. Given a data set $D = \{(\mathrm{x}_i, \vec{\mathrm{x}}_i)\}_{i=1}^{N}$ with N paired input-label examples, the goal of training is to update the weights so that the output y_i of the model is close enough to the label $\vec{\mathrm{x}}_i$ for each input x_i, $i = 1, \ldots, N$. In this work, the mean-square error (MSE) is applied as the loss L_{x} to be minimized for evaluation. We train two distinct MLP models, M_1 and M_2, utilizing datasets customized for specific downstream tasks. These models are equipped with different weights to enhance their adaptability. To effectively address challenges associated with constraints and the scaling up of the search space, we meticulously prepare training data for M_1 and M_2, tailoring them to two distinct optimization tasks.

Algorithm 2 presents the detailed training process, which starts with two random MLP models. To ensure the generalization ability, we first create N different scalarized subproblems using a set of N uniformly distributed reference vectors, i.e., $V = (v_1, \ldots, v_N)$. For each $v \in V$, two different solutions (x^1, x^2) $\in P$ that are closest to v can be found by computing $\theta(\mathrm{x}, v)$ for each $\mathrm{x} \in P$. Here, $\theta(\mathrm{x}, v)$ indicates the acute angle between x and v, which can be computed as:

$$\theta(\mathrm{x}, v) = \arccos\left|\frac{\mathrm{F}'(\mathrm{x}) \times v}{\|\mathrm{F}'(\mathrm{x})\| \times \|v\|}\right|, \tag{8}$$

where $\mathrm{F}'(\mathrm{x}) = (f_1'(\mathrm{x}), \ldots, f_m'(\mathrm{x}))$ is the normalized vector of x, and $f_i'(\mathbf{x})$ is computed as follows,

$$f_i'(\mathrm{x}) = \frac{f_i(\mathrm{x}) - z_i^{\min}}{z_i^{\max} - z_i^{\min}}, i = 1, 2, \ldots, m, \tag{9}$$

where $z_i^{\min}$ and $z_i^{\max}$ are the minimum and maximum values of the i-th objective for all solutions in P. Next, we just need to determine which of x^1 and x^2 is the input and which is the label. For M_1 on the constraint-ignored task, we get the input x and its label $\vec{\mathrm{x}}$ from (x^1, x^2) as follows,

$$\begin{cases} \mathrm{x} = \left[fit(\mathrm{x}^1, v) \geq fit(\mathrm{x}^2, v) \right] ?\ \mathrm{x}^1 : \mathrm{x}^2 \\ \vec{\mathrm{x}} = \left[fit(\mathrm{x}^1, v) < fit(\mathrm{x}^2, v) \right] ?\ \mathrm{x}^1 : \mathrm{x}^2 \end{cases}, \tag{10}$$

where $fit(\mathrm{x}, v)$ indicates the objective-based performance of x

Algorithm 4 ClusteringAidedSelection

Input: the parent population P and the child population Q
Output: the updated P
1: initialize $U = P + Q$ and reset $P = \varnothing$
2: initialize $c = (c^1, \ldots, c^{2N})$, where $c^i = \mathrm{x}^i \in U$, i=1,…,2N
3: initialize $cc = (cc^1, \ldots, cc^{2N})$ for $cc^i = \mathrm{F}'(\mathrm{x}^i)$, i=1,…,2N
4: **for** k = 1 to N **do**
5: get the cluster pair (c^u, c^h) by (14), $c^u \neq c^h \neq \varnothing$
6: update cluster $c^u = c^u + c^h$ and set $c^h = \varnothing$
7: update cluster centroid cc^u based on (15)
8: **end for**
9: add $\mathrm{x} \in c^i$ into P by (17) for i = 1,…,2N and $c^i \neq \varnothing$
10: **return** P

in solving the subproblem corresponding to reference vector v, which can be computed as follows,

$$fit(\mathrm{x}) = \sum_{i=1}^{m} v_i \, f_i(\mathrm{x}) \ , \tag{11}$$

For M_2 on the feasibility-first task, we first get the input x and its label $\vec{\mathrm{x}}$ as follows,

$$\begin{cases} \mathrm{x} = \left[cv(\mathrm{x}^1) > cv(\mathrm{x}^2) \right] ?\ \mathrm{x}^1 : \mathrm{x}^2 \\ \vec{\mathrm{x}} = \left[cv(\mathrm{x}^1) < cv(\mathrm{x}^2) \right] ?\ \mathrm{x}^1 : \mathrm{x}^2 \end{cases} , \tag{12}$$

If $cv(\mathrm{x}^1) = cv(\mathrm{x}^2)$, we can further confirm the input x and its label $\vec{\mathrm{x}}$ according to (10). In this way, we can compute the loss L_{x} of M_1 (the same of M_2) according to x and its label $\vec{\mathrm{x}}$, followed by updating its weights (denoted by w_1) via back-propagation with gradient descent (denoted by $\partial L_{\mathrm{x}} / \partial \mathrm{w}_1$). It is worth pointing out that only the normalized variable vector of each solution is involved in the training process. Concretely, the ith variable x_i of x is normalized as follows:

$$x_i' = \frac{x_i - lb_i}{ub_i - lb_i}, i = 1, 2, ..., n \ , \tag{13}$$

where ub = (ub_1, …, ub_n) and lb = (lb_1, …, lb_n) are the corresponding upper and lower bounds of the n-dimension variable space. We use the N input-label pairs obtained above as the dataset to train M_1 and M_2 for 10 epochs. Finally, the trained two models are outputted to aid the evolutionary search in (7). To simplify comprehension, the process of preparing training data for both tasks is illustrated in Fig. 4.

The CMOP is divided into subproblems using uniformly distributed reference vectors. Training data is then created for each subproblem for M_1 and M_2, considering objective-based performance and CV degree, respectively, to maximize model generalization. However, conflicts may arise in M_1 and M_2 training data, particularly when handling fully separated UPF and CPF, as illustrated in Fig. 4's solution pairing. Hence, the parameter α in (7) plays a crucial role in balancing conflicts introduced by these models during LDE's evolutionary search. The general strategy is to guide the population swiftly toward the UPF initially and redirect it toward the CPF at a specific evolutionary stage, with parameter α adjusted accordingly to align with this insight.

3) The overall process of learnable reproduction

Algorithm 3 presents the overall process of the proposed learnable reproduction to get an offspring population Q by the LDE, which requires three inputs: the trained M_1, M_2, and P.

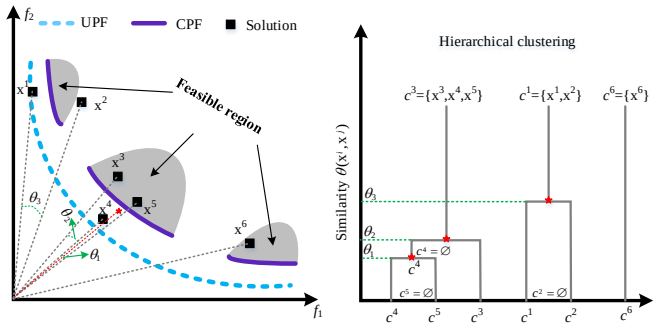


Fig. 5 Illustration of the bottom-up hierarchical clustering process.

At first, the Q is initialized as an empty-set in line 1. Then, it is updated in the main loop, including the procedures from line 2 to line 9, by saving the N new child solutions. Concretely, for each $\mathrm{x} \in P$ in line 2, the two guidance vectors y^1 and y^2 can be computed by inputting x into the trained M_1 and M_2, respectively. Then, other two different solutions x^{d_1} and x^{d_2} are randomly selected from P. After that, the LDE is run on x, y^1, y^2, x^{d_1} and x^{d_2} to reproduce a new child solution x^c, followed by an addition operator to update Q. Please note that the polynomial mutation is also used in the reproduction to update x^c for further avoiding the trap of local optimality. At last, the obtained Q with N child solutions is outputted in line 10.

The computational complexity of this learnable reproduction is mainly determined by the training and search process. Specifically, the training of MLP models has a time complexity of O($NEnK$), where N represents the population size, E is the number of epochs for training, n is the number of decision variables, and K is the hidden layer size. The time complexity for generating offspring solutions with LDE is O(NnK). The time complexity of this reproduction strategy should be consistent with common evolutionary operations, necessitating the set of small values for E and K.

C. Clustering-aided Environmental Selection

The CPFs of CMOPs are often irregular (e.g., disconnected) due to the intervention of constraints. Such irregularity often brings great challenges to existing CMOEAs in maintaining considerable population's diversity, especially in the case of many-objective optimization. The clustering-aided environmental selections [63]-[65] have been proved to be effective in solving UMOPs with irregular UPFs. To continue this research direction, an improved clustering-aided selection strategy that takes solution's feasibility into account is developed in LCMOEA for solving CMOPs.

Algorithm 4 shows the pseudo-code of the developed environmental selection, which includes three main components: initialization in lines 1-3, bottom-up hierarchical clustering in lines 4-8, and representative solution selection in line 9. Concretely, the inputted parent population P and child population Q together form a combined population U with 2N solutions. Each of these 2N solutions (denoted by $\mathrm{x}^i \in U$) is initialized to a cluster c^i, and the location of x^i in the normalized objective space, i.e., $\mathrm{F}'(\mathrm{x}^i)$, is initialized to the corresponding cluster center cc^i. The clustering process is then iterated for N rounds (indicated by k = 1 to N). In each round, the cluster pair (c^u, c^h) with the highest similarity are found by

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2024.3380366

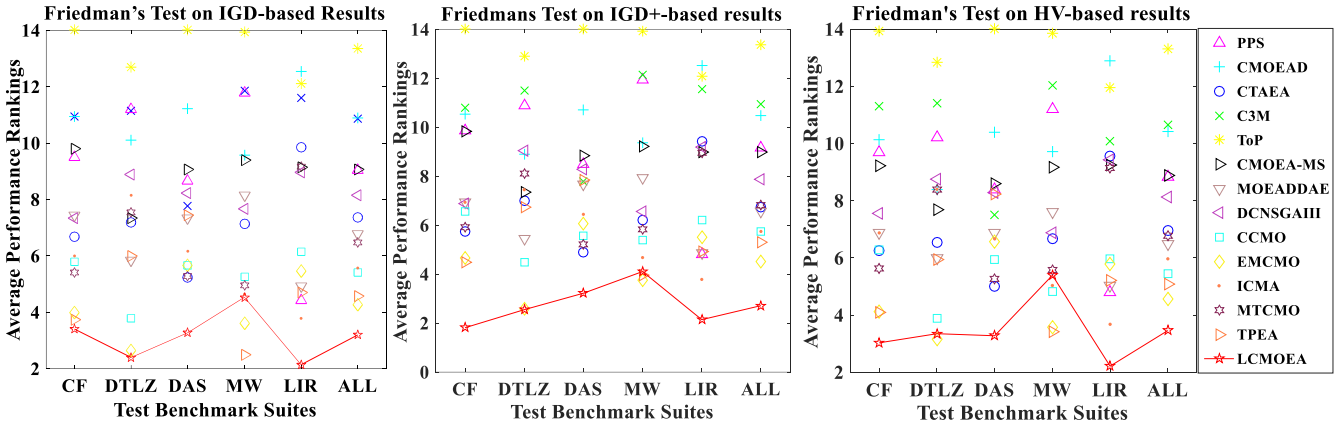


Fig. 6 Illustration of the average performance rankings of LCMOEA and its thirteen competitors in solving the five CMOP benchmark suites.

$$(c^u, c^h) = \underset{c^i \neq \varnothing, c^j \neq \varnothing}{(c^i, c^j)} : \arg\min_{i,j \in \{1,\ldots,2N\}, i \neq j} \theta(cc^i, cc^j) \quad (14)$$

where $\theta(cc^i, cc^j)$ indicates the similarity between the i-th and the j-th cluster (i.e., c^i and c^j), evaluated by the cosine distance between their corresponding cluster centers (i.e., cc^i and cc^j). $\theta(cc^i, cc^j)$ can be calculated similar to (8). Then the two most similar clusters are merged into a cluster and saved in the c^u (i.e., $c^u = c^u + c^h$), while c^h is emptied directly (i.e., $c^h = \varnothing$). After that, cc^u is updated by the following formula,

$$cc^u = \sum_{x \in c^u} F'(x) \Big/ |c^u| \quad (15)$$

After N iterations, we obtain N non-empty clusters, exemplified in Fig. 5 with N=3. The subsequent step involves selecting a representative from each non-empty cluster, which presents three possible cases: 1) only one solution in the cluster (e.g., the final c^6); 2) the cluster contains multiple solutions, all either feasible or infeasible (e.g., the c^1); 3) a mix of both feasible and infeasible solutions in the cluster (e.g., the c^3 in Fig. 5). No choice is required for the first case. In the second case, if all solutions are feasible, the one with the best objective-based performance is selected directly. However, if all solutions are infeasible, a dilemma arises: whether to choose based on objective performance or CV degree. This dilemma is also encountered in the third case. Opting for selection solely based on objective performance leads the population to the UPF, while relying solely on CV can make the population susceptible to deceptive regions. To resolve this dilemma, we define a comprehensive indicator $CI(x)$ to reflect the fitness of solution x, computed as follows:

$$CI(x) = w \times rank_{obj}(x) + (1-w) \times rank_{cv}(x) \quad (16)$$

where $rank_{obj}(x)$ and $rank_{cv}(x)$ respectively indicate the objective-based performance rank and the CV-based feasibility rank. $rank_{obj}(x)$ is obtained by ascending sorting all the solutions in the same cluster based on the sum of their normalized objectives; $rank_{cv}(x)$ is obtained similarly, taking into account the CV degree of solutions. Since only solutions belonging to the same cluster are ranked, this is a localized CHT that balances convergence and feasibility by setting the value of w appropriately. To be consistent with evolutionary search, the value of w is adaptively set as w = 1.0 if $FE < 0.4FE_{max}$; w = 0.1 if $FE > 0.6FE_{max}$; w=-4.5FE/FE_{max}+2.8 otherwise (Fig.A1 in appendix shows how w changes). This configuration is rational, focusing on the objective-based ranking in the initial stages of evolution and shifting to CV-based ranking in the later stages. Therefore, in each cluster $c^i \neq \varnothing$, its representative solution x^t with the best fitness can be found by

$$x^t = x : \arg\min_{x \in c^i, c^i \neq \varnothing} CI(x) \quad (17)$$

In this process, the updated P is formed by selecting representative solutions from each of the N non-empty clusters. The environmental selection, achieved through clustering, ensures population diversity. Additionally, the choice of representative solutions, guided by equation (17), contributes to a balanced consideration of both convergence and feasibility.

The computational complexity of this cluster-based selection is mainly determined by the clustering process. Specifically, computing the $\theta(cc^i, cc^j)$ for all pairs exhibits a time complexity of $O(mN^2)$. Then we can find the current most similar partner of each cluster and record it in memory, which contributes a time complexity of $O(mN^2)$. In the loop, the main contributor to time complexity is the procedure in line 5, which requires $O(N)$ time complexity to find the smallest of N pairs (already recorded in memory). The subsequent update is localized to those clusters linked to the clusters c^h and c^u, avoiding a full update of all clusters, including the current most similar partner for each. Thus, the overall worst-case time complexity of this cluster-based selection is $O(mN^2)$, comparable to the time complexities of most MOEAs [63].

IV. Experimental Studies

In this section, extensive experiments are conducted on the PlatEMO [66] to confirm the superior performance of the proposed LCMOEA. First, it is compared with thirteen competitive CMOEAs in solving five test benchmark suites. Then, its scalability is further verified by solving CMOPs with high-dimensional objective or search spaces. In addition, its superior performance is validated on a real-world test suite. Finally, an ablation study along with parameters analysis and computational complexity analysis of LCMOEA is conducted to investigate the effectiveness of its main components.

A. Preliminary description of the experiments

As listed in Table A1 of the appendix, five different benchmark suites are adopted for the experimental studies:

TABLE I
STATISTICAL COMPARISON RESULTS OF THIRTEEN CMOEA COMPETITORS (VS. LCMOEA) ON THE FIVE CMOP BENCHMARK SUITES

Vs. LCMOEA on		PPS	CMOEAD	CTAEA	C3M	ToP	CMOEAMS	MOEADAE	DCNSGAIII	CCMO	EMCMO	ICMA	MTCMO	TPEA
		+/−/=	+/−/=	+/−/=	+/−/=	+/−/=	+/−/=	+/−/=	+/−/=	+/−/=	+/−/=	+/−/=	+/−/=	+/−/=
C-DTLZ	IGD	0/10/0	0/9/1	1/9/0	0/10/0	0/10/0	1/7/2	1/7/2	0/8/2	3/5/2	4/5/1	0/9/1	2/7/1	2/7/1
	IGD+	0/10/0	1/7/2	1/9/0	0/10/0	0/10/0	1/8/1	2/7/1	0/9/1	1/5/4	2/5/3	2/8/0	0/8/2	0/8/2
	HV	0/10/0	1/7/2	1/7/2	0/10/0	0/10/0	2/7/1	2/6/2	1/8/1	**4/4/2**	3/5/2	2/7/1	2/6/2	2/6/2
DASCMOP	IGD	0/9/0	0/9/0	3/6/0	1/7/1	0/9/0	0/7/2	1/6/2	2/6/1	3/5/1	2/5/2	1/4/4	2/5/2	0/9/0
	IGD+	0/9/0	1/8/0	2/6/1	0/8/1	0/9/0	0/7/2	2/6/1	2/6/1	2/6/1	2/6/1	2/4/3	2/6/1	0/9/0
	HV	0/8/1	1/7/1	2/6/1	0/7/2	0/9/0	0/6/3	1/6/2	2/6/1	2/5/2	2/4/3	1/5/3	2/5/2	0/9/0
ZXH-CF	IGD	1/15/0	1/15/0	2/13/1	0/16/0	0/16/0	2/13/1	4/11/1	0/16/0	5/11/0	5/11/0	6/8/2	5/11/0	5/8/3
	IGD+	0/16/0	0/16/0	1/15/0	0/16/0	0/16/0	1/14/1	2/14/0	1/14/1	1/15/0	2/12/2	3/12/1	1/13/2	2/13/1
	HV	1/15/0	0/15/1	2/13/1	0/15/1	0/16/0	2/13/1	2/11/3	1/14/1	2/10/4	3/10/3	3/11/2	4/11/1	3/10/3
MW	IGD	0/14/0	2/12/0	4/9/1	0/14/0	0/14/0	0/12/2	2/10/2	3/8/3	4/6/4	**7/5/2**	4/6/4	**5/5/4**	**7/2/5**
	IGD+	0/14/0	2/12/0	4/9/1	0/14/0	0/14/0	0/13/1	0/13/1	4/8/2	3/7/4	**5/7/2**	4/8/2	2/6/6	**8/4/2**
	HV	0/12/2	1/12/1	5/8/1	0/13/1	0/14/0	1/8/5	1/9/4	4/7/3	2/3/9	**6/2/6**	**4/7/3**	**1/4/9**	**6/2/6**
LIRCMOP	IGD	3/10/1	0/14/0	0/12/2	0/14/0	0/14/0	2/12/0	2/11/1	0/14/0	2/12/0	2/12/0	2/9/3	0/14/0	2/12/0
	IGD+	3/10/1	0/14/0	2/12/0	0/14/0	0/14/0	2/12/0	2/10/2	0/13/1	2/12/0	2/12/0	2/10/2	0/14/0	1/12/1
	HV	2/10/2	0/14/0	1/12/1	0/14/0	0/14/0	2/12/0	2/11/1	0/14/0	1/12/1	2/12/0	1/9/4	0/14/0	1/12/1
p-value	IGD	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.000002	0.0000	0.00297	0.015363	0.001454	0.000011	0.005461
	IGD+	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.000045	0.015645	0.000048	0.0000	0.000498
	HV	0.0000	0.0000	0.000003	0.0000	0.0000	0.0000	0.000048	0.0000	0.000729	0.013885	0.000766	0.000008	0.002984
Hommel	IGD	0.005556	0.004167	0.007143	0.004545	0.003846	0.005	0.008333	0.00625	0.016667	0.05	0.0125	0.01	0.025
	IGD+	0.005	0.004545	0.008333	0.004167	0.003846	0.005556	0.01	0.00625	0.0125	0.05	0.016667	0.007143	0.025
	HV	0.005556	0.004545	0.007143	0.004167	0.003846	0.005	0.01	0.00625	0.016667	0.05	0.0125	0.008333	0.025

Note: *p*-values obtained in by applying post hoc methods over the results of Friedman procedure for $\alpha = 0.05$. A smaller *p*-value indicates a significant difference.

CDTLZ [10], [31], MW [4], DASCMOP [67], ZXH-CF [68], and LIRCMOP [69]. There are a total of sixty-three test problems, which basically cover all the different types of CMOPs. Three well-known performance indicators: inverted generational distance (IGD) [70], IGD+ [71], and hypervolume (HV) [72] are used to evaluate a COMEA comprehensively from the diversity and convergence of its obtained final nondominated solutions. Higher HV values denote superior performance, while lower values for IGD or IGD+ are desirable. The calculation of IGD or IGD+ requires a sufficient number of uniformly sampled points on the true CPF. Calculating HV requires setting such a reference point that can be dominated by all points on the true CPF. In this study, all settings for these three indicators follow the default settings on the PlatEMO. Specifically, 10^4 sample points from the true CPF are used for the calculations of IGD and IGD+, while the reference point for HV calculation is established as (1, 1, . . ., 1), with objective vectors in the final solution sets normalized by 1.1×($f_1^{\max}$, . . . , $f_m^{\max}$), where $f_k^{\max}$ (k = 1, . . . , m) is the maximum value of the k-th objective in the true CPF. Each algorithm undergoes 30 runs, and the average values of the indicators across these runs are used for subsequent analysis. In each run, the $FE_{\max}$ is set as the termination condition. Moreover, the population size (N) is set as N = 105 for CMOPs with 2 and 3 objectives, N = 210, N = 275, and N = 240 for 5-, 10-, and 15-objective CMOPs, respectively.

Thirteen competitors covering the four different types of CMOEAs are included for performance comparison, namely PPS [47], CMOEA/D [10], CTAEA [31], C3M [57], ToP [11], CMOEA-MS [6], MOEADAE [58], DCNSGAIII [30], CCMO [33], EMCMO [37], ICMA [20], MTCMO [42], and TPEA [36]. To ensure a fair comparison, the recommended parameter settings in their respective references are adopted, aiming to attain their optimal performance. These fair parameters align with the default settings on the PlatEMO, most of which are directly provided by the authors of the respective algorithms. For the proposed LCMOEA, the value of α in (7) is setting as follows:

$$\begin{cases} \alpha = 0.5 \ \ if \ \ FE \le 0.5FE_{\max} \\ \alpha = 0.0 \ \ if \ \ FE > 0.5FE_{\max} \end{cases}, \tag{18}$$

which is set based on our experimental results.

B. Results on the Five Benchmark Suites

In this study, LCMOEA and its thirteen competitors are applied to solve five distinct benchmark suites with default settings. The termination condition, set uniformly at $FE_{\max}$ = 10^5 for all cases, aimed to ensure a consistent comparison. Comprehensive IGD, IGD+, and HV results obtained by these CMOEAs are presented in Tables A2 to A16 in the appendix. The statistical analysis relied on the Wilcoxon rank test with a significance level of 0.05. Utilizing symbols such as "+", "−", and "=" facilitated the indication of whether the performance of the compared CMOEA was significantly superior to, inferior to, or similar to LCMOEA. Detailed statistical comparison results are elucidated in Table I. Notably, LCMOEA exhibited superior performance across all test suites, with the exception of MW.

To present a clearer overview of the comparison results, the IGD, IGD+, and HV results of LCMOEA and its competitors are sorted using the Friedman's test method, as depicted in Fig. 5. The average performance ranking values for each algorithm on these three indicators are detailed in Tables A17 to A19 in the appendix. A higher average ranking signifies better performance. Fig. 6 underscores that LCMOEA excelled in performance rankings across all test suites, with the exception of MW. On MW problems, LCMOEA can only get the average performance ranking of 4.5357 on the IGD+ and IGD-based

TABLE II
HV RESULTS OF SIX SELECTED CMOEA COMPETITORS (VS. LCMOEA) ON THE SCALABLE CMOPS WITH HIGH-DIMENSIONAL OBJECTIVE SPACE

Problem	*m*	*n*	CCMO	DCNSGAIII	EMCMO	ICMA	TPEA	CRVEA	LCMOEA
C1_DTLZ1	5	9	9.7770e-1(6.44e-3)-	9.7886e-1(1.65e-3)-	9.7843e-1(3.58e-4)-	9.7595e-1(9.52e-4)-	9.7828e-1(1.44e-3)=	9.7684e-1(2.33e-3)-	**9.7963e-1(3.72e-4)**
	10	14	9.7854e-1(2.38e-2)-	9.9736e-1(1.67e-3)-	9.8610e-1(1.23e-2)-	9.9425e-1(1.56e-3)-	9.9646e-1(1.48e-3)-	9.9309e-1(2.42e-3)-	**9.9959e-1(3.61e-5)**
	15	19	9.9302e-1(1.08e-2)-	9.9906e-1(9.33e-4)-	9.9297e-1(7.88e-3)-	9.9590e-1(1.55e-3)-	9.9616e-1(3.50e-3)-	9.8289e-1(1.36e-2)-	**9.9996e-1(1.02e-5)**
C1_DTLZ3	5	14	7.4913e-1(3.49e-2)-	1.6215e-1(3.19e-1)-	7.6459e-1(2.73e-2)-	4.3748e-2(1.86e-1)-	5.5488e-1(2.46e-1)-	7.8927e-1(3.03e-3)-	**8.0813e-1(1.36e-3)**
	10	19	0.0000e+0(0.00e+0)-	1.1369e-2(4.82e-2)-	0.0000e+0(0.00e+0)-	0.0000e+0(0.00e+0)-	0.0000e+0(0.00e+0)-	1.7521e-5(7.43e-5)-	**3.7217e-1(1.17e-1)**
	15	24	0.0000e+0(0.00e+0)-	2.0954e-1(3.72e-1)-	0.0000e+0(0.00e+0)-	0.0000e+0(0.00e+0)-	0.0000e+0(0.00e+0)-	0.0000e+0(0.00e+0)-	**6.5720e-1(4.78e-1)**
C2_DTLZ2	5	14	7.3487e-1(3.35e-3)=	7.5390e-1(1.60e-3)+	7.5122e-1(3.17e-3)+	7.2463e-1(3.73e-3)-	7.3343e-1(4.63e-3)=	**7.5535e-1(1.50e-3)+**	7.3316e-1(3.97e-3)
	10	19	8.4042e-1(9.95e-3)-	8.9139e-1(1.71e-2)+	8.4625e-1(7.49e-3)=	8.6020e-1(3.57e-3)+	8.4475e-1(7.69e-3)=	**8.9476e-1(2.88e-3)+**	8.4911e-1(9.81e-3)
	15	24	9.5102e-1(2.02e-3)+	9.4213e-1(3.50e-2)+	9.5202e-1(1.38e-3)+	9.2141e-1(5.56e-3)-	9.3601e-1(3.73e-3)=	**9.6452e-1(1.62e-3)+**	9.3781e-1(4.75e-3)
C3_DTLZ4	5	14	9.4671e-1(1.26e-3)-	9.5877e-1(1.54e-2)-	9.4763e-1(1.81e-3)-	9.6158e-1(5.71e-4)=	9.4236e-1(2.49e-3)-	8.9008e-1(1.79e-2)-	**9.6184e-1(5.05e-4)**
	10	19	8.5601e-2(4.25e-3)-	9.9573e-1(2.51e-3)-	9.6525e-2(1.53e-2)-	9.9935e-1(2.65e-5)-	7.6415e-1(3.78e-2)-	9.9886e-1(1.14e-4)-	**9.9953e-1(2.35e-5)**
	15	24	1.0752e-1(6.59e-3)-	9.9967e-1(2.80e-4)-	1.0498e-1(4.99e-3)-	9.9997e-1(5.89e-6)-	4.4796e-1(5.73e-2)-	9.9996e-1(8.07e-6)-	**9.9999e-1(3.01e-6)**
DC1_DTLZ1	5	9	7.6986e-1(6.14e-4)+	**7.6989e-1(8.90e-4)+**	7.6880e-1(1.61e-3)=	7.6271e-1(1.47e-3)-	7.6306e-1(2.44e-3)-	7.6154e-1(3.46e-3)-	7.6910e-1(9.02e-4)
	10	14	3.6725e-3(1.56e-2)-	7.9631e-1(4.15e-4)-	0.0000e+0(0.00e+0)-	3.3517e-1(3.01e-1)-	0.0000e+0(0.00e+0)-	7.8918e-1(1.79e-3)-	**7.9656e-1(2.68e-4)**
	15	19	0.0000e+0(0.00e+0)-	7.9657e-1(3.55e-4)-	0.0000e+0(0.00e+0)-	4.6822e-1(2.35e-1)-	0.0000e+0(0.00e+0)-	7.9053e-1(6.12e-3)-	**7.9748e-1(4.97e-4)**
DC1_DTLZ3	5	14	7.3124e-1(9.84e-3)-	**7.6318e-1(1.79e-3)+**	7.3472e-1(3.63e-2)-	3.3422e-1(3.10e-1)-	6.1616e-1(1.07e-1)-	7.6284e-1(5.35e-3)+	7.5247e-1(2.94e-3)
	10	19	0.0000e+0(0.00e+0)-	5.8539e-1(2.43e-1)-	0.0000e+0(0.00e+0)-	0.0000e+0(0.00e+0)-	0.0000e+0(0.00e+0)-	9.3473e-1(1.27e-2)-	**9.4838e-1(2.63e-3)**
	15	24	0.0000e+0(0.00e+0)-	8.8097e-1(1.00e-1)-	0.0000e+0(0.00e+0)-	0.0000e+0(0.00e+0)-	0.0000e+0(0.00e+0)-	9.8357e-1(2.55e-3)-	**9.8867e-1(1.97e-3)**
DC2_DTLZ1	5	9	9.7524e-1(1.27e-3)-	9.2912e-1(7.87e-2)=	**9.7981e-1(2.21e-4)+**	9.7515e-1(6.19e-3)-	9.7257e-1(8.89e-4)-	9.7888e-1(2.79e-4)+	9.7855e-1(4.22e-4)
	10	14	9.6488e-1(0.00e+0)=	9.7502e-1(2.82e-2)=	9.5990e-1(1.10e-2)-	9.0255e-1(2.86e-1)-	9.3603e-1(1.52e-1)-	9.9825e-1(5.45e-3)-	**9.9960e-1(3.63e-5)**
	15	19	NaN(NaN) -	9.2979e-1(1.78e-1)-	9.6262e-1(5.38e-3)-	9.9849e-1(1.67e-3)-	9.7096e-1(6.41e-3)-	9.9861e-1(4.08e-3)-	**9.9996e-1(1.56e-5)**
DC2_DTLZ3	5	14	2.4982e-1(3.28e-1)-	5.5686e-2(1.66e-2)-	2.4610e-1(2.80e-1)-	6.5336e-2(3.63e-2)-	4.9594e-2(9.40e-3)-	3.1431e-1(3.57e-1)=	**7.8852e-1(3.72e-3)**
	10	19	9.9946e-2(1.38e-2)=	1.5903e-1(4.39e-2)=	1.3034e-1(1.19e-3)=	1.1065e-1(3.43e-2)=	NaN(NaN) -	1.6967e-1(3.58e-2)=	**6.6693e-1(4.23e-1)**
	15	24	NaN(NaN) -	2.9269e-2(1.10e-2)-	NaN(NaN) -	1.3469e-1(2.20e-2)=	NaN(NaN) -	1.3306e-1(2.14e-1)-	**4.5130e-1(4.43e-1)**
DC3_DTLZ3	5	14	1.7186e-1(2.49e-1)-	9.5495e-2(2.20e-1)-	2.7306e-1(2.50e-1)-	0.0000e+0(0.00e+0)-	3.1028e-2(1.32e-1)-	3.2483e-1(2.67e-1)-	**5.4445e-1(1.36e-1)**
	10	19	0.0000e+0(0.00e+0)-	0.0000e+0(0.00e+0)-	0.0000e+0(0.00e+0)-	0.0000e+0(0.00e+0)-	0.0000e+0(0.00e+0)-	0.0000e+0(0.00e+0)-	**4.6787e-1(3.84e-1)**
	15	24	0.0000e+0(0.00e+0)-	3.2640e-3(1.38e-2)-	0.0000e+0(0.00e+0)-	0.0000e+0(0.00e+0)-	0.0000e+0(0.00e+0)-	0.0000e+0(0.00e+0)-	**1.2022e-1(6.83e-4)**
MW4	5	15	9.7332e-1(8.26e-4)-	9.7980e-1(1.49e-4)+	9.7451e-1(5.33e-4)-	9.7444e-1(4.62e-4)-	9.7000e-1(8.79e-4)-	9.7864e-1(3.43e-4)=	**9.7991e-1(1.63e-4)**
	10	15	9.9309e-1(3.12e-3)-	9.9959e-1(1.70e-4)=	9.9343e-1(2.20e-3)-	9.9943e-1(3.19e-5)-	9.9483e-1(7.38e-4)-	9.9949e-1(6.97e-5)-	**9.9962e-1(3.31e-5)**
	15	15	9.9995e-1(4.19e-6)=	9.9991e-1(1.23e-5)-	9.9997e-1(5.17e-6)=	9.9997e-1(6.13e-6)=	9.9997e-1(6.08e-6)=	9.6777e-1(2.00e-2)-	**9.9998e-1(1.23e-5)**
MW8	5	15	7.6128e-1(6.51e-3)-	**8.1118e-1(2.12e-3)+**	7.8306e-1(7.34e-3)-	7.6601e-1(8.64e-3)-	7.5974e-1(8.46e-3)-	8.0724e-1(4.59e-3)+	7.9124e-1(4.90e-3)
	10	15	7.8996e-1(1.34e-2)-	9.6324e-1(1.67e-2)+	7.9674e-1(1.21e-2)-	9.4327e-1(2.30e-3)-	8.1189e-1(8.20e-3)-	**9.6900e-1(2.98e-4)+**	9.6227e-1(1.20e-3)
	15	15	7.9212e-1(9.90e-3)-	9.8346e-1(6.76e-3)=	7.9093e-1(5.12e-3)-	9.8042e-1(1.42e-3)-	8.2138e-1(7.52e-3)-	9.8388e-1(2.86e-3)-	**9.9112e-1(2.61e-4)**
MW14	5	15	3.8900e-1(2.53e-3)-	3.5932e-1(1.63e-2)-	3.9054e-1(2.03e-3)-	3.7744e-1(4.21e-3)-	3.6933e-1(2.79e-3)-	3.9821e-1(8.33e-3)-	**4.0935e-1(1.36e-3)**
	10	15	1.3950e-1(1.65e-1)-	2.0336e-1(3.65e-3)-	8.7471e-2(2.07e-2)-	1.4309e-1(1.33e-2)-	1.0596e-1(1.91e-2)-	2.5203e-1(5.19e-3)-	**2.6807e-1(4.39e-3)**
	15	15	5.7315e-3(2.98e-3)-	1.6269e-1(2.98e-3)-	7.8381e-3(5.55e-3)-	9.2013e-2(3.24e-3)-	4.7978e-3(4.43e-3)-	1.2484e-1(4.70e-2)-	**2.1140e-1(2.19e-3)**
+/−/=(5-objective)			2/10/1	5/7/1	3/9/1	0/12/1	0/11/2	4/7/2	----------
+/−/=(10-objective)			0/10/2	2/7/3	0/10/2	1/10/1	0/11/1	2/9/1	----------
+/−/=(15-objective)			0/11/1	4/6/2	0/11/1	1/9/2	011/1	3/8/1	----------

results, which is worse than EMCMO (3.6071 on IGD and 3.75 on IGD+) and TPEA (2.5 on IGD and 3.9643 on IGD+). In addition, LCMOEA can only rank fifth (5.3929) in the HV results of solving the MW problems, and the competitors that rank higher are ICMA (5.0357), CCMO (4.8214), EMCMO (3.5714), and TPEA (3.4286). These CO-based CMOEAs proved advantageous in scenarios where the majority of the CPF lies on the UPF, as observed in MW1-2, MW4-6, MW8, MW10, and MW14. Additionally, MW problems, unlike the other test suites (CDTLZ, DASCMOP, ZXH-CF, and LIRCMOP), feature constraints solely based on objective functions, unrelated to variables. The MLP model trained for constraint-first tasks in LCMOEA may not be as effective for this CMOP type. Nonetheless, LCMOEA demonstrated the best overall performance across all three indicators.

C. Scalability to the High-Dimensional Objective Space

This section focuses on solving thirteen scalable CMOPs, encompassing all CDTLZ problems and three MW problems, each with 5-, 10-, and 15-objectives. LCMOEA is evaluated against ten selected CMOEA competitors, including two designed explicitly for many-objective CMOPs (m>3), CRVEA [12] and MaOEAIT [73]. The termination conditions are set as $FE_{\max}$ = 210,000, 275,000, and 480,000 for 5-, 10-, and 15-objective cases. Detailed IGD, IGD+, and HV results are provided in Tables A20 to A22 of the appendix. Partial HV results are outlined in Table II. LCMOEA consistently demonstrates superior performance across most many objective CMOPs, with the exception of C2_DTLZ2. It's noteworthy that while LCMOEA may perform less favorably than CCMO, EMCMO, TPEA, and ICMA in solving MW problems with 2 and 3 objectives, it significantly outperforms them in tackling the scalable MW problems (MW4, MW8, and MW14) featuring 5 to 15 objectives. This underscores the superior scalability of LCMOEA in addressing many objective CMOPs compared to its competitors.

In addition, the DCNSGAIII, which performs poorly for 2- and 3-objective CMOPs, achieves remarkable performance in solving many-objective CMOPs (m>=5). CRVEA has similar performance to DCNSGAIII in that they both use a reference vector-guided environmental selection and a CDP-based CHT. However, their uniformly distributed reference vectors often mismatch irregular CPFs, rendering them unsuitable for solving irregular CMOPs. Consequently, their performance is inherently not superior to LCMOEA in addressing irregular and many-objective CMOPs. LCMOEA's stronger adaptability to the scaling-up of the objective space can be attributed to the proposed clustering-aided environmental selection strategy, which has contributed the most in getting a promising population with balanceable diversity, convergence, and feasibility.

TABLE III
IGD+ RESULTS OF FIVE SELECTED CMOEA COMPETITORS (VS. LCMOEA) ON THE SCALABLE CMOPS WITH HIGH-DIMENSIONAL SEARCH SPACE

Problem	m	n	CCMO	EMCMO	ICMA	LMOCSO	POCEA	LCMOEA
LIRCMOP1	2	100	2.7680e-1 (9.27e-3) -	2.8023e-1 (6.43e-3) -	2.6636e-1 (1.05e-2) -	NaN (NaN) -	2.3102e-1 (2.27e-2) -	**2.1499e-1 (2.17e-2)**
	2	500	2.8603e-1 (5.75e-3) -	2.8512e-1 (5.88e-3) -	2.8391e-1 (1.49e-3) -	NaN (NaN) -	2.8074e-1 (8.34e-3) -	**2.6643e-1 (8.57e-3)**
	2	1000	2.8762e-1 (3.77e-3) -	2.8691e-1 (4.98e-3) -	2.8663e-1 (9.46e-4) -	NaN (NaN) -	2.8651e-1 (6.42e-3) -	**2.7313e-1 (7.19e-3)**
LIRCMOP2	2	100	1.9056e-1 (9.87e-3) -	1.9104e-1 (8.38e-3) -	1.6828e-1 (8.83e-3) -	NaN (NaN) -	1.4812e-1 (1.18e-2) -	**1.3235e-1 (4.59e-3)**
	2	500	1.9413e-1 (8.11e-3) -	1.9451e-1 (7.79e-3) -	1.9535e-1 (3.12e-3) -	NaN (NaN) -	1.9190e-1 (1.01e-2) -	**1.7002e-1 (4.06e-3)**
	2	1000	1.9492e-1 (6.69e-3) -	1.9545e-1 (5.91e-3) -	1.9922e-1 (2.23e-3) -	NaN (NaN) -	1.9855e-1 (4.95e-3) -	**1.8626e-1 (7.76e-3)**
LIRCMOP3	2	100	2.7838e-1 (4.05e-3) -	2.7879e-1 (4.44e-3) -	2.5684e-1 (2.13e-2) -	NaN (NaN) -	2.5707e-1 (2.27e-2) -	**2.0275e-1 (1.68e-2)**
	2	500	2.8355e-1 (2.36e-3) -	2.8321e-1 (2.02e-3) -	2.8184e-1 (2.72e-3) -	NaN (NaN) -	2.8268e-1 (5.57e-3) -	**2.7272e-1 (2.43e-3)**
	2	1000	2.8345e-1 (3.44e-3) -	2.8297e-1 (4.76e-3) -	2.8353e-1 (1.65e-3) -	NaN (NaN) -	2.8563e-1 (3.17e-3) -	**2.7604e-1 (5.61e-3)**
LIRCMOP4	2	100	2.0434e-1 (1.43e-2) -	2.0058e-1 (1.24e-2) -	1.8653e-1 (1.86e-2) -	NaN (NaN) -	1.8141e-1 (2.17e-2) -	**1.4933e-1 (1.53e-2)**
	2	500	2.0698e-1 (3.31e-3) -	2.0730e-1 (1.86e-3) -	2.0778e-1 (1.62e-3) -	NaN (NaN) -	2.0682e-1 (7.86e-3) -	**2.0123e-1 (2.43e-3)**
	2	1000	2.0824e-1 (1.04e-3) -	2.0771e-1 (2.38e-3) -	2.0800e-1 (1.08e-3) -	NaN (NaN) -	2.1074e-1 (2.78e-3) -	**2.0375e-1 (3.07e-3)**
LIRCMOP5	2	100	**2.7651e-1 (2.58e-2) =**	2.8238e-1 (2.50e-2) =	1.4609e+0 (5.11e-1) -	2.8850e-1 (2.55e-2) =	1.9119e+0 (6.92e-1) -	2.8995e-1 (2.12e-2)
	2	500	**2.8530e-1 (1.24e-2) +**	2.9573e-1 (1.59e-2) +	2.5712e+0 (1.09e-4) -	3.0037e-1 (1.51e-2) +	2.5786e+0 (2.43e-3) -	1.2368e+0 (3.09e-3)
	2	1000	**2.8411e-1 (1.41e-2) +**	2.8435e-1 (1.37e-2) +	2.5715e+0 (2.88e-4) -	3.1662e-1 (1.03e-2) +	2.5767e+0 (1.66e-3) -	1.2505e+0 (4.19e-3)
LIRCMOP6	2	100	3.3176e-1 (1.84e-2) =	3.3026e-1 (2.88e-2) =	1.3493e+0 (1.17e-2) -	3.3245e-1 (1.92e-2) =	1.9802e+0 (7.24e-1) -	**3.2176e-1 (1.84e-2)**
	2	500	3.3553e-1 (8.63e-3) -	3.3485e-1 (8.85e-3) -	2.7571e+0 (1.40e-4) -	3.3211e-1 (7.35e-3) -	2.7669e+0 (4.03e-3) -	**3.2485e-1 (8.85e-3)**
	2	1000	3.3709e-1 (6.34e-3) -	3.3695e-1 (7.80e-3) -	2.7572e+0 (2.52e-4) -	3.6225e-1 (6.59e-3) -	2.7652e+0 (3.21e-3) -	**3.2695e-1 (7.80e-3)**
LIRCMOP7	2	100	**1.2077e-1 (7.45e-3) +**	1.4354e-1 (1.43e-2) =	1.6811e+0 (2.03e-4) -	NaN (NaN) -	1.7181e+0 (7.97e-1) -	1.4587e-1 (9.88e-3)
	2	500	**1.2595e-1 (7.31e-3) +**	1.2949e-1 (8.59e-3) +	3.4278e+0 (2.00e-2) -	NaN (NaN) -	2.9651e+0 (7.94e-1) -	1.9476e-1 (1.02e-2)
	2	1000	**1.2541e-1 (4.64e-3) +**	1.2871e-1 (4.73e-3) +	3.4326e+0 (3.05e-4) -	NaN (NaN) -	3.4424e+0 (2.96e-3) -	2.7854e-1 (1.24e-2)
LIRCMOP8	2	100	**2.0524e-1 (2.36e-2) +**	2.1283e-1 (1.86e-2) =	1.6810e+0 (1.93e-4) -	NaN (NaN) -	1.5477e+0 (7.29e-1) -	2.1714e-1 (1.03e-2)
	2	500	**2.1001e-1 (6.59e-3) +**	2.1292e-1 (7.16e-3) +	3.4325e+0 (1.97e-4) -	NaN (NaN) -	3.3493e+0 (3.98e-1) -	2.9125e-1 (8.74e-3)
	2	1000	2.1320e-1 (5.35e-3) +	**2.1176e-1 (5.03e-3) +**	3.4327e+0 (2.27e-4) -	NaN (NaN) -	3.4430e+0 (4.10e-3) -	3.7339e-1 (7.81e-3)
LIRCMOP9	2	100	8.6926e-1 (3.36e-2) =	8.8497e-1 (2.73e-2) =	**5.0206e-1 (2.50e-1) +**	7.9843e-1 (5.50e-2) +	7.7978e-1 (2.08e-1) =	8.6117e-1 (1.20e-1)
	2	500	9.5361e-1 (5.35e-3) +	9.5510e-1 (6.43e-3) +	9.6433e-1 (8.76e-2) +	9.4903e-1 (2.48e-3) +	3.2517e+2 (8.94e+0) -	2.1403e+2 (1.33e+1)
	2	1000	9.5970e-1 (4.49e-3) -	9.5846e-1 (6.23e-3) -	1.0128e+0 (2.24e-2) -	9.7006e-1 (1.20e-2) -	1.5467e+0 (1.87e-1) -	**9.4165e-1 (4.76e-4)**
LIRCMOP10	2	100	6.0561e-1 (1.06e-1) +	6.3777e-1 (9.56e-2) +	**5.3051e-1 (8.82e-2) +**	7.0319e-1 (5.36e-1) +	8.3896e-1 (2.22e-1) =	7.0718e-1 (1.04e-1)
	2	500	8.1069e-1 (1.24e-1) =	8.5598e-1 (1.07e-1) =	**6.3045e-1 (5.85e-2) +**	9.8283e+1 (1.13e+2) -	9.8168e-1 (5.85e-2) -	8.5235e-1 (7.83e-2)
	2	1000	9.2569e-1 (6.91e-2) -	9.1132e-1 (2.18e-3) -	8.7112e-1 (2.89e-3) =	1.0660e+0 (4.17e-2) -	1.0820e+0 (3.20e-1) -	**8.3549e-1 (1.38e-1)**
LIRCMOP11	2	100	4.6404e-1 (2.04e-1) +	4.4877e-1 (1.74e-1) +	**3.5265e-1 (8.87e-2) +**	5.2105e-1 (3.24e-1) +	8.6057e-1 (1.83e-1) -	6.4153e-1 (1.47e-1)
	2	500	7.7007e-1 (3.31e-1) =	9.9780e-1 (2.47e-1) +	**5.9048e-1 (1.27e-1) +**	1.6146e+2 (1.04e+2) -	1.3437e+0 (2.90e-1) -	1.0374e+0 (6.98e-3)
	2	1000	9.3719e-1 (1.80e-1) =	1.1072e+0 (1.37e-2) -	**7.3271e-1 (2.21e-1) +**	1.1219e+0 (7.61e-2) -	1.9098e+0 (4.99e-1) -	1.0597e+0 (7.95e-3)
LIRCMOP12	2	100	6.3386e-1 (5.65e-2) -	6.3872e-1 (4.93e-2) -	**4.5461e-1 (2.41e-1) =**	8.9955e-1 (3.23e-1) -	6.4620e-1 (2.09e-1) -	5.4131e-1 (1.10e-1)
	2	500	6.8265e-1 (7.65e-3) -	6.8683e-1 (9.62e-3) -	7.0182e-1 (4.96e-3) -	7.7527e-1 (1.89e-3) -	3.2762e+2 (1.03e+1) -	**6.6677e-1 (4.89e-4)**
	2	1000	6.8681e-1 (1.01e-2) -	6.8551e-1 (5.76e-3) -	7.2571e-1 (1.31e-2) -	7.9684e-1 (1.11e-2) -	1.2576e+0 (3.85e-1) -	**6.7032e-1 (2.52e-4)**
LIRCMOP13	3	100	6.3535e-2 (3.69e-3) -	5.8721e-2 (2.78e-3) -	1.3127e+0 (1.02e-3) -	2.0185e-1 (2.23e-2) -	1.3022e+0 (2.11e-1) -	**5.5742e-2 (1.90e-3)**
	3	500	2.2262e-1 (1.88e-3) +	**2.2233e-1 (1.59e-3) +**	1.3148e+0 (1.50e-3) -	1.0484e+0 (7.58e-1) -	1.3547e+0 (1.93e-2) -	2.2843e-1 (2.07e-3)
	3	1000	1.3122e+0 (1.36e-3) =	1.2603e+0 (2.17e-1) =	2.1992e+0 (2.18e-1) -	1.3611e+0 (1.46e-2) -	1.6154e+0 (2.94e-1) -	**1.1142e+0 (4.14e-1)**
LIRCMOP14	3	100	4.4019e-2 (9.90e-4) -	5.1326e-2 (1.83e-3) -	1.2692e+0 (1.19e-3) -	1.7044e-1 (1.84e-2) -	1.2512e+0 (2.30e-1) -	**4.1943e-2 (9.48e-4)**
	3	500	1.8417e-1 (1.21e-3) -	1.8607e-1 (2.05e-3) -	1.2707e+0 (1.22e-3) -	1.8927e+0 (6.22e-1) -	1.3027e+0 (6.03e-3) -	**1.8049e-1 (1.14e-3)**
	3	1000	1.2677e+0 (1.42e-3) =	1.2116e+0 (2.38e-1) =	2.1500e+0 (2.48e-1) -	1.3290e+0 (4.28e-2) -	1.7990e+0 (2.48e-1) -	**1.0286e+0 (4.62e-1)**
+/−/= (n = 100)			3/6/5	2/7/5	3/10/1	3/9/2	0/12/2	
+/−/= (n = 500)			5/7/2	6/7/1	3/11/0	2/12/0	0/14/0	
+/−/= (n = 1000)			3/8/3	3/9/2	1/12/1	1/13/0	0/14/0	

D. Scalability to the High-Dimensional Search Space

In this section, we assess the performance of LCMOEA alongside nine selected CMOEA competitors on fourteen scalable LIRCMOP problems with dimensions n = 100, 500, and 1000. We include two CMOEAs specifically designed for large-scale CMOPs, namely LMOCSO [74] and POCEA [62], for comparative analysis. The termination condition is uniformly set as $FE_{\max} = 1000n$ for all cases. The detailed results, including IGD, IGD+, and HV metrics, are presented in Tables A23 to A25 of the appendix. Notably, PPS, CMOEAD, DC-NSGAIII, TPEA, LMOCSO, and POCEA exhibit suboptimal performance in solving most of these large-scale LIRCMOP problems, with LMOCSO struggling to find feasible solutions for specific instances. In contrast, LCMOEA attains the best IGD, IGD+, and HV results across all LIRCMOP instances, followed by CCMO, EMCMO, and ICMA. The scalability advantage of LCMOEA becomes evident in solving 1000-dimensional LIRCMOP problems, emphasizing its superior adaptability as the search space scales up. The favorable results achieved by LCMOEA primarily result from its learning-aided search, facilitating the rapid convergence of the evolutionary population towards the CPF.

Although LCMOEA demonstrated superior performance in tackling large-scale LIRCMOP problems, some challenges persisted in the final results. For instance, when solving both 30-D and 100-D LIRCMOP6 problems, LCMOEA exhibited the best performance; however, the final IGD values in these cases showed significant discrepancies. To visually represent these results, Fig. 6 illustrates the final feasible solutions obtained by LCMOEA for these scenarios. It's apparent that the population generated by LCMOEA tends to converge locally when addressing the 100-D LIRCMOP6 problems. Similar challenges were observed in solving the LIRCMOP7 problem, as depicted in Fig. 7. This predicament may be attributed to two potential factors: 1) the learning-aided search strategy proposed might be prone to causing local convergence; 2) the computational resources may be insufficient, requiring additional resources for exploring the entire CPF. Future efforts should focus on efficiently addressing these challenges to enhance the efficacy of solutions for large-scale CMOPs.

Additionally, considering the training of two MLP models in the learning-aided differential evolution, the computational

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2024.3380366

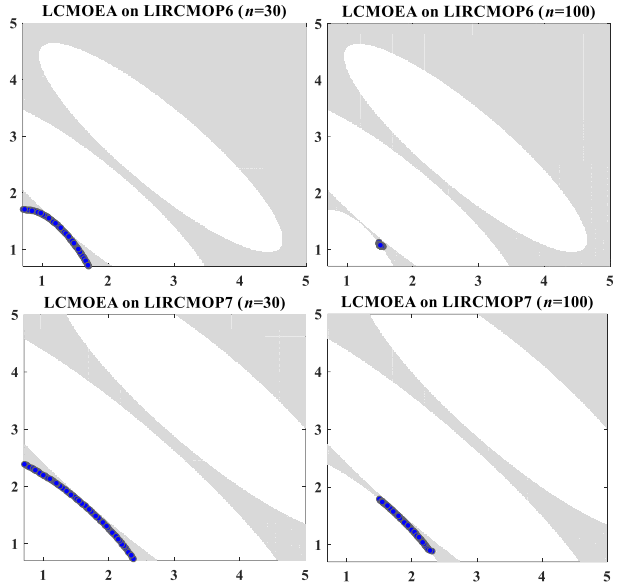


Fig. 7 Illustration of the final feasible nondominated solutions obtained by LCMOEA in solving LIRCMOP6-7 problems with n = 30 and n = 100.

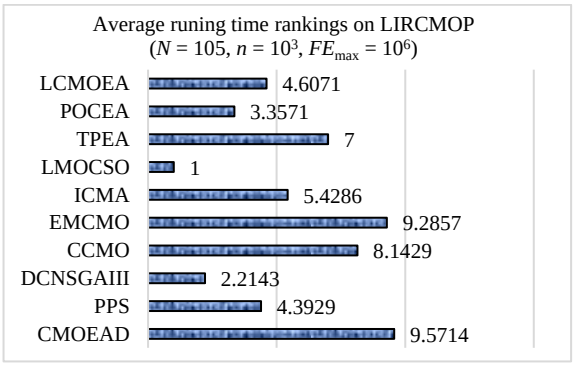


Fig. 8 Illustration of the running time rank on LCMOEA and its competitors in solving LIRCMOP problems with n = 1000 and $FE_{\max}$ = 1000000.

efficiency of LCMOEA is compared with its competitors in terms of running time. The average running times (in seconds: s) for solving 1000-D LIRCMOP problems are presented in Table A26 in the appendix, and their relative rankings are visualized in Fig. 8. LCMOEA's ranking falls in the middle, demonstrating superior computational efficiency compared to certain contemporary CMOEAs like TPEA, EMCMO, and CCMO. This emphasizes that the time complexity of LCMOEA is comparable to most of the competitors.

E. Results on a Real-World Test Suite

Table A27 in the appendix lists the HV results obtained by LCMOEA and its nine competitors when solving fifty CMOP problems (RWMOPs) designed based on practical problems. The termination condition is set as $FE_{\max}$ = 500000. Compared to the nine competitors, including PPS, CMOEAD, CTAEA, CCMO, ICMA, EMCMO, DCNSGAIII, MTCMO, and TPEA, LCMOEA's performance in solving these RWMOP problems is obviously more outstanding. It is matched only by ICMA and TPEA, which get (+/−/=) as (6/13/14) and (10/15/8), respectively. However, under the provision of relatively sufficient computing resources ($FE_{\max}$=500000), the performance of all these CMOEAs in solving RWMOP problems cannot meet the requirements, and even no solver can find feasible solutions for RWMOP36 to RWMOP49. Perhaps these RWMOP problems are too complex to solve, or the computational resources provided are not enough, but either way, there is a lot of room for improvement in the performance of LCMOEA and existing CMOEA.

F. Ablation Study and Parameter Analysis

LCMOEA involves two key components: a learning-aided search (LDE) and clustering-aided selection. Here, we'll test their effectiveness in solving nine DASCMOP problems, and the IGD results are provided in Table A28 of the appendix. Firstly, the effectiveness of LDE is verified by ablation study, in which Eq. (7) is tailored into the following four variants:

$$\begin{cases} \text{LDE1}: \mathrm{x}^c = \mathrm{x} + 0.5(y^2 - \mathrm{x}) + r(\mathrm{x}^1 - \mathrm{x}^2) \\ \text{LDE2}: \mathrm{x}^c = \mathrm{x} + 0.5(y^1 - \mathrm{x}) + r(\mathrm{x}^1 - \mathrm{x}^2) \\ \text{LDE3}: \mathrm{x}^c = \mathrm{x} + r(\mathrm{x}^1 - \mathrm{x}^2) \\ \text{LDE4}: \mathrm{x}^c = \mathrm{x} + \alpha(y^1 - \mathrm{x}) + (0.5 - \alpha)(y^2 - \mathrm{x}) \end{cases} \tag{19}$$

It can be seen from the results that without the guidance of y^1 (learned by the MLP on constraint-ignored task) in LDE1 or y^2 (learned by the MLP on feasibility-first task) in LDE2, the performance will be worse (vs. LDE), and they can only get the comparison result (+/−/=) as (0/4/5). It performs worst, i.e., (+/−/=) is (1/6/2), when neither y^1 nor y^2 is available in LDE3, which proves the effectiveness of these two learned improvement vectors in guiding the evolutionary search. Moreover, excluding the final random term, i.e., $r(\mathrm{x}^1\text{-}\mathrm{x}^2)$, in LDE4 results in a significant performance decline, with 7 out of 9 cases showing notably worse results. The absence of the random term may lead to insufficient diversity in LDE's search, causing the online updated MLP models to collapse and converge to a local optimum more easily. Furthermore, LDE5 replaces y^1 with a random non-dominated solution and y^2 with a random CDP-based non-dominated solution. The failure to perform better than the original LDE on any of the cases proves that y^1 and y^2 learned by the MLP models are more advantageous in solving these DASCMOP problems.

In addition, parameter α in LDE is analyzed. α is set to a random value within [0, 0.5] and fixed as α =0.5, respectively. Correspondingly, the comparison results are 1/4/4 and 1/7/1, which validates that our adaptive setting of its value in (18) is relatively reasonable. This setup resembles a two-stage process: the initial phase concentrates on exploring near the UPF, while the subsequent phase centers on exploration towards the CPF. Detailed discussions are provided in the appendix.

Achieving a balance in directing the population between the DPF and UPF necessitates aligning the clustering-aided environmental selection with controlled preferences for representative selection in each cluster. Thus, the influence of parameter w in (16) is also analyzed, where w is set as follows:

$$\begin{cases} \text{Case1}: if\ \ FE < 0.25FE_{\max}\ \ w = 1.0; else\ w = 0.1 \\ \text{Case2}: if\ \ FE < 0.50FE_{\max}\ \ w = 1.0; else\ w = 0.1 \\ \text{Case3}: if\ \ FE < 0.75FE_{\max}\ \ w = 1.0; else\ w = 0.1 \end{cases} \tag{20}$$

The experimental findings highlight the substantial impact of

the *w* on LCMOEA's performance. Based on the results, it can be inferred that adopting the values depicted in Fig. A1 is a reasonable choice. Ultimately, the analysis reveals that, in addressing these DASCMOP problems, the clustering-aided selection plays a pivotal role to interact with the learning-aided search. In solving the CMOPs with separated CPF and UPF, along with the presence of DPF, LCMOEA outperforms its competitors by balancing convergence, diversity, and feasibility. This superiority is evident in the populations obtained by solving DASCMOP1-4 and LIRCMOP5-12 illustrated in Figs. A2-15 of the appendix. The setting of α and *w* still needs to be further studied in our future work.

G. Two MLP models in Improving Offspring Performance

To evaluate the impact of the two MLP models on the performance improvement of newly generated solutions, we compare the quality of solution x^c generated according to $x^c=x+(y^1-x)$ or $x^c=x+(y^2-x)$ with the quality of its corresponding parent solution x. In the first case, y^1 is derived from mode M_1 that focuses on constraint-ignored task. In this case, x^c is assessed against x based on the weighted sum value guided by the reference vector attached to x (the reference vector with the smallest angle value to x). If x^c outperforms x, *T* (starting at 0) is added 1; if x^c performs worse, *T* is decremented by 1. Similarly, in the second case, y^2 is generated by M_2, tailored for the feasibility-first task, and x^c is only compared to x based on the *CV* value. If the *CV* value of x^c is smaller than that of x, *T* is incremented by 1. If x^c gets a greater CV, *T* is decremented by 1; otherwise *T* is adjusted as in the first case. The performance gains ratio of M_1 or M_2 is defined as *T*/*N*, where *N* is the population size. Fig. A14 in the appendix displays the results of the performance improvement gain rate of the two models tested on the DASCMOP1 to DASCMOP7 problems for these two cases. Detailed discussions are provided in the appendix.

V. Conclusions and Future Work

This work has proposed a learning-aided constrained multiobjective evolutionary algorithm (LCMOEA) to tackle the challenges posed by a diverse set of constrained multiobjective optimization problems (CMOPs). To enhance search efficiency, we propose a learnable differential evolution (LDE) guided by two improvement vectors derived from well-trained multilayer perceptron models—specifically designed for constraint-ignored and feasibility-first tasks. Additionally, we propose an enhanced clustering-aided environmental selection, integrating a comprehensive indicator for the selection of representative solutions within each cluster. Through the synergy of learning-aided search and clustering-aided selection, LCMOEA achieves a well-balanced population, ensuring diversity, convergence, and feasibility. Extensive experiments underscore its superiority over existing CMOEAs, demonstrating remarkable scalability in solving high-dimensional CMOPs. Future research directions include refining learning models, exploring real-world applications, solving large-scale CMOPs, conducting robustness analyses, and investigating hybrid approaches for further performance enhancements.

References

[1] C. Ramirez-Atencia and D. Camacho, "Constrained multi-objective optimization for multi-uav planning," *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, no. 6, pp. 2467–2484, 2019.

[2] M. Basu, "Economic environmental dispatch using multi-objective differential evolution," *Applied soft computing*, vol. 11, no. 2, pp. 2845–2853, 2011.

[3] A. Sadollah, H. Eskandar, and J. H. Kim, "Water cycle algorithm for solving constrained multi-objective optimization problems," *Applied Soft Computing*, vol. 27, pp. 279–298, 2015.

[4] Z. Ma and Y. Wang, "Evolutionary constrained multiobjective optimization: Test suite construction and performance comparisons," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 6, pp. 972-986, 2019.

[5] C. A. Coello Coello, "Evolutionary multi-objective optimization: a historical view of the field," *IEEE Computational Intelligence Magazine*, vol. 1, no. 1, pp. 28-36, 2006.

[6] Y. Tian, Y. Zhang, Y. Su, X. Zhang, K. C. Tan, and Y. Jin, "Balancing objective optimization and constraint satisfaction in constrained evolutionary multi-objective optimization," *IEEE Transactions on Cybernetics*, vol. 52, no. 9, pp. 9559-9572, 2022.

[7] J. Liang, et al., "A Survey on Evolutionary Constrained Multi-objective Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 27, no. 2, pp. 201-221, 2023.

[8] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[9] T. Takahama and S. Sakai, "Constrained optimization by the constrained differential evolution with gradient-based mutation and feasible elites," in Proc. *IEEE Congr. Evol. Comput*. (CEC), 2006, pp. 1–8.

[10] H. Jain and K. Deb, "An evolutionary many-objective optimization algorithm using reference-point based non-dominated sorting approach, part II: Handling constraints and extending to an adaptive approach," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 602-622, 2014.

[11] Z. Liu and Y. Wang, "Handling constrained multiobjective optimization problems with constraints in both the decision and objective spaces," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 5, pp. 870-884, 2019.

[12] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "A reference vector guided evolutionary algorithm for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 5, pp. 773-791, 2016.

[13] Z. Liu, Y. Wang, and B. Wang, "Indicator-based constrained multiobjective evolutionary algorithms," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 9, pp. 5414–5426, 2021.

[14] Z. Fan, Y. Fang, W. Li, X. Cai, C. Wei, and E. Goodman, "MOEA/D with angle-based constrained dominance principle for constrained multi-objective optimization problems," *Applied Soft Computing*, vol. 74, pp. 621–633, 2019.

[15] J. Zhou, Y. Zhang, J. Zheng, M. Li, "Domination-based Selection and Shift-based Density Estimation for Constrained Multiobjective Optimization," *IEEE Transactions on Evolutionary Computation*, in press, 2022.

[16] F. Ming, W. Gong, L. Wang, L. Gao, "A Constrained Many Objective Optimization Evolutionary Algorithm With Enhanced Mating and Environmental Selections," *IEEE Transactions on Cybernetics*, vol. 53, no. 8, pp. 4934-4946, 2023.

[17] Z. Sun, H. Ren, G. G. Yen, T. Chen, J. Wu, H. An, J. Yang, "An Evolutionary Algorithm With Constraint Relaxation Strategy for Highly Constrained Multiobjective Optimization," *IEEE Transactions on Cybernetics*, vol. 53, no. 5, pp. 3190-3204, 2023.

[18] T. P. Runarsson and X. Yao, "Stochastic ranking for constrained evolutionary optimization," *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 3, pp. 284–294, Sep. 2000.

[19] O. S. Ajani, R. Mallipeddi, S. S. R. M, "I^F_{SDE+} - An Indicator for Constrained Multi-Objective Optimization", arXiv:2305.18734v1

[20] J. Yuan, H. Liu, Y. S. Ong, Z. He, "Indicator-Based Evolutionary Algorithm for Solving Constrained Multiobjective Optimization Problems," *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 2, pp. 379-391, 2022.

[21] J. Liang, H. Lin, C. Yue, K. Yu, Y. Guo, K, Qiao, "Multiobjective

Differential Evolution with Speciation for Constrained Multimodal Multiobjective Optimization," *IEEE Transactions on Evolutionary Computation*, in press, 2022.

[22] M. A. Jan and Q. Zhang, "MOEA/D for constrained multiobjective optimization: Some preliminary experimental results," in Proc. U.K. *Workshop Comput. Intell.* (UKCI), 2010, pp. 1–6.

[23] Z. Liu, Y. Qin, W. Song, J. Zhang, K. Li, "Multiobjective-based Constraint-handling Technique for Evolutionary Constrained Multiobjective Optimization: A New Perspective," *IEEE Transactions on Evolutionary Computation*, in press, 2022.

[24] Y. Wang, Z. Cai, Y. Zhou, and W. Zeng, "An adaptive tradeoff model for constrained evolutionary optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 1, pp. 80–92, Feb. 2008.

[25] T. Ray, H. K. Singh, A. Isaacs, and W. Smith, "Infeasibility driven evolutionary algorithm for constrained optimization," in *Constraint Handling in Evolutionary Optimization*. Berlin, Germany: Springer, 2009, pp. 145–165.

[26] Q. Long, "A constraint handling technique for constrained multi-objective genetic algorithm," *Swarm Evol. Comput.*, vol. 15, pp. 66–79, Apr. 2014.

[27] Y. Zhou, M. Zhu, J. Wang, Z. Zhang, Y. Xiang, and J. Zhang, "Tri-goal evolution framework for constrained many-objective optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 8, pp. 3086–3099, 2018.

[28] G. Woldesenbet, G. G. Yen, and B. G. Tessema, "Constraint handling in multiobjective evolutionary optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 3, pp. 514–525, Jun. 2009.

[29] L. Jiao, J. Luo, R. Shang, and F. Liu, "A modified objective function method with feasible-guiding strategy to solve constrained multi-objective optimization problems," *Appl. Soft Comput.*, vol. 14, pp. 363–380, 2014.

[30] R. Jiao, S. Zeng, C. Li, S. Yang, and Y. S. Ong, "Handling constrained many-objective optimization problems via problem transformation," *IEEE Transactions on Cybernetics*, vol. 51, no. 10, pp. 4834-4847, 2021.

[31] K. Li, R. Chen, G. Fu, and X. Yao, "Two-archive evolutionary algorithm for constrained multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 2, pp. 303–315, 2019.

[32] Q. Bao, M. Wang, G. Dai, X. Chen, Z. Song, S. Li, "A dual-population based bidirectional coevolution algorithm for constrained multi-objective optimization problems," *Expert Syst. Appl.*, vol. 215: 119258, 2023.

[33] Y. Tian, T. Zhang, J. Xiao, X. Zhang, and Y. Jin, "A coevolutionary framework for constrained multi-objective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 1, pp. 102–116, 2021.

[34] J. Liang, K. Qiao, K. Yu, B. Qu, C. Yue, W. Guo, and L. Wang, "Utilizing the relationship between unconstrained and constrained Pareto fronts for constrained multiobjective optimization," *IEEE Transactions on Cybernetics*, pp. 1–14, 2022.

[35] K. Qiao, et al., "A Self-Adaptive Evolutionary Multi-Task Based Constrained Multi-Objective Evolutionary Algorithm," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 7, no. 4, pp. 1098-1112, 2023.

[36] Z. Liu, Fan Wu, J. Liu, Y. Qin, K. Li, "Constrained Multiobjective Optimization with Escape and Expansion Forces," *IEEE Transactions on Evolutionary Computation*, in press, 2023.

[37] K. Qiao, K. Yu, B. Qu, J. Liang, H. Song, and C. Yue, "An evolutionary multitasking optimization framework for constrained multiobjective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 2, pp. 263–277, 2022.

[38] J. Zou, et al., "A Multi-Population Evolutionary Algorithm Using New Cooperative Mechanism for Solving Multi-Objective Problems with Multi-Constraint," *IEEE Transactions on Evolutionary Computation*, in press, 2023.

[39] F. Ming, W. Gong, D. Li, L. Wang, L. Gao, "A Competitive and Cooperative Swarm Optimizer for Constrained Multi-objective Optimization Problems," *IEEE Transactions on Evolutionary Computation*, in press, 2022.

[40] R. Jiao, B Xue, M. Zhang, "A Multiform Optimization Framework for Constrained Multiobjective Optimization," *IEEE Transactions on Cybernetics*, vol. 53, no. 8, pp. 5165 - 5177, 2023.

[41] K. Yang, J. Zheng, J. Zou, F. Yu, S. Yang, "A dual-population evolutionary algorithm based on adaptive constraint strength for constrained multi-objective optimization," *Swarm and Evolutionary Computation*, vol. 77: 101247, 2023.

[42] K. Qiao, K. Yu, B. Qu, J. Liang, H. Song, C. Yue, H. Lin, and K. C. Tan, "Dynamic auxiliary task-based evolutionary multitasking for constrained multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 27, no. 3, pp. 642-656, 2023.

[43] F. Ming, W. Gong, L. Wang, and C. Lu, "A tri-population based co-evolutionary framework for constrained multi-objective optimization problems," *Swarm and Evolutionary Computation*, vol. 70: 101055, 2022.

[44] F. Ming, W. Gong, L. Wang, and L. Gao, "Constrained multi-objective optimization via multitasking and knowledge transfer," *IEEE Transactions on Evolutionary Computation*, pp. 1–15, 2022.

[45] K. C. Tan, L. Feng, M. Jiang, "Evolutionary Transfer Optimization - A New Frontier in Evolutionary Computation Research," *IEEE Computational Intelligence Magazine*, vol. 16, no.1, pp. 22-33, 2021.

[46] S. Liu, Q. Lin, L. Feng, K. C. Wong, K. C. Tan, "Evolutionary Multitasking for Large-Scale Multiobjective Optimization," *IEEE Transactions on Evolutionary Computation*, in press: 1-15, 2022.

[47] Z. Fan, W. Li, X. Cai, H. Li, C. Wei, Q. Zhang, K. Deb, and E. Goodman, "Push and pull search for solving constrained multi-objective optimization problems," *Swarm and Evolutionary Computation*, vol. 44, pp. 665–679, 2019.

[48] K. Zhang, Z. Xu, G. G. Yen, L. Zhang, "Two-Stage Multi-Objective Evolution Strategy for Constrained Multi-Objective Optimization," *IEEE Transactions on Evolutionary Computation*, in press, 2022.

[49] F. Ming, W. Gong, H. Zhen, S. Li, L. Wang, Z. Liao, "A simple two-stage evolutionary algorithm for constrained multi-objective optimization," *Knowledge-Based Systems*, vol. 228: 107263, 2021.

[50] M. Ming, R. Wang, H. Ishibuchi, T. Zhang, "A Novel Dual-Stage Dual-Population Evolutionary Algorithm for Constrained Multiobjective Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 5, pp. 1129-1143, 2022.

[51] K. Yu, J. Liang, B. Qu, and C. Yue, "Purpose-directed two-phase multiobjective differential evolution for constrained multiobjective optimization," *Swarm and Evolutionary Computation*, vol. 60, p. 100799, 2021.

[52] L. Li, C. He, W. Xu, and L. Pan, "Pioneer selection for evolutionary multiobjective optimization with discontinuous feasible region," *Swarm and Evolutionary Computation*, vol. 65, p. 100932, 2021.

[53] C. Qin, F. Ming, W. Gong, Q. Gu, "Constrained multi-objective optimization via two archives assisted push–pull evolutionary algorithm," *Swarm and Evolutionary Computation*, vol. 75: 101178, 2022.

[54] J. Zuo, J. Luo, Y. Liu, S. Yang, J. Zheng, "A flexible two-stage constrained multi-objective evolutionary algorithm based on automatic regulation," *Information Sciences*, vol. 634, pp. 227-243, 2023.

[55] F. Ming, W. Gong, L. Wang, "A Two-Stage Evolutionary Algorithm With Balanced Convergence and Diversity for Many-Objective Optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 10, pp. 6222-6234, 2022.

[56] H. Ma, H. Wei, Y. Tian, R. Cheng, and X. Zhang, "A multi-stage evolutionary algorithm for multi-objective optimization with complex constraints," *Information Sciences*, vol. 560, pp. 68–91, 2021.

[57] R. Sun, J. Zou, Y. Liu, S. Yang, J. Zheng, "A Multi-stage Algorithm for Solving Multi-objective Optimization Problems with Multi-constraints," *IEEE Transactions on Evolutionary Computation*, in press, 2022.

[58] Q. Zhu, Q. Zhang, and Q. Lin, "A constrained multiobjective evolutionary algorithm with detect-and-escape strategy," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 5, pp. 938–947, 2020.

[59] M. Zuo, D. Gong, Y. Wang, X. Ye, B. Zeng, F. Meng, "Process Knowledge-guided Autonomous Evolutionary Optimization for Constrained Multiobjective Problems," *IEEE Transactions on Evolutionary Computation*, in press, 2023.

[60] J. Zou, R. Sun, S. Yang, and J. Zheng, "A dual-population algorithm based on alternative evolution and degeneration for solving constrained multi-objective optimization problems," *Information Sciences*, vol. 579, pp. 89–102, 2021.

[61] K. Yu, J. Liang, B. Qu, Y. Luo, and C. Yue, "Dynamic selection preference-assisted constrained multiobjective differential evolution," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 5, pp. 2954–2965, 2022.

[62] C. He, R. Cheng, Y. Tian, X. Zhang, K. C. Tan, Y. Jin, "Paired Offspring Generation for Constrained Large-Scale Multiobjective Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 3, pp. 448-462, 2021.
[63] Q. Lin, S. Liu, K. C. Wong, C. A. Coello Coello, J. Chen, "A Clustering-based Evolutionary Algorithm for Many-objective Optimization Problems," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 3, pp. 391-405, 2019.
[64] S. Liu, Q. Yu, Q. Lin, K. C. Tan, "An adaptive clustering-based evolutionary algorithm for many-objective optimization problems," *Information Sciences*, vol. 537, pp. 261-283, 2020.
[65] S. Liu, J. Zheng, Q. Lin, K. C. Tan, "Evolutionary Multi and Many-objective Optimization via Clustering for Environmental Selection," *Information Sciences*, vol. 578, pp. 930-949, 2021.
[66] Y. Tian, R. Cheng, X. Zhang, Y. Jin, "PlatEMO: A MATLAB platform for evolutionary multi-objective optimization," *IEEE Computational Intelligence Magazine*, vol. 12, no. 4, pp. 73-87, 2017.
[67] Z. Fan, W. Li, X. Cai, H. Li, C. Wei, Q. Zhang, K. Deb, and E. Goodman, "Difficulty adjustable and scalable constrained multi-objective test problem toolkit," *Evolutionary Computation*, vol. 28, no. 3, pp. 339-378, 2020.
[68] Y. Zhou, Y. Xiang, and X. He, "Constrained multiobjective optimization: Test problem construction and performance evaluations," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 1, pp. 172-186, 2021.
[69] Z. Fan, W. Li, X. Cai, H. Huang, Y. Fang, Y. You, J. Mo, C. Wei, and E. Goodman, "An improved epsilon constraint-handling method in MOEA/D for CMOPs with large infeasible regions," *Soft Computing*, vol. 23, pp. 12491-12510, 2019.
[70] C. A. Coello Coello, N. C. Cortes, "Solving multiobjective optimization problems using an artificial immune system," *Genetic Programming and Evolvable Machines*, vol. 6, no. 2, pp. 163-190, 2005.
[71] H. Ishibuchi, H. Masuda, Y. Tanigaki, Y. Nojima, "Modified distance calculation in generational distance and inverted generational distance" *Proceedings of the International Conference on Evolutionary Multi-Criterion Optimization*, pp. 110-125, 2015.
[72] E. Zitzler, L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257-271, 1999.
[73] Y. Sun, B. Xue, M. Zhang, G. G. Yen, "A new two-stage evolutionary algorithm for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 5, pp. 748-761, 2019.
[74] Y. Tian, X. Zheng, X. Zhang, Y. Jin, "Efficient large-scale multi-objective optimization based on a competitive swarm optimizer," *IEEE Transactions on Cybernetics*, vol. 50, no. 8, pp. 3696-3708, 2020.
[75] S. Liu, Q. Lin, J. Li, K. C. Tan, "A Survey on Learnable Evolutionary Algorithms for Scalable Multiobjective Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 27, no. 6, pp. 1941-1961, 2023.

Songbai Liu (Member IEEE) received the B.S. degree from Changsha University and the M.S. degree from Shenzhen University, China, in 2012 and 2018, respectively. He received the Ph.D. degree from Department of Computer Sciences, City University of Hong Kong, in 2022.

He is currently an assistant professor in College of Computer Science and Software Engineering, Shenzhen University. His research interests include evolutionary algorithms + machine learning, multi-objective optimization, and their applications.

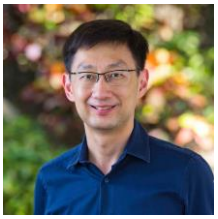
Zeyi Wang received the B.S. degree from Wuhan Institute of Technology, Wuhan, China, in 2022. He is currently a Master student in the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China.

His current research interests include evolutionary multiobjective optimization, evolutionary constrained multiobjective optimization, and their applications.

Qiuzhen Lin (Member, IEEE) received the B.S. degree from Zhaoqing University, China, in 2007, the M.S. degree from Shenzhen University, China, in 2010, and the Ph.D. degree from the Department of Electronic Engineering, City University of Hong Kong, Hong Kong, in 2014. He is currently an Associate Professor with the College of Computer Science and Software Engineering, Shenzhen University. He has been published more than 100 research papers since 2008. His research interests include artificial immune systems, multi-objective optimization, and dynamic systems. He is an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTATIONAL INTELLIGENCE.

Jianqiang Li (Member, IEEE) received the B.S. and Ph.D. degrees in automation from the South China University of Technology, Guangzhou, China, in 2003 and 2008, respectively.

He is a professor at the College of Computer and Software Engineering of Shenzhen University. He led five projects of the National Natural Science Foundation and four projects of the Natural Science Foundation of Guangdong Province, China. His current research interests include robotics, embedded systems, and Internet of Things.



Kay Chen Tan (Fellow, IEEE) received the B.Eng. degree (First Class Hons.) and the Ph.D. degree from the University of Glasgow, U.K., in 1994 and 1997, respectively. He is currently a Chair Professor (Computational Intelligence) of the Department of Computing, the Hong Kong Polytechnic University. He has published over 300 refereed articles and seven books. Prof. Tan is currently the Vice-President (Publications) of IEEE Computational Intelligence Society, USA. He has served as the Editor-in-Chief of the IEEE Computational Intelligence Magazine from 2010 to 2013 and the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION from 2015 to 2020, and currently serves as the Editorial Board Member for more than ten journals. He is the Chief Co-Editor of Springer Book Series on Machine Learning: Foundations, Methodologies, and Applications.