

A Constrained Learning-Based Competitive Swarm Optimizer for Large-scale Multi-Objective Optimization

Yongfeng Li, Lingjie Li, *Member, IEEE*, Qiuzhen Lin, *Member, IEEE*, Zhong Ming Victor. C. M. Leung, *Life Fellow, IEEE*, and Carlos A. Coello Coello, *Fellow, IEEE*

Abstract—Competitive swarm optimizer (CSO) is considered as a prominent paradigm for solving large-scale multi-objective optimization problems (LMOPs). However, the pairwise random competition (PRC) mechanism used in most existing CSOs may limit their performance in solving LMOPs due to the following reasons. First, when the winner particle obtained by PRC is of poor quality, it may limit the learning effect of its corresponding loser particle. Second, due to the stochastic nature of PRC, the evolutionary direction of the loser particles may be drastically perturbed over the iterations, thus slowing down their convergence speed. To alleviate the above issues, this paper proposes a constrained learning-based competitive swarm optimizer for tackling LMOPs, called CL-CSO. First, CL-CSO adopts a set of reference vectors to divide the original objective space into several subregions. Second, CL-CSO designs a constrained learning-based (CL) strategy, including the intra-subregion learning and cross-subregion learning strategy, which let the loser particles only learn from the winner particles in their intra subregions or neighboring subregions, respectively. Moreover, CL-CSO designs a Gaussian model assisted evolutionary strategy to help the evolution of winner particles, aiming to further improve the diversity and quality of winner particles. This way, the learning effect of particles and the overall convergence speed can be significantly enhanced. Compared to several competitive algorithms for tackling LMOPs, experimental results show that CL-CSO performs well in solving two well-known benchmark LMOPs (containing 2-3 objectives and 500 to 5000 decision variables), as

well as real-world instance selection problems.

Index Terms—Competitive swarm optimizer, large-scale optimization, multi-objective optimization, constrained learning.

I. INTRODUCTION

MULTI-objective optimization problems (MOPs) often exist in many real-world applications, such as traffic signal control [1], electrical machine design [2], GPU energy optimization [3], virtual machine scheduling [4], and flowshop scheduling [5]. In general, MOPs need to optimize multiple conflicting objectives, which usually are equally optimal solutions called Pareto-optimal solutions (PS) [6] to solve MOPs. The mapping of PS onto the objective space is called Pareto-optimal front (PF) [6]. Recently, some multi-objective evolutionary algorithms (MOEAs) have been proposed for tackling MOPs, including decomposition-based [7]–[10], Pareto-based [11]–[13] and indicator-based MOEAs [14]–[16]. However, when tackling MOPs with a large number of decision variables (i.e., the well-known large-scale MOPs (LMOPs)), the performance of these MOEAs degrades dramatically [17], which can be attributed to the curse of dimensionality [18]. To effectively address LMOPs, a series of large-scale MOEAs (LMOEAs) have been designed, which can be broadly classified into the following three categories: divide and-conquer based LMOEAs, problem transformation based LMOEAs, and efficient search strategy based LMOEAs.

The divide and-conquer based LMOEAs use some decision variable analysis (DVA) methods [19], [20] or grouping based techniques [21], [22] to reduce the search space, thus speeding up the solving of LMOPs. The problem transformation based LMOEAs translate the original search space into several small-scale subspaces, so as to reduce the complexity of the target problems [23]–[25]. The efficient search strategy based LMOEAs adopt some efficient search strategies or evolutionary operators, which can provide stronger search abilities [26]–[29] to directly solve the original LMOPs.

However, the above first two types of LMOEAs still have some limitations in tackling LMOPs: (1) Divide-and-conquer based LMOEAs commonly consume excessive evaluation resources to conduct DVA or mislead the search direction under the incorrect grouping result; (2) Problem transformation based LMOEAs tend to risk the loss of useful search space information. On the other hand, efficient search strategy based LMOEAs directly solve the original LMOPs via the efficient search mechanism, which can improve the optimization performance when tackling LMOPs. More details about these three types of LMOEAs are introduced in Section II-B.

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grants 62376163 and 62506238; in part by the Guangdong Regional Joint Foundation Key Project under Grant 2022B1515120076; and in part by the Scientific Research Capacity Enhancement Program for Key Construction Disciplines in Guangdong Province under Grant 2024ZDJ063, and in part by the Shenzhen Natural Science Foundation (the Stable Support Plan Program) under Grant 20231122104038002. Carlos A. Coello Coello was also partially supported by the Basque Government through the BERC 2022-2025 program and by the Ministry of Science and Innovation: BCAM Severo Ochoa accreditation CEX2021-001142-S/MICIN/AEI/10.13039/501100011033. (Corresponding Authors: *Lingjie Li* and *Qiuzhen Lin*)

Y.F. Li and Q.Z. Lin are affiliated with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China. (email of Q. Lin: qiuzhlin@szu.edu.cn).

L.J. Li is affiliated with the College of Big Data and Internet, Shenzhen Technology University, Shenzhen 518118, China. (email: lil-jingjie@sztu.edu.cn).

Z. Ming is affiliated with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China, and he is also affiliated with the College of Big Data and Internet, Shenzhen Technology University, Shenzhen 518118, China.

V.C.M. Leung is affiliated with the Artificial Intelligence Research Institute, Shenzhen MSU-BIT University, Shenzhen 518172, China, and is also affiliated with Department of Electrical and Computer Engineering, The University of British Columbia, Vancouver, Canada V6T 1Z4.

Carlos A. Coello Coello is with the Department of Computer Science, CINVESTAV-IPN (Evolutionary Computation Group), México, D.F. 07300, MÉXICO. He is also (as part of a sabbatical leave) with the Basque Center for Applied Mathematics (BCAM) & Ikerbasque, Spain.

In particular, the competitive swarm optimizer (CSO) is considered as an important and popular search paradigm to handle LMOPs due to its efficient search capability and easy implementation [30]. Unlike other swarm intelligence frameworks such as particle swarm optimizer (PSO) [29] or ant colony optimizer (ACO) [31], which often suffer from scalability issues due to premature convergence [32], [33] or insufficient exploitation [31], CSOs inherently adopt a competition-based mechanism, making them more effective in the large-scale search space [34]. Therefore, this paper focuses on the CSO-based LMOEAs.

Nevertheless, most existing CSOs use the pairwise random competition (PRC) mechanism to guide the evolution of loser particles, which may limit their performance in solving LMOPs due to the following reasons. First, when the winner particle obtained by the PRC mechanism is of poor quality, it may limit the learning effect of its corresponding loser particle, which has been explained in [35]. Second, due to the stochastic nature of PRC, the evolutionary direction of the loser particles may be drastically perturbed during the iteration process, thus slowing down the convergence of loser particles. In addition, most existing CSOs pay little attention to the evolution of winner particles, which actually affects their performance to a large extent. Although some improvements were made in [36] by employing a neural network model to improve the quality of winner particles, its model training often involves multiple complex steps (e.g., structure design, parameter settings, etc), which inevitably incurs additional costs. Therefore, the study of CSOs, including the learning ways of loser particles as well as the evolutionary methods of winner particles, deserves further study.

Given above, this paper proposes a constrained learning-based CSO for solving LMOPs, called CL-CSO. Specifically, to enhance the learning effect of loser particles, CL-CSO first uses a set of reference vectors to divide the objective space into several subregions. Then, CL-CSO designs a new constrained learning-based (CL) strategy, which restricts the learning direction of loser particles. On the other hand, to improve the diversity and quality of winner particles, CL-CSO designs a Gaussian model (GM) assisted evolutionary strategy to evolve winner particles. Therefore, the issues mentioned above can be effectively mitigated by this way. In summary, the main contributions of this paper are clarified as follows:

- 1) A constrained learning-based strategy is designed in CL-CSO, which restricts the loser particles only learn from the winner particles in their intra subregions or neighboring subregions. This way, the learning effect of loser particles can be significantly improved.
- 2) A GM-assisted evolutionary strategy is designed in CL-CSO, where the winner particles are sampled for constructing the GM, and then the output particles based on the well-constructed GM are used to evolve the winner particles. This way, the diversity and quality of winner particles are improved.
- 3) To justify the validity of CL-CSO, two well-known benchmark suites of LMOPs (i.e., LSMOP1-LSMOP9 [17] and IMF1-IMF9 [37]) and three real-world instance selection problems [25] are adopted in our experiments.

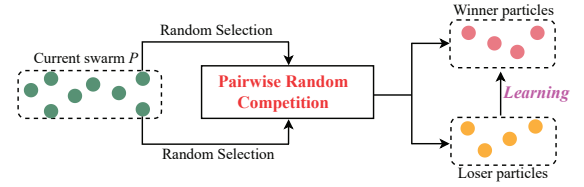


Fig. 1: The pairwise random competitive mechanism in CSOs.

Experimental results show that CL-CSO outperforms several representative LMOEAs, i.e., FDV [38], LMEA [39], LMOCSO [34], S-ECSSO [40], LSMOEA/D [18], CCSO [33] and NN-CSO [36].

The rest of this paper is organized as follows. Firstly, some backgrounds and related studies about LMOEAs, and Gaussian model are provided in Section II. Then, the details of our proposed CL-CSO are introduced in Section III. Afterwards, the experimental discussions are presented in Section IV. Finally, the conclusions and future direction are given in Section V.

II. PRELIMINARY

A. Competitive Swarm Optimizer

Cheng *et. al* [41] first introduced the PRC mechanism in the evolutionary process of particles to enhance its ability in solving LMOPs, which is well known as CSO. Specifically, the CSO randomly selects two particles from the current swarm and then compares these two selected particles based on some specific metrics (e.g., fitness value, crowding distance, convergence degree, etc.). Then, the particle with better performance is regarded as the winner particle, while the other particle is regarded as the loser particle. Finally, the loser particle updates its velocity and position by learning from corresponding winner particle, as follows:

$$\begin{aligned} V_l &= r_1 V_l + r_2 (X_w - X_l) + \varphi r_3 (\bar{X} - X_l) \\ X_l &= X_l + V_l \end{aligned} \quad (1)$$

where r_1, r_2 and r_3 represent three random vectors generated at $[0, 1]^D$, \bar{X} means the average position of all particles in current swarm and φ denotes the control parameter. To further understand the evolutionary process and the PRC mechanism in CSOs, Fig. 1 gives a schematic diagram of a generic CSO.

B. Literature Review of LMOEAs

In this section, a detailed literature review of LMOEAs is given. Specifically, the LMOEAs can be classified into divide-and-conquer LMOEAs, problem transformation-based LMOEAs and efficient search strategy based LMOEAs, elaborated as follows:

1) Related Work of Divide-and-conquer based LMOEAs

In general, the divide-and-conquer based LMOEAs can be further broadly classified into the following two subcategories.

i) Decision variable analysis (DVA) methods:

The first subcategory is designed by using the DVA methods [19], [20], which analyzes the characteristic of each decision variable and then divides the decision variables into several groups by some specific indicators (e.g., interrelation and contributions,

etc.). Specifically, the analysis methods for control variables and interdependent variables were proposed in MOEA/DVA [19], which classifies the decision variables as three types of decision variables, i.e., convergent, diversified and mixed variables. Additionally, the DVA was transformed into the feature selection problems in LERD [20], which allocates the variables with similar importance into the same group.

ii) Grouping based methods: The second subcategory is designed based on the grouping methods [21], [22], which allocates the decision variables into several groups without consuming evaluation resource. For instance, the random grouping approach was applied in [21] to randomly divide the original decision variables into multiple groups, aiming to accelerate the solving of LMOPs. Besides, in [22], all decision variables are divided into some groups by the differential grouping method and each group of decision variables are optimized independently.

2) Related Work of Problem Transformation based LMOEAs

The second category of LMOEAs is designed using problem transformation techniques [23]–[25]. For example, a problem transformation technique was proposed in [23], which transforms the original huge search space into several small subspaces. Additionally, a new framework based on the problem reformulation was proposed in APTEA [24], which reformulates the original search space into several low-dimensional space with the autoencoder. Similarly, two different unsupervised neural networks were used in MOEA/PSL [25] to learn the distribution information of the population.

3) Related Work of Efficient Search Strategy based LMOEAs

The efficient search strategy based LMOEAs design some efficient search tricks to improve the optimization performance when solving LMOPs. Among efficient search approaches, CSOs have shown the promising performance in both large-scale single-objective optimization problems (LSOPs) and LMOPs, due to their efficient search ability [41]. Existing CSOs can be broadly divided into two subtypes:

i) CSOs for solving LSOPs: To date, several promising methods of CSOs have been proposed for solving LSOPs. For example, the level-based learning was proposed in LLSO [42], which divides the particles into different levels according to their fitness values and uses the particles in higher levels to optimize particles in lower levels. A segment-based predominant learning was devised in SPLSO [30], which divides entire decision variables space into several segments, aiming to improve the exploration ability of CSO for each segment. In addition, a two-phase learning was devised in TPLSO [43], which uses the tri-competition mechanism to update the particles in mass learning phase and evolves the elite particles in elite learning phase. Although the above CSOs for solving LSOPs enhance the search ability of algorithms in the huge search space by the competition mechanism, most of them ignore the evolution of winner particles, which also plays the crucial impact for performance improvements [36]. Therefore, if the evolution of winner particles falls into stagnation, CSOs for solving LSOPs may trap into the premature convergence.

ii) CSOs for solving LMOPs: Recently, CSOs have shown the good performance in addressing LMOPs. Specifically, the new particle updating strategy was proposed in LMOCSO [34], which considers the acceleration term and adds it into the position updating strategy to accelerate the convergence speed of CSO. Moreover, a tri-particle competition strategy and a sparse search operator were adopted in S-ECSO [40], which aims to improve the search efficacy in solving sparse LMOPs [45]. A comprehensive competitive swarm optimizer was devised in CCSO [33], which enhances the performance of loser particles in several ways, i.e., environmental learning, cognitive learning and social learning. A neural net-enhanced CSO was proposed in [36], which applies the loser particles and winner particles obtained by the PRC mechanism to train a neural network model and then uses the well-trained model to evolve winner particles. Additionally, a flexibility ranking was used in [44] to identify winning particles, which ensures that winning particles with good diversity can be used as the search direction. The above CSOs improve the diversity of loser particles to some extent by random competition mechanism or customized strategy when solving LMOPs, while their convergence speed may be restricted within the limited evaluation resources. Thus, when the convergence speed of loser particles is too slow, the performance of CSOs for solving LMOPs may deteriorate. How to design the suitable learning way for loser particles to improve the convergence speed is crucial for CSOs for solving LMOPs.

Furthermore, different strategies have been employed in these CSO variants to handle multiple conflicting objectives. For instance, a shift-based density estimation (SDE) method was adopted in LMOCSO [34] to compare particles based on convergence and diversity simultaneously. Additionally, the winner and loser particles are determined using Pareto dominance in NN-CSO [36]. The quality of particles is ranked based on a combination of crowding distance and a locally defined convergence indicator in FR-CSO [44], aiming to balance the diversity and convergence. A decomposition-based strategy, i.e., the penalty-based boundary intersection (PBI) method, was employed in CCSO [33] to scalarize multiple objectives and compare particles accordingly. These different strategies reflect a variety of design approaches in handling multiple objectives within the CSO framework, including dominance-based, indicator-based, and decomposition-based approaches.

Due to page limitations, this paper does not aim to provide an exhaustive review of all LMOEAs. More comprehensive reviews about LMOEAs can refer to [46].

C. The Application of Gaussian Model to Multiobjective Optimization Problems

Gaussian model (GM), known as Gaussian distribution, is widely used in statistics and machine learning for modeling continuous data, which is defined by the probability density function as follows:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad (2)$$

TABLE I: Summary of the Contributions, Learning Ways of Loser Particles and Evolutionary Methods of Winner Particles in Existing CSOs.

Algorithms	Year	Contributions	Learning ways of loser particles	Evolutionary methods of winner particles
CSO [41]	2015	the original CSO	PRC mechanism	none operator
SPLSO [30]	2017	segment-based learning and predominant learning	PRC mechanism	none operator
LLSO [42]	2018	level-based learning strategy	level-based learning	none operator
LMOCSO [34]	2020	novel position update strategy	PRC mechanism	mutation operator
TPLSO [43]	2021	mass learning and elite learning	PRC mechanism	none operator
S-ECSO [40]	2022	tri-competition mechanism and sparse operator	PRC mechanism	none operator
CCSO [33]	2022	comprehensive competitive learning strategy	cognitive and social learning	mutation operator
NN-CSO [36]	2023	neural net-enhanced competitive learning strategy	PRC mechanism	neural network operator
FRCO [44]	2025	flexible ranking-based strategy	diverse learning	mutation operator

where μ is the mean and δ^2 is the variance. GM is often employed as the sampling model to enhance offspring generation. By estimating the statistical characteristics (e.g., mean and variance) of a set of promising solutions, GM provides a probabilistic way to generate new candidate solutions that are likely to inherit desirable properties. This model-based search can effectively balance exploration and exploitation, especially in multiobjective optimization scenarios. For example, the GM was adopted to select the valuable solution for enhancing the transfer effect in [47]. Additionally, GM was applied in [48] to generate more promising solutions in the dynamic environment. Based on the GM, some knowledge from the source task was transferred to the target task [49], so as to facilitate the positive transfer. In [50], GM was employed to obtain the distribution of elite solutions, which can help to measure the similarity between source task and target task, aiming to produce more promising solutions. Besides, GM was applied to enhance the search ability of the population [51].

D. Our Motivations

Although numerous algorithms have made good progress in solving LMOPs, they still have some limitations in handling LMOPs, which are clarified as follows:

1) Firstly, divide-and-conquer based LMOEAs mainly suffer from two challenges. On the one hand, DVA based methods can allocate the interrelated variables into the same group to improve the optimization performance, while it tends to consume an unbearable evaluation cost [36]. On the other hand, the grouping based methods divide the variables into different groups without consuming the evaluation resources; however, the interrelated variables may be allocated into different groups, which may mislead the evolutionary direction [52].

2) Secondly, problem transformation based LMOEAs translate the original search space into several low-dimensional subspaces by using transformation techniques, which aim to quickly find the optima solutions. However, some solutions near PS may become unavailable after using problem transformation, which lets the search process fall into local optima [53]. Therefore, how to balance the diversity and convergence is crucial for tackling LMOPs.

3) Thirdly, although some swarm intelligence based frameworks (e.g., PSO and ACO) have shown comparable performance in solving MOPs, they still face some limitations in solving LMOPs due to the premature convergence [32],

[33] and inadequate exploitation [31]. Thus, designing a more efficient nature-inspired paradigm is important for improving the performance when solving LMOPs.

Therefore, it is a nature idea to design an efficient search strategy to directly search in the original search space. Within recent decades, CSO-based LMOEAs have provided a promising direction to directly search in the large-scale search space when solving LMOPs, while they still have some drawbacks. Table I summarizes the main contributions, the learning way of loser particles and the evolutionary method of winner particles in existing CSOs mentioned above. Some drawbacks about existing CSOs are clarified as follows.

1) Regarding the learning ways of loser particles, as shown in the penultimate column of Table I, most existing CSOs [30], [34], [36], [40], [41], [43] follow the PRC mechanism. However, as analyzed in the fifth paragraph of Section I, the PRC mechanism still faces the following challenges. First, the PRC mechanism cannot guarantee that the quality of winner particle is sufficient to improve the performance of its corresponding loser particle [35]. Specifically, the winner particles obtained by the PRC mechanism may have the similar quality to loser particles, leading to the little improvement for the loser particles. Second, it is also prone to perturb the evolutionary direction of loser particles, thus seriously affecting the convergence speed. The reason behind this may be the PRC mechanism, which causes the learning direction of loser particles to vary significantly in each iteration. In this way, the convergence speed of loser particles may be severely limited. Although some meaningful attempts have been made in LLSO [42], CCSO [33], and FRCO [44], they only seem to alleviate the first challenge mentioned above, as the evolutionary direction of the loser particles remains uncertain in each iteration. Therefore, the learning way of loser particles deserves further study.

2) Regarding the evolutionary methods of winner particles, as shown in the last column of Table I, most existing CSOs adopt none operator or a simple mutation operator to evolve the winner particles. However, the quality of winner particles largely determines the performance of CSOs. This is because the evolution of loser particles can be effectively guided if and only if the quality of winner particles is good. In [36], a neural network model was used to assist the evolution of winner particles. Although the experimental results showed that this method would significantly improve the performance of the traditional CSOs, this method brings some inevitable complex steps, including model structure design, model training,

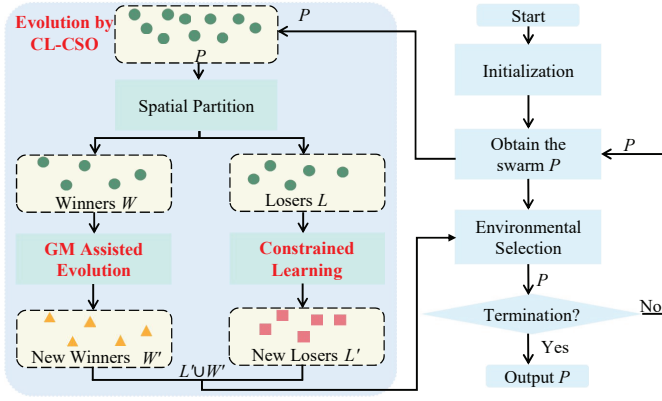


Fig. 2: The outline of the proposed CL-CSO.

parameter settings, etc. Therefore, how to design an efficient evolutionary method for winner particles also deserves further study.

Given above, this paper proposes CL-CSO, which includes the CL-based strategy and the GM-assisted evolutionary strategy. These two strategies are applied to improve the quality of loser particles and winner particles, respectively. With the combination of these two strategies, the evolutionary search can be effectively guided, thereby improving the performance of CL-CSO in solving LMOPs. The details of our proposed CL-CSO are presented in the next section.

III. OUR PROPOSED CL-CSO

A. Framework of the Proposed CL-CSO

To better understand the general framework for CL-CSO, its pseudo-code is provided in **Algorithm 1**, where N is the swarm size and T denotes the number of neighborhood winner particles for each winner particle. Similar to the flowchart of traditional MOEAs, the initialization procedure is first conducted in Lines 1-2. Subsequently, the algorithm enters the main loop. To be specific, the spatial partition process is conducted in Line 4, which divides the objective space into several subregions by using \mathbf{r} and identifies the loser particles \mathbf{L} and winner particles \mathbf{W} of \mathbf{P} in each subregion. More details about the spatial partition process are introduced in **Algorithm 2**. After that, the constrained learning-based strategy is conducted in Line 5 to get the new loser particles \mathbf{L}' by using \mathbf{L} and \mathbf{W} . The details about the constrained learning-based strategy are described in **Algorithm 3**. Next, the GM-assisted evolutionary strategy is performed in Line 6, which is used to obtain the new winner particles \mathbf{W}' . The details of the GM-assisted evolutionary strategy are explained in **Algorithm 4**. Then, the reference vector-guided environmental selection [54] is performed in Line 7, which selects N best particles from the union particle swarm of current swarm \mathbf{P} and the updated particles ($\mathbf{L}' \cup \mathbf{W}'$) as the new swarm \mathbf{P} . Specifically, all candidate solutions are associated with their closest reference vectors based on the minimum acute angle. Within each subregion, the angle-penalized distance (APD) [54] is calculated to balance convergence and diversity. The solution with the smallest APD in each subregion is selected

Algorithm 1 The framework of CL-CSO

Input: N (swarm size), T (the number of neighborhood winner particles for each winner particle).

Output: \mathbf{P} (the best particle set).

- 1: $\mathbf{P} \leftarrow$ Initialize the particle set of size N ;
- 2: $\mathbf{r} \leftarrow$ Generate the uniform reference vectors of size $N/2$;
- 3: **while** the termination condition is not fulfilled **do**
- 4: $\mathbf{L}, \mathbf{W} \leftarrow$ **Spatial-Partition**(\mathbf{P}, \mathbf{r});
- 5: $\mathbf{L}' \leftarrow$ **Constrained-Learning**($\mathbf{L}, \mathbf{W}, T$);
- 6: $\mathbf{W}' \leftarrow$ **GM-Assisted-Evolution**(\mathbf{W}, N);
- 7: $\mathbf{P} \leftarrow$ **Environmental-Selection**($\mathbf{L}' \cup \mathbf{W}' \cup \mathbf{P}, \mathbf{r}$);
- 8: **end while**

for the next generation. The above steps are repeated until the termination condition is reached. Finally, the particle swarm \mathbf{P} is outputted as the best particle set. To further comprehend the outline of CL-CSO, its flowchart is drawn in Fig. 2.

B. Spatial Partition

This section presents the process of spatial partition. Firstly, the objective space is divided into several subregions by using some reference vectors. Subsequently, two particles closest to each reference vector are assigned to the same subregion. Then, two particles in the same subregion are identified as the loser particle and winner particle, respectively, according to their performance comparison. The performance of each particle \mathbf{s} is measured by Tchebycheff [55] in this paper, as defined by,

$$g^{te}(\mathbf{s}|\mathbf{r}^j, \mathbf{z}) = \max_{1 \leq i \leq m} \left\{ r_i^j |f_i(\mathbf{s}) - z_i| \right\} \quad (3)$$

where \mathbf{r}^j is a reference vector, r_i^j denotes i -th objective value of \mathbf{r}^j , and $\mathbf{z} = (z_1, \dots, z_m)$ is the reference point, i.e., $z_i = \min \{f_i(\mathbf{s})\}$ for each $i = 1, \dots, m$. Thus, the particle in the subregion which has the smaller Tchebycheff value is identified as the winner particle. On the contrary, the particle in the sub-region which has the larger Tchebycheff value is considered as the loser particle. Specifically, the reference vectors proposed in [54] is suitable for CL-CSO, which has been validated as a useful strategy to guide the search in the multiobjective subspaces [56].

Algorithm 2 gives the details about the process of spatial partition. In Line 1, the loser particles set \mathbf{L} and the winner particles set \mathbf{W} are initialized. Subsequently, in Lines 2-4, the angle between each particle \mathbf{s} and each reference vector \mathbf{r}^j in \mathbf{r} is computed as follows:

$$\text{angle}(\mathbf{r}^j, \mathbf{s}) = \arccos \left| \frac{\sum_{k=1}^M f_k(\mathbf{r}^j) \cdot f_k(\mathbf{s})}{\sqrt{\sum_{k=1}^M f_k(\mathbf{r}^j)^2} \cdot \sqrt{\sum_{k=1}^M f_k(\mathbf{s})^2}} \right| \quad (4)$$

where $f_k(\mathbf{s})$ indicates the value of k -th objective for \mathbf{s} .

Subsequently, two closest particles to \mathbf{r}^j are added in \mathbf{C}_j by $\text{angle}(\mathbf{r}^j, \mathbf{s})$ in Lines 6-7, so as to achieve the pairwise competition. Here, the smaller value of $\text{angle}(\mathbf{r}^j, \mathbf{s})$ indicates that the particle \mathbf{s} is closer to \mathbf{r}^j . In Lines 9-17, the loser

Algorithm 2 Spatial-Partition (P, r)

Input: P (the current swarm), r (the reference vectors).
Output: W (the winner particles), L (the loser particles).

- 1: $L \leftarrow \emptyset, W \leftarrow \emptyset$;
- 2: **for** each particle s in P **do**
- 3: Compute $angle(r^j, s)$ by (4) for each $r^j \in r$;
- 4: **end for**
- 5: **for** each r^j in r **do**
- 6: Add two particles (i.e., x_1 and x_2) from P closest to r^j into the cluster C_j based on $angle(r^j, s)$;
- 7: $P = P \setminus \{x_1, x_2\}$;
- 8: **end for**
- 9: **for** each cluster C_j in C **do**
- 10: **if** $g^{te}(C_j^1|r^j, z) < g^{te}(C_j^2|r^j, z)$ **then**
- 11: $L = L \cup C_j^1$;
- 12: $W = W \cup C_j^2$;
- 13: **else**
- 14: $L = L \cup C_j^2$;
- 15: $W = W \cup C_j^1$;
- 16: **end if**
- 17: **end for**

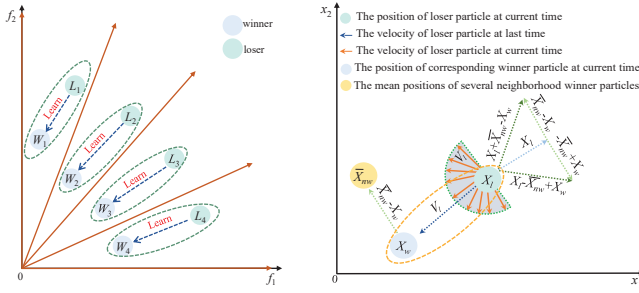


Fig. 3: The schematic of ISL. Fig. 4: The schematic of CSL.

particle and the winner particle in C_j are identified and then added into L and W , respectively. In Lines 10-12, when the Tchbycheff value of the first particle in C_j (C_j^1), i.e., $g^{te}(C_j^1|r^j, z)$, is smaller than the Tchbycheff value of the second particle in C_j (C_j^2), i.e., $g^{te}(C_j^2|r^j, z)$, C_j^2 is added into L and C_j^1 is added into W . On the contrary, in Lines 13-15, C_j^1 is added into L and C_j^2 is added into W . Here, the Tchbycheff value of one particle is computed by (3). Finally, the winner particles W and the loser particles L are returned.

C. Constrained Learning-Based Strategy for Loser Particles

As mentioned in most existing studies about CSOs [30], [34], [36], [40], [41], [43], loser particles only learn from the winner particles that win PRC, which limits the learning effect of loser particles [35]. To alleviate this issue, the constrained learning-based (CL) strategy is proposed in this article. When the above process of spatial partition in **Algorithm 2** is completed, the CL strategy is performed on the loser particles. To be specific, the CL strategy consists of two different learning strategies, clarified as follows:

1) Intra-subregion Learning (ISL): Different from the learning strategies in most existing studies about CSOs [30],

[34], [36], [40], [41], [43] that loser particles only learn from the winner particles in the PRC mechanism, the ISL strategy of CL can accelerate the convergence speed of loser particles. Specifically, when the above process of spatial partition in **Algorithm 2** is completed, loser particles and winner particles in each sub-region are identified. Then, the loser particles learn from the winner particles in the same sub-region. To be specific, the velocity and position of loser particle are updated as follows:

$$\begin{aligned} \mathbf{V}_l &= \mathbf{r}_1 \mathbf{V}_l + \mathbf{r}_2 (\mathbf{X}_w - \mathbf{X}_l) \\ \mathbf{X}_l &= \mathbf{X}_l + \mathbf{V}_l \end{aligned} \quad (5)$$

where \mathbf{r}_1 and \mathbf{r}_2 are two random vectors generated at $[0, 1]^D$, \mathbf{X}_w represents the position of corresponding winner particle in the same sub-region, while \mathbf{V}_l and \mathbf{X}_l stand for the velocity and position of the loser particle, respectively. Fig. 3 gives the schematic of ISL strategy. As observed from Fig. 3, each loser particle learns from its corresponding winner particle in the same subregion, which can significantly accelerate the convergence speed of loser particles.

2) Cross-subregion Learning (CSL): To enhance the exploitation ability of loser particles and fully exploit the information among sub-regions, the CSL strategy of CL is devised in this article. To be specific, after the above process in **Algorithm 2**, the loser particles and the winner particles in all sub-regions are identified. Then, the CSL strategy is performed on the loser particles, so as to enhance the exploitation ability of loser particles. At first, several neighborhood winner particles of each winner particle are identified. Subsequently, the loser particles learn from the useful information between the neighborhood sub-regions. Therefore, the velocity and position of loser particle are updated as below:

$$\begin{aligned} \mathbf{V}_l &= \mathbf{r}_1 \mathbf{V}_l + \mathbf{r}_2 \cdot U(\mathbf{X}_l - |\overline{\mathbf{X}_{nw}} - \mathbf{X}_w|, \mathbf{X}_l + |\overline{\mathbf{X}_{nw}} - \mathbf{X}_w|) \\ \mathbf{X}_l &= \mathbf{X}_l + \mathbf{V}_l \end{aligned} \quad (6)$$

where $\overline{\mathbf{X}_{nw}}$ denotes the average position of several neighborhood winner particles, and U stands for the uniform distribution, which means that a value is randomly selected from the interval $[\mathbf{X}_l - |\overline{\mathbf{X}_{nw}} - \mathbf{X}_w|, \mathbf{X}_l + |\overline{\mathbf{X}_{nw}} - \mathbf{X}_w|]$, and each value has an equal probability of being selected. In this way, the exploitation ability of loser particles can be promoted. Please note that the theoretical analysis about CSL strategy is provided in Section A of the supplementary file. To comprehend the CSL strategy of CL, Fig. 4 provides the schematic of CSL strategy. As seen from Fig. 4, the loser particle has many potential learning directions among its position, as shown in orange arrows. Therefore, the CSL strategy of CL can enhance the exploitation ability of loser particles.

The pseudocode of CL strategy is given in **Algorithm 3**. Firstly, the updated loser particles set (L) is initialized in Line 1. Specifically, T is set as 10 based on the discussions given in Section IV-E. Subsequently, in Lines 2-10, each particle in L (\mathbf{X}_l) performs the ISL strategy or CSL strategy. Specifically, in Lines 3-4, when the random probability $rand$ is smaller than the given probability p , the loser particle \mathbf{X}_l is performed

Algorithm 3 Constrained-Learning (L, W, T)

Input: L (the loser particles), W (the winner particles), T (the number of neighborhood winner particles for each winner particle).

Output: L' (the updated loser particles).

```

1:  $L' \leftarrow \emptyset$ ;
2: for each particle  $X_l$  in  $L$  do
3:   if  $rand < p$  then
4:     Update  $X_l$  with (5);
5:   else
6:      $\bar{X}_{nw} \leftarrow$  Compute the average positions of  $T$ 
       closest neighbourhood of each winner in  $W$ ;
7:     Update  $X_l$  with (6);
8:   end if
9:   Add  $X_l$  into  $L'$ ;
10: end for

```

on ISL strategy and X_l is updated by (5). Contrarily, X_l is performed on CSL strategy in Lines 6-7. The average position of T closest neighbors for each winner in W is computed as \bar{X}_{nw} in Line 6, where T represents the number of neighborhood particles for each winner particle. Afterwards, X_l is updated by (6) in Line 7. To be specific, p is set as 0.9 based on the discussions provided in Section IV-E. Afterwards, the updated loser particle X_l is added into L' in Line 9. Finally, the updated loser particles L' are returned.

D. GM-Assisted Evolutionary Strategy for Winner Particles

In most existing studies about CSOs [30], [33], [34], [40]–[43], the winner particles are saved directly or polynomially mutated [57] to next generation, which weakens the search capability of CSOs. Thus, the further evolution of winner particles is crucial for the performance of CSOs. In this paper, a GM-assisted evolutionary strategy is devised for the evolution of winner particles, as GM has been validated as an efficient evolution strategy in [48]. To be specific, all the winner particles are used for constructing the GM at first. Subsequently, the mean vector and the covariance matrix of decision variables for all winner particles are obtained. Here, the mean vector is computed as follows:

$$\mathbf{u} = \frac{\sum_{i=1}^{N/2} \mathbf{W}^i}{N/2} \quad (7)$$

where \mathbf{W}^i represents the i -th particle of winner particles and N is the population size. In addition, the covariance matrix is computed as follows:

$$\mathbf{cov} = \frac{\sum_{i=1}^{N/2} (\mathbf{W}^i - \mu) (\mathbf{W}^i - \mu)^T}{N/2 - 1} \quad (8)$$

Subsequently, the mean vector and the covariance matrix are adopted as the input of GM, aiming to construct the GM. Then, some particles can be sampled from the output of well-constructed GM. Finally, the sampled particles from well-constructed GM are randomly selected to guide the evolution of winner particles, which can improve the diversity and quality of winner particles.

Algorithm 4 GM-Assisted-Evolution (W, N)

Input: W (the winner particles), N (the population size).

Output: W' (the updated winner particles).

```

1:  $W' \leftarrow \emptyset$ ;
2:  $\mathbf{u} \leftarrow$  The mean vector of the decision variables of  $W$  is
   computed by (7);
3:  $\mathbf{cov} \leftarrow$ The covariance matrix of decision variables of  $W$ 
   is computed by (8);
4:  $Z \leftarrow$  Sample  $N/2$  particles from multivariate normal
   distribution  $N(\mathbf{u}, \mathbf{cov})$ ;
5: for each particle  $X_w$  in  $W$  do
6:    $X_z \leftarrow$  Randomly select one particle from  $Z$ ;
7:   Update  $X_w$  by (9);
8:   Remove  $X_z$  from  $Z$ ;
9:   Add  $X_w$  into  $W'$ ;
10: end for

```

The pseudocode of GM-assisted evolutionary strategy for winner particles is provided in **Algorithm 4**. Firstly, the updated winner particles set W' is initialized in Line 1. Then, the mean vector of decision variables of the winner particles set (W) is computed by Eq. (7) as \mathbf{u} in Line 2. Subsequently, the covariance matrix of W is computed in Line 3, denoted as \mathbf{cov} . In Line 4, \mathbf{u} and \mathbf{cov} are used as the input of GM for constructing the GM and $N/2$ particles are sampled from the well-constructed GM, denoted as Z . Each winner particle in W is updated by the sampled particles from Z in Lines 6-9. To be specific, in Line 6, a particle is randomly selected from Z , denoted as X_z . Subsequently, the velocity and position of each winner particle in W are updated as follows:

$$\begin{aligned} \mathbf{V}_w &= \mathbf{r}_1 \mathbf{V}_w + \mathbf{r}_2 (\mathbf{X}_z - \mathbf{X}_w) \\ \mathbf{X}_w &= \mathbf{V}_w + \mathbf{X}_w \end{aligned} \quad (9)$$

where \mathbf{r}_1 and \mathbf{r}_2 are two random real-valued vectors within $[0, 1]^D$, \mathbf{V}_w and \mathbf{X}_w represent the velocity and position of the winner particle, respectively, and \mathbf{X}_z denotes the position of one particle in the sampled particles from the well-constructed GM. Then, the particle X_z is removed from Z in Line 8. Afterwards, the updated winner particle X_w is added into W' in Line 9. Finally, the updated winner particles W' are returned.

E. Computational Complexity of CL-CSO

At each iteration, CL-CSO contains four main processes, i.e., the spatial division in line 4, the reproduction of loser particles in line 5, the evolution of winner particles in line 6, and the environmental selection in line 7. To be specific, the worst time complexity of the spatial partition in **Algorithm 2** is $O(MN^2)$, where M and N denote the number of objectives and the swarm size, respectively. Additionally, the reproduction of loser particles by **Algorithm 3** requires the time complexity of $O(p(ND) + (1-p)(N^2D))$, where p is the predefined probability and D is the dimension of decision variables. Afterwards, the time complexity of the evolution of winner particles is $O(ND)$. Lastly, the time complexity of environmental selection is $O(MN^2)$, which is as same as that of

NSGA-II [11]. Summarily, the overall time complexity of CL-CSO in one iteration is $O(MN^2 + p(ND) + (1-p)(N^2D))$, which is comparable to that of state-of-the-art LMOEAs like NN-CSO [36] when the value of p is approximately the same as 1.

IV. EXPERIMENTAL STUDIES

To evaluate the performance of our proposed CL-CSO, comprehensive experiments are conducted in this paper. First, CL-CSO is compared with seven competitive LMOEAs in solving two well-known benchmark suites (LSMOP [17] and IMF [37]). Second, CL-CSO is compared with four different kinds of SOTA LMOEAs in solving LSMOP and IMF test suite. Thirdly, the parameter analysis of CL-CSO is conducted. Subsequently, ablation studies on CL-CSO are designed to investigate the superiority of each component. Finally, three real-world problems from instance selection are used to verify the robustness of CL-CSO. Note that the source codes of CL-CSO are available at <https://github.com/xiaoyuge20/CLCSOCODE>.

A. Benchmark Problems and Performance Metrics

Benchmark problems: Here, two benchmark problems (LSMOP1-LSMOP9 [17] and IMF1-IMF9 [37]) are used to investigate the advantage of CL-CSO. For LSMOP suite, the number of objectives (M) is set as 2 and 3, and for IMF problems, M is set as 2, except for IMF4 and IMF8 problems with $M = 3$. The numbers of decision variables (D) are set as $D = \{500, 1000, 2000, 5000\}$ for all problems.

Performance indicators: To evaluate both the convergence and diversity of the final solutions across all algorithms, we utilize the inverted generational distance (IGD) [58] and hypervolume (HV) [59] as the performance measure. Note that a smaller IGD (a larger HV) value indicates a superior performance for the given test problem. Specifically, we evenly sample 10,000 reference points from the true PF of each test problem to calculate the IGD. Additionally, the reference point for HV calculation is set to a vector of ones, i.e., (1.0, 1.0) for bi-objective and (1.0, 1.0, 1.0) for tri-objective optimization problems, as recommended in [60]. The IGD values are computed using the standard implementation in PlatEMO [60], which does not perform normalization before calculation. This setting is to ensure the fair comparison among all competitors. In contrast, HV is calculated with objective values normalized to [0,1], following the default implementation in PlatEMO. It's worth noting that each problem undergoes 20 independent runs, and we record the mean and standard deviation of the IGD and HV values. Moreover, for robust statistical analysis, we employ the Wilcoxon rank sum test at a significance level of 0.05 for experimental comparisons. In the tables below, symbols such as "+", "-", and "=" signify that other compared algorithms have the better, worse or similar performance over our algorithm.

B. Comparison Algorithms and Parameter Settings

Seven competitive LMOEAs are adopted for comparison with CL-CSO, i.e., FDV [38], LMOCSO [34], CCSO [33],

TABLE II: Summary of the IGD value based comparison of CL-CSO with seven comparative algorithms on LSMOP problems.

Test Sets (+/-/=)	D	FDV	LMEA	LMOCSO	S-ECSO	LSMOEA/D	CCSO	NN-CSO
LSMOP	500	0/9/0	0/9/0	0/9/0	2/4/3	1/8/0	1/7/1	2/6/1
	1000	0/9/0	0/9/0	0/9/0	2/5/2	0/9/0	1/8/0	1/6/2
	2000	0/9/0	0/9/0	1/8/0	2/5/2	1/8/0	0/7/2	1/6/2
	5000	0/9/0	0/9/0	0/9/0	2/4/3	0/8/1	0/8/1	1/8/0
	ALL	0/36/0	0/36/0	1/35/0	8/18/10	2/33/1	2/30/4	5/26/5

S-ECSO [40], NN-CSO [36], LMEA [39] and LSMOEA/D [18]. To be specific, FDV is one representative DR-based LMOEA. LMOCSO, CCSO, S-ECSO and NN-CSO are four search-based LMOEAs using CSO. LMEA and LSMOEA/D are two DVA-based LMOEAs. For a fair comparison, the corresponding parameters of these compared LMOEAs are set as suggested in corresponding papers. In our proposed CL-CSO, the probability used to adjust the ratio between the ISL strategy and the CSL strategy is set as $p = 0.9$. For CSL strategy, the number of neighbourhood of winner particles is set to $T = 10$. The values of p and T are set based on the parameters sensitivity analysis of this paper, which is detailed in Section IV-E. As for environmental selection, the penalty coefficient is set as $\alpha = 2$, as suggested by [34]. Besides, for each test suite, the population size (N) is set as $N = 100$ and the termination condition (the maximum function evaluation) is configured as $100 \times D$, as suggested by [61]. Note that all the parameter settings are summarized in Table A. 1 of the supplementary file.¹

C. Comparisons with Seven Competitive LMOEAs

In this part, to verify the superiority of our proposed CL-CSO, seven competitive LMOEAs, including FDV [38], LMEA [39], LMOCSO [34], S-ECSO [40], LSMOEA/D [18], CCSO [33] and NN-CSO [36], are compared in 9 LSMOP test problems [17] and 9 IMF test problems [37].

1) Comparisons on LSMOP Problems

Here, CL-CSO is compared with FDV, LMOCSO, S-ECSO, LSMOEA/D, LMEA, CCSO and NN-CSO on LSMOP problems with $M = 2$, and $D = \{500, 1000, 2000, 5000\}$. The detailed average IGD results of CL-CSO and these seven LMOEAs on LSMOP problems with 2 objectives and D ranging from 500 to 5000 are presented in Table A. 2 of the supplementary file, in which the best result in each case is written in bold. As seen in Table A. 2, CL-CSO achieves the best performance in 26 out of 36 cases, S-ECSO obtains 6 best results, LSMOEA/D only gets 1 best result, CCSO only gains 1 best result and NN-CSO only obtains 2 best results, while FDV, LMEA and LMOCSO doesn't get any best result. According to the last row of Table A. 2, CL-CSO is worse than other competitors only in 0, 0, 1, 8, 2, 2 and 5 cases, respectively, while it outperforms the seven competitors in 36, 36, 35, 18, 33, 30 and 26 out of 36 LSMOP problems. Therefore, our proposed CL-CSO has the better superiority than these advanced LMOEAs on most of LSMOP problems. Due to page limitations, the specific HV results of

¹The supplementary file of CL-CSO can be downloaded at: <https://github.com/xiaoyuge20/CLCSOCODE>

TABLE III: Summary of the IGD value based comparison of CL-CSO with seven comparative algorithms on IMF problems.

Test Sets (+/-/=)	D	FDV	LMEA	LMOCSO	S-ECSO	LSMOEA/D	CCSO	NN-CSO
IMF	500	0/9/0	0/9/0	0/9/0	0/8/1	0/9/0	0/9/0	0/6/3
	1000	0/9/0	0/9/0	0/9/0	2/7/0	0/9/0	0/9/0	2/6/1
	2000	0/9/0	0/9/0	0/9/0	0/8/1	0/9/0	0/9/0	1/7/1
	5000	0/9/0	0/9/0	0/9/0	1/7/1	0/9/0	0/9/0	0/6/3
	ALL	0/36/0	0/36/0	0/36/0	3/30/3	0/36/0	0/36/0	3/25/8

all algorithms on solving LSMOP problems are given in Table A. 3 of the supplementary file.

Similarly, when solving LSMOPs with $M = 2$ and a specific value of D , the separate statistical results of seven LMOEAs compared to CL-CSO in addressing LSMOP benchmark suites based on IGD metric are provided in Table II. Obviously, CL-CSO outperforms all compared LMOEAs on all LSMOP test problems. When compared to other LMOEAs, CL-CSO is never beaten by these LMOEAs in each case of D when solving LSMOP test problems. Therefore, our proposed CL-CSO shows the better superiority in handling LSMOP test problems with 500-5000 decision variables when compared to these advanced LMOEAs.

Additionally, to further highlight the superiority of our proposed CL-CSO, it is compared with FDV, LMOCSO, S-ECSO, LSMOEA/D, LMEA, CCSO and NN-CSO on LSMOP problems with $M = 3$ and $D = \{500, 1000, 2000, 5000\}$. Due to page limitations, the detailed average IGD and HV comparison results of all algorithms are shown in Tables A. 4-A. 5 of the supplementary file. As provided in Table A. 4, CL-CSO is better than these seven compared algorithms on 33, 32, 31, 21, 29, 27, and 19 problems out of total 36 problems, while it performs worse than the seven competitors only in 3, 4, 5, 13, 6, 8, and 11 out of 36 LSMOP problems. Therefore, our proposed CL-CSO shows better performance than these competitive LMOEAs on LSMOP problems with 3 objectives.

2) Comparisons on IMF Problems

To further study the generalization of CL-CSO, the IMF test suite [37] is also selected for performance comparisons. Here, CL-CSO is compared with FDV, LMEA, LMOCSO, S-ECSO, LSMOEA/D, CCSO and NN-CSO on 36 IMF problems. Note that the detailed average IGD results of CL-CSO and seven LMOEAs on IMF problems with $D = \{500, 1000, 2000, 5000\}$ are presented in Table A. 6 of the supplementary file. According to the summarized results collected in the last row of Table A. 6, CL-CSO is better than other seven competitors in 36, 36, 36, 30, 36, 36 and 25 cases, respectively, while it performs worse than other seven competitors only in 0, 0, 0, 3, 0, 0 and 3 cases. In addition, CL-CSO achieves 27 best cases out of 36 IMF test problems, while other competitors only obtain 9 best cases. Thus, it can be inferred that CL-CSO has the better superiority than seven compared algorithms on most of IMF test instances. Due to page limitations, the specific HV results of all algorithms on solving IMF problems are provided in Table A. 7 of the supplementary file.

Furthermore, under all dimensions of decision variables (D), the statistical results of CL-CSO with its seven compared LMOEAs in handling IMF problems are given in Table III. Obviously, CL-CSO has the much better performance than seven LMOEAs on most of IMF test suites. As seen in

Table III, CL-CSO is obviously better than all compared LMOEAs on most of IMF problems with each case of $D = \{500, 1000, 2000, 5000\}$. Therefore, it can be concluded that our proposed CL-CSO exhibits the superiority on addressing most of IMF test suites when compared to these advanced LMOEAs.

3) Further Discussions on Experimental Results

Some conclusions can be drawn based on the above extensive experimental results, as follows.

When compared with these competitive LMOEAs, our proposed CL-CSO shows the superior performance in solving most test problems. For LSMOP problems, our proposed CL-CSO exhibits the satisfactory performance especially in solving LSMOP1, LSMOP2, LSMOP4, LSMOP5, LSMOP7, LSMOP8 and LSMOP9 problems with 2 objectives. Specifically, in unimodal problems like LSMOP1 and LSMOP5, the reason behind the better performance of CL-CSO in these two problems may be that the proposed CL strategy can significantly improve the quality of loser particles. Additionally, CL-CSO also shows its superiority in solving other LSMOP problems with mixed or multi-modal landscapes (i.e., mixed LSMOP2, LSMOP4, LSMOP8, LSMOP9 and multi-modal LSMOP7), which can be attributed to the fact that the elaborated GM-assisted evolutionary strategy improves the diversity of population. In addition to LSMOP problems, CL-CSO also achieves the best optimization performance in solving IMF1, IMF3, IMF5, IMF7 and IMF9 problems. Particularly, the reason behind the advantage of our proposed CL-CSO in solving IMF1 and IMF3 with linear variable linkage can be attributed to the customized CL-strategy, which can further enhance the search ability of loser particles. When solving IMF5, IMF7 and IMF9 with nonlinear variable linkage, the designed GM-assisted evolutionary strategy contributes to improving the diversity of winner particles, thereby mitigating the issue of local optima. Summarily, the tailored CL strategy and GM-assisted evolutionary strategy both have the significant impact to the performance of CL-CSO.

Compared to the first two types of LMOEAs (i.e., DVA-based and transformation-based methods), our proposed CL-CSO exhibits significant superiority on most of the adopted test problems. Specifically, FDV is beaten by our proposed CL-CSO in all cases of LSMOP and IMF test problems. FDV uses the fuzzy evolution to reduce the search scope, which can improve the search efficacy but lose some optimal solutions in the meantime. As for LMEA and LSMOEA/D, they behave worse than our proposed CL-CSO in most cases of LSMOP and IMF test suites. The reason for this might be that LMEA and LSMOEA/D consume unbearable evaluation cost to conduct DVA. Thus, LMEA and LSMOEA/D behave poorly when limited evaluation cost is given.

Compared to CSO-based LMOEAs (i.e., LMOCSO, S-ECSO, CCSO, and NN-CSO), our proposed CL-CSO also achieves significant advantages in most of the adopted test problems for the following two main reasons: firstly, in terms of the learning way for loser particles, LMOCSO, S-ECSO and NN-CSO adopt PRC mechanism to evolve loser particles, which restricts the learning effect of loser particles. As for CCSO, it uses cognitive and social learning to evolve loser

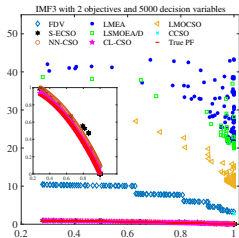


Fig. 5: The final population obtained by CL-CSO and its seven compared algorithms on the IMF3 problem.

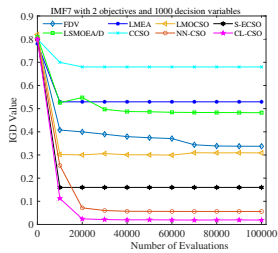


Fig. 6: The convergence profile of CL-CSO with its seven compared algorithms on IMF7 problems.

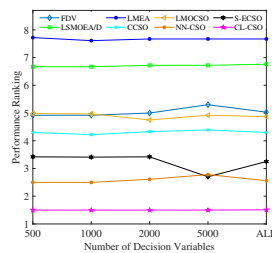
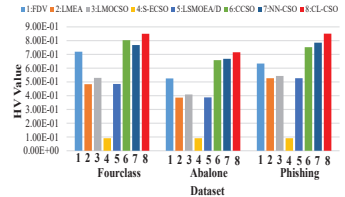


Fig. 7: The illustration of the average performance ranks obtained by all algorithms over each problem under all three datasets of instance selection.



particles, which may slow down the convergence speed of loser particles since the evolutionary direction of loser particles is uncertain. Secondly, for the evolutionary way of winner particles, LMOCSO, S-ECSO and CCSO only use the mutation or none operator for the evolution of winner particles, which have little improvement on the quality of winner particles. As for NN-CSO, it uses a neural network to evolve winner particles, so as to accelerate the convergence speed of winner particles while ignoring the diversity of winner particles. To be summarized, LMOCSO, S-ECSO, CCSO and NN-CSO behave poorly due to the lack of strong search ability.

In summary, as our proposed CL-CSO is much better than seven compared algorithms in solving LSMOP and IMF problems, the issues of compared algorithms mentioned above can be alleviated due to the designed CL strategy and GM-assisted evolutionary strategy.

4) Visualizations of the Final Solution Sets

To visualize the performance difference between all algorithms, Fig. 5 and Fig. A. 1 of the supplementary file depict the final population by all algorithms on IMF3, IMF5, LSMOP2 and LSMOP4 problems. Specifically, the number of objectives and decision variables in these test problems are set as 2 and 5000, respectively. Among these test problems, IMF3 has a concave PF with linear variable linkage, IMF5 has a convex PF with nonlinear variable linkage, LSMOP2 has a linear PF with linear variable linkage and LSMOP4 has a linear PF with linear variable linkage. As seen in Fig. 5 and Fig. A. 1, the final population obtained by our proposed CL-CSO shows the best convergence and distribution when compared to the population obtained by other algorithms. In summary, CL-CSO shows the best performance over other compared algorithms, especially when compared with LMOCSO, S-ECSO, CCSO and NN-CSO. Therefore, the superiority of the proposed CL strategy and GM-assisted evolutionary strategy can be confirmed.

5) Discussions on Convergence Profile

To observe the convergence trend of CL-CSO with its seven compared LMOEAs, their convergence profiles on IMF7, IMF9, LSMOP1 and LSMOP9 with $M = 2$ and $D = 1000$ are depicted in Fig. 6 and Fig. A. 2 of the supplementary file. Specifically, IMF7 has the concave PF and nonlinear variable linkage, IMF9 has the convex PF and nonlinear variable linkage, LSMOP1 has the linear PF and linear variable linkage and LSMOP9 has the disconnected PF and nonlinear

variable linkage. According to these figures, only S-ECSO and NN-CSO converge faster than CL-CSO at the early stage in some cases. The reason behind S-ECSO can be attributed to the sparse operator in S-ECSO, which can accelerate the convergence speed of population. As for NN-CSO, the reason behind this can be attributed to the adopted neural network operator in NN-CSO, aiming to accelerate the convergence speed of winner particles. However, S-ECSO and NN-CSO perform worse than CL-CSO in the later stage of all cases, which indicate that S-ECSO and NN-CSO appear to trap into local optima. Thus, CL-CSO shows better performance, which can verify the superiority of our CL strategy and GM-assisted evolutionary strategy. As for other compared algorithms, they converge slower than CL-CSO and have the worse IGD level than that of CL-CSO, which can demonstrate the superiority of our proposed CL-CSO. Therefore, the advantage of CL-CSO can be attributed to the proposed CL strategy and GM-assisted evolutionary strategy. Summarily, our proposed CL-CSO shows the superiority when compared with its seven compared algorithms in solving LMOPs.

6) Average Performance Rank

To visually show the advantages of our proposed CL-CSO over other algorithms when tackling LSMOP test suites and IMF test instances with $M = 2$ and $D = \{500, 1000, 2000, 5000\}$, CL-CSO and its seven compared LMOEAs are ranked by the Friedman test using the *KEEL* platform [62] with D ranging from 500 to 5000 for LSMOP and IMF test problems. Note that a lower ranking score indicates a better performance of corresponding algorithm. Fig. 7 shows the ranking scores of all algorithms based on all test problems with D ranging from 500 to 5000. As seen in Fig. 7, among all comparison algorithms, CL-CSO shows the significant superiority for test problems with $D = \{500, 1000, 2000, 5000\}$, as it has the lowest scores among all dimensions, i.e., 1.5, 1.5, 1.5 and 1.5. Finally, CL-CSO has the lowest average ranking score of 1.5 for all problems with all dimensions, as can be seen in the last column “ALL” of Fig. 7, while other competitors only gain the average ranking scores of 5.03, 7.67, 4.87, 3.25, 6.76, 4.30 and 2.56, respectively. The reason behind the superiority of our proposed algorithm can be attributed to the proposed CL strategy and GM-assisted evolutionary strategy.

TABLE IV: Summary of Statistical Comparison Results of CL-CSO and Four SOTA LMOEAs in Tackling LSMOP Test Problems Based on IGD and HV Results.

Metric	Test problems +/-/=	CL-CSO vs SOTA LMOEAs			
		MPSOEBDCD	LERD	LSTPA	APTEA
IGD	on LSMOP	8/59/5	10/60/2	13/53/6	24/41/7
	on IMF	0/36/0	0/34/2	1/28/7	1/30/5
	in total	8/95/5	10/94/4	14/81/13	25/71/12
HV	on LSMOP	9/46/17	7/48/17	10/39/23	8/41/23
	on IMF	0/36/0	0/36/0	0/30/6	6/21/9
	in total	9/82/17	7/84/17	10/69/29	14/62/32

D. Comparisons with Four SOTA Large-scale Multiobjective Evolutionary Algorithms

To further verify the superiority of our proposed CL-CSO, it is compared with four SOTA LSMOEAs, (i.e., a PSO-based LMOEA (MPSOEBDCD [29]), a divide-and-conquer based LMOEA (LERD [20]), an efficient search strategy based LMOEA (LSTPA [63]), and a problem transformation based LMOEA (APTEA/R [64])), in solving LSMOP test suite with $M = \{2, 3\}$ and $D = \{500, 1000, 2000, 5000\}$, and IMF test problems with $D = \{500, 1000, 2000, 5000\}$. Due to page limitations, the IGD and HV values obtained by these four SOTA LMOEAs and our proposed CL-CSO in solving all test problems are provided in Tables A. 8-A. 13 of the supplementary file. Table IV summarizes the statistical results of all algorithms in solving all test problems based on IGD and HV results. As observed from Table IV, in the case of IGD metric, our proposed CL-CSO performs better than MPSOEBDCD, LERD, LSTPA, and APTEA/R in 95, 94, 81, and 71 cases, while it only performs worse in 8, 10, 14, and 25 cases. As for HV results, our proposed CL-CSO performs better than other SOTA LMOEAs in 82, 84, 69, and 62 cases, while it only performs worse in 9, 7, 10, and 14 cases in total of 108 test problems. Therefore, the superiority of our proposed CL-CSO can be demonstrated, which can be attributed to our proposed CL strategy and GM-assisted evolutionary strategy.

E. Parameters Sensitivity Analysis of CL-CSO

In this part, the sensitivity to the value of given probability p and the neighborhood number of winner particles T are analyzed, as they are important parameters in CL-CSO to affect its performance. Here, five different values of p , i.e., 0, 0.3, 0.6, 0.9 and 1, are considered here for solving IMF test problems with 500 decision variables. Similarly, six different values of T , i.e., 0, 5, 10, 20, 35 and 50, are adopted here for solving IMF test suites with $D = 500$.

Table A. 14 of the supplementary file presents the average IGD values from 20 runs and Friedman's rank [62] of CL-CSO with all combinations of different values of p and T in solving IMF problems with $D = 500$. According to Table A. 14, if p is set relatively large (i.e., $p = 1$), CL-CSO doesn't obtain any best cases. The reason behind this may be that CL-CSO only uses the ISL strategy, which may lead to the insufficient diversity of loser particles. However, when p is set too small (i.e., $p = 0$), CL-CSO also doesn't obtain any best results in solving these IMF problems, which can be attributed that the only use of CSL strategy results in the slow convergence speed

of loser particles. As observed from Table A. 14, CL-CSO with $p = 0.9$ shows the best performance in most cases when solving IMF test problems with 500 decision variables, as it obtains 4 best results out of all 9 test problems, while other two variants (i.e., CL-CSO with $p = 0.3$ and $p = 0.6$) only obtain 2 best results on all 9 test problems, respectively. Therefore, we adopt $p = 0.9$ in this paper.

To determine the value of T , the average IGD values from 20 runs and Friedman's rank [62] of CL-CSO with different T values and a fixed p value of 0.9 for solving IMF test suites with $D = 500$ are given in Table A. 15 of the supplementary file. As seen in the last row of Table A. 15, the rank of CL-CSO significantly improves from $T = 0$ to $T = 5$, and then is similar from $T = 5$ to $T = 20$, while deteriorates from $T = 20$ to $T = 50$. That is, when T is set too small (i.e., $T = 0$), the CSL strategy lacks of neighbourhood solutions for winner particles, which can not improve the diversity of loser particles. However, when T is set large (i.e., $T = \{35, 50\}$), the neighbourhood solutions for winner particles may be far from the loser particles, which limits the learning effect of loser particles. Given that the most cases of optimal results are achieved for $T = 5$ to $T = 20$, it is reasonable to consider values for T within [5, 20].

To further determine the value of T , the comparison experiments on the running time of CL-CSO with $T = 5, 10, 20$ on IMF1 problem with 500 decision variables are conducted. As seen in Fig. A. 3 of the supplementary file, the running time obtained by CL-CSO is steady from $T = 5$ to $T = 10$, while the running time consumed by CL-CSO increases from $T = 10$ to $T = 20$. This means that more neighbourhood solutions need more computational budget. Considering both the performance and running time, T should be set as 5 or 10. However, when T is set as 5, the performance of CL-CSO is significantly worse than that of CL-CSO with $T = 10$ when solving IMF4 with 500 decision variables, which may be attributed to the fact that too few neighbourhood solutions result in the limited search range for winner particles, thus degrading the performance of CL-CSO in some cases. Therefore, considering the performance, running time and the robustness, we adopt $T=10$ in this paper.

F. Ablation Studies of CL-CSO

In this part, ablation studies are performed to validate the superiority of each component contained in CL-CSO. To be specific, the ablation experiments of CL-CSO with its six variants are organized as follows:

1) To verify the effectiveness of CL strategy in CL-CSO, the variant CL-CSO-I is designed, where CL-CSO-I applies the PRC mechanism rather than the CL strategy. To verify the superiority of CL-CSO over CL-CSO-I, LSMOP1 and LSMOP9 test problems with $M = 2$ and $D = 2000$ are adopted. Figs. A. 4 of the supplementary file gives the convergence profiles of CL-CSO and CL-CSO-I in solving LSMOP1 and LSMOP9 test problems with $M = 2$ and $D = 2000$. As learned from Figs. A. 4, CL-CSO outperforms CL-CSO-I in the later stage of evolution. According to Figs. A. 4, the convergence performance of CL-CSO is faster than that of CL-CSO-I at the early stage of evolution. The reason behind this is that

the CL strategy restricts the loser particles to only learn from the high-quality winner particles in the same sub-region or neighbourhood sub-regions, which significantly improves the learning effect of loser particles. Therefore, the superiority of CL strategy can be verified in our proposed CL-CSO.

2) To further verify the robustness of our proposed CL strategy, the variant LMOCSO_CL is devised, where LMOCSO_CL indicates that LMOCSO [34] adopts the CL strategy instead of PRC mechanism. Then, LMOCSO_CL is compared with LMOCSO on solving LSMOP4 and LSMOP8 with $M = 2$ and $D = 2000$. Fig. A. 5 of the supplementary file depicts their convergence profiles on solving LSMOP4 and LSMOP8 with $M = 2$ and $D = 2000$ based on the inverted generational distance (IGD). As seen in Fig. A. 5, LMOCSO_CL not only converges obviously faster than LMOCSO at the early evolutionary stage, but also reaches the better IGD level than LMOCSO at the end of the evolutionary process. The reason behind this phenomenon is that the CL-based strategy lets the loser particles to learn from the more appropriate winner particles, thus enhancing the search ability of loser particles. Therefore, the effectiveness of CL strategy can be demonstrated.

3) To demonstrate the effectiveness of the CSL mechanism in CL strategy, the variants CL-CSO-II, and CL-CSO-III are designed, where CL-CSO-II removes the X_l in the velocity update formula, while CL-CSO-III removes the $|X_{nw} - X_w|$ in the velocity update formula. To demonstrate the advantage of CL-CSO over CL-CSO-II and CL-CSO-III, IMF test suite with $M = 2$ and $D = 500$ are adopted. Table A. 16 of the supplementary file gives the mean IGD results and derivations of CL-CSO, CL-CSO-II and CL-CSO-III in solving IMF test problems with $M = 2$ and $D = 500$. As seen in Table A. 16, CL-CSO only performs worse than CL-CSO-II and CL-CSO-III in 2 and 2 cases, while performs better than CL-CSO-II and CL-CSO-III in 5 and 4 cases in total of 9 test problems. The reason behind this may be attributed to the fact that the combination of X_l and $|X_{nw} - X_w|$ can improve the search ability of the population. Therefore, the superiority of CSL strategy can be demonstrated in our proposed CL-CSO.

4) To verify the superiority of GM-assisted evolutionary strategy for winner particles in CL-CSO, two variants (i.e., CL-CSO-IV and CL-CSO-V) are designed. To be specific, CL-CSO-IV adopts the polynomial mutation (PM) [57] for the evolution of winner particles to replace our GM-assisted evolutionary strategy. As for CL-CSO-V, the winner particles are saved directly (SD) into the next iteration instead of using GM-assisted evolutionary strategy. To verify the advantage of CL-CSO over CL-CSO-IV and CL-CSO-V, LSMOP1 and LSMOP9 test problems with $M = 2$ and $D = 2000$ are used. Fig. A. 6 of the supplementary file provides the IGD results obtained by CL-CSO-IV, CL-CSO-V and CL-CSO on LSMOP1 and LSMOP9 test problems with $M = 2$ and $D = 2000$. As shown in Fig. A. 6, CL-CSO obtains the lower IGD level than CL-CSO-IV and CL-CSO-V, which shows obvious advantages than CL-CSO-IV and CL-CSO-V. Thus, the GM-assisted evolutionary strategy has the greater impact than the PM strategy and the SD strategy in improving the performance of CSO, which can validate the superiority of our designed

GM-assisted evolutionary strategy.

Summarily, these ablation experiments can verify the effectiveness of CL strategy and GM-assisted evolutionary strategy designed in CL-CSO.

G. Application Experiments in Instance Selection

To explore the performance of our proposed CL-CSO in solving real-world application, instance selection (IS) is used to investigate the practicalities of our proposed CL-CSO. IS plays an important role in data mining, which chooses a subset of instances from a large dataset based on some criterions [65]. The goal of IS is to preserve the essential information and relationships that the learning algorithm needs to learn. Therefore, IS can reduce the computational resources during the learning process. IS can be treated as the LMOP, which contains large-scale decision variables and various objectives. In the experiment, the objectives are adopted as minimizing the error rate and the size of selected instances, which are formulated as follows:

$$\min \begin{cases} f_1(x) = |x| \\ f_2(x) = \frac{FP+FN}{TP+TN+FP+FN} \end{cases} \quad (10)$$

where x is the selected instances, $f_1(x)$ is the size of selected instances, and $f_2(x)$ is the error rate of maintaining the most informative and representative instances. TP , TN , FP and FN denote true positives, true negatives, false positives, and false negatives, respectively.

In our experiment, three public datasets are adopted to verify the superiority of our proposed CL-CSO, i.e., Fourclass dataset, Abalone dataset and Phishing dataset, which can be found in UCI machine learning repository [66]. To be specific, Fourclass dataset has 862 samples, Abalone dataset has 4177 samples and Phishing dataset has 11055 samples. Since real-world problems have no true PF, the hypervolume (HV) [59] is used to measure the performance of all algorithms, where a larger value of HV indicates a better performance of the algorithm in solving IS problems. The parameters of all algorithms are set as the same as mentioned in Section IV-B. The termination condition for each algorithm is set as 10000 evaluations. Fig. 8 presents the median HV values of seven algorithms on all datasets. According to Fig. 8, in terms of HV values, it is evident that CL-CSO is statistically better than other compared algorithms for solving IS application problems, which validates the effectiveness and robustness of the proposed method.

V. CONCLUSIONS AND FUTURE WORK

In this paper, a constrained learning-based competitive swarm optimizer, called CL-CSO, has been proposed for solving LMOPs. Specifically, the constrained learning-based strategy is designed in CL-CSO, including two strategies, i.e., the ISL strategy and the CSL strategy. The ISL strategy and CSL strategy are adopted to restrict the loser particles to only learn from the winner particles in their corresponding subregions or neighboring subregions, respectively. By this way, the learning effect of loser particles can be improved. Moreover, the GM is designed as the evolutionary method to

improve the quality of winner particles, aiming to enhance the diversity and quality of winner particles. When compared to several competitive LMOEAs, the experimental results also verified the superior performance of CL-CSO in solving LMOPs and real-world problems, which can be attributed to the designed CL strategy and GM-assisted evolutionary strategy.

In our future work, the machine learning method in improving the performance of CSO will be studied since machine learning methods can extract useful features from the data and provide an effective search path. In addition, how to construct an effective multitasking method for handling LMOPs will also be further studied. Finally, the ability of CSO in solving real-life scheduling optimization problems will also be one part of our future work.

REFERENCES

- [1] P. W. Shaikh, M. El-Abd, M. Khanafer, and K. Gao, "A review on swarm intelligence and evolutionary algorithms for solving the traffic signal control problem," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 1, pp. 48–63, January. 2022.
- [2] M. C. Huber, M. Fuhlrländer, and S. Schöps, "Multi-objective yield optimization for electrical machines using gaussian processes to learn faulty design," *IEEE Transactions on Industry Applications*, vol. 59, no. 2, pp. 1340–1350, March-April. 2023.
- [3] F. Wang, W. Zhang, S. Lai, M. Hao, and Z. Wang, "Dynamic gpu energy optimization for machine learning training workloads," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 11, pp. 2943–2954, November. 2022.
- [4] Z. Xiao, Q. Qiu, L. Li, Y. Feng, Q. Lin, and Z. Ming, "An efficient service-aware virtual machine scheduling approach based on multi-objective evolutionary algorithm," *IEEE Transactions on Services Computing*, pp. 1–14, 2023.
- [5] X. He, Q.-k. Pan, L. Gao, L. Wang, and P. N. Suganthan, "A greedy cooperative co-evolutionary algorithm with problem-specific knowledge for multi-objective flowshop group scheduling problems," *IEEE Transactions on Evolutionary Computation*, vol. 27, no. 3, pp. 430–444, June. 2023.
- [6] F. Gu, H.-L. Liu, Y.-M. Cheung, and M. Zheng, "A rough-to-fine evolutionary multiobjective optimization algorithm," *IEEE Transactions on Cybernetics*, vol. 52, no. 12, pp. 13472–13485, July. 2022.
- [7] C. Zhang, L. Gao, X. Li, W. Shen, J. Zhou, and K. C. Tan, "Resetting weight vectors in MOEA/D for multiobjective optimization problems with discontinuous Pareto front," *IEEE Transactions on Cybernetics*, vol. 52, no. 9, pp. 9770–9783, April. 2022.
- [8] L. He, K. Shang, Y. Nan, H. Ishibuchi, and D. Srinivasan, "Relation between objective space normalization and weight vector scaling in decomposition-based multiobjective evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 27, no. 5, pp. 1177–1191, 2023.
- [9] Y.-L. Li, Z.-Y. Chai, F. Tan, and X. Liu, "Temporal data scheduling in internet of vehicles using an improved decomposition-based multi-objective evolutionary algorithm," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 5, pp. 5282–5295, May. 2023.
- [10] Y.-L. Lan, F. Liu, W. W. Y. Ng, J. Zhang, and M. Gui, "Decomposition based multi-objective variable neighborhood descent algorithm for logistics dispatching," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 5, no. 5, pp. 826–839, October. 2021.
- [11] 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.
- [12] X. Zheng, B. Han, and Z. Ni, "Tourism route recommendation based on a multi-objective evolutionary algorithm using two-stage decomposition and pareto layering," *IEEE/CAA Journal of Automatica Sinica*, vol. 10, no. 2, pp. 486–500, February. 2023.
- [13] X. Cai, H. Sun, Q. Zhang, and Y. Huang, "A grid weighted sum pareto local search for combinatorial multi and many-objective optimization," *IEEE Transactions on Cybernetics*, vol. 49, no. 9, pp. 3586–3598, September. 2019.
- [14] J. M. Sanchez-Gomez, M. A. Vega-Rodríguez, and C. J. Pérez, "An indicator-based multi-objective optimization approach applied to extractive multi-document text summarization," *IEEE Latin America Transactions*, vol. 17, no. 08, pp. 1291–1299, August. 2019.
- [15] C. L. d. V. Lopes, F. V. C. Martins, E. F. Wanner, and K. Deb, "Analyzing dominance move (mip-dom) indicator for multiobjective and many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 3, pp. 476–489, June. 2022.
- [16] X. Cai, Y. Xiao, M. Li, H. Hu, H. Ishibuchi, and X. Li, "A grid-based inverted generational distance for multi/many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 1, pp. 21–34, February. 2021.
- [17] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "Test problems for large-scale multiobjective and many-objective optimization," *IEEE Transactions on Cybernetics*, vol. 47, no. 12, pp. 4108–4121, December. 2017.
- [18] L. Ma, M. Huang, S. Yang, R. Wang, and X. Wang, "An adaptive localized decision variable analysis approach to large-scale multiobjective and many-objective optimization," *IEEE Transactions on Cybernetics*, vol. 52, no. 7, pp. 6684–6696, July. 2022.
- [19] X. Ma, F. Liu, Y. Qi, X. Wang, L. Li, L. Jiao, M. Yin, and M. Gong, "A multiobjective evolutionary algorithm based on decision variable analyses for multiobjective optimization problems with large-scale variables," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 2, pp. 275–298, April. 2016.
- [20] C. He, R. Cheng, L. Li, K. C. Tan, and Y. Jin, "Large-scale multiobjective optimization via reformulated decision variable analysis," *IEEE Transactions on Evolutionary Computation*, vol. 28, no. 1, pp. 47–61, 2024.
- [21] L. M. Antonio and C. A. C. Coello, "Use of cooperative coevolution for solving large scale multiobjective optimization problems," in *2013 IEEE Congress on Evolutionary Computation*, 2013, pp. 2758–2765.
- [22] M. Li and J. Wei, "A cooperative co-evolutionary algorithm for large-scale multi-objective optimization problems," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, ser. GECCO '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 1716–1721. [Online]. Available: <https://doi.org/10.1145/3205651.3208250>
- [23] H. Zille, H. Ishibuchi, S. Mostaghim, and Y. Nojima, "A framework for large-scale multiobjective optimization based on problem transformation," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 2, pp. 260–275, April. 2018.
- [24] S. Liu, J. Li, Q. Lin, Y. Tian, J. Li, and K. C. Tan, "Evolutionary large-scale multiobjective optimization via autoencoder-based problem transformation," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 8, no. 4, pp. 2709–2722, 2024.
- [25] Y. Tian, C. Lu, X. Zhang, K. C. Tan, and Y. Jin, "Solving large-scale multiobjective optimization problems with sparse optimal solutions via unsupervised neural networks," *IEEE Transactions on Cybernetics*, vol. 51, no. 6, pp. 3115–3128, June. 2021.
- [26] C. He, R. Cheng, and D. Yazdani, "Adaptive offspring generation for evolutionary large-scale multiobjective optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 2, pp. 786–798, February. 2022.
- [27] W. Hong, K. Tang, A. Zhou, H. Ishibuchi, and X. Yao, "A scalable indicator-based evolutionary algorithm for large-scale multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 3, pp. 525–537, June. 2019.
- [28] Z. Wang, H. Hong, K. Ye, G.-E. Zhang, M. Jiang, and K. C. Tan, "Manifold interpolation for large-scale multiobjective optimization via generative adversarial networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 8, pp. 4631–4645, 2023.
- [29] D. Li, L. Wang, L. Li, W. Guo, Q. Wu, and A. Lerch, "A large-scale multiobjective particle swarm optimizer with enhanced balance of convergence and diversity," *IEEE Transactions on Cybernetics*, vol. 54, no. 3, pp. 1596–1607, 2024.
- [30] Q. Yang, W.-N. Chen, T. Gu, H. Zhang, J. D. Deng, Y. Li, and J. Zhang, "Segment-based predominant learning swarm optimizer for large-scale optimization," *IEEE Transactions on Cybernetics*, vol. 47, no. 9, pp. 2896–2910, September. 2017.
- [31] Z. Wang, S. Gao, M. Zhou, S. Sato, J. Cheng, and J. Wang, "Information-theory-based nondominated sorting ant colony optimization for multi-objective feature selection in classification," *IEEE Transactions on Cybernetics*, vol. 53, no. 8, pp. 5276–5289, 2023.
- [32] E. T. Oldewage, A. P. Engelbrecht, and C. W. Cleghorn, "The merits of velocity clamping particle swarm optimisation in high dimensional spaces," in *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2017, pp. 1–8.

- [33] S. Liu, Q. Lin, Q. Li, and K. C. Tan, "A comprehensive competitive swarm optimizer for large-scale multiobjective optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 9, pp. 5829–5842, September. 2022.
- [34] Y. Tian, X. Zheng, X. Zhang, and 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, August. 2020.
- [35] B. H. Nguyen, B. Xue, and M. Zhang, "A constrained competitive swarm optimizer with an svm-based surrogate model for feature selection," *IEEE Transactions on Evolutionary Computation*, vol. 28, no. 1, pp. 2–16, 2024.
- [36] L. Li, Y. Li, Q. Lin, S. Liu, J. Zhou, Z. Ming, and C. A. Coello Coello, "Neural net-enhanced competitive swarm optimizer for large-scale multiobjective optimization," *IEEE Transactions on Cybernetics*, vol. 54, no. 6, pp. 3502–3515, 2024.
- [37] R. Cheng, Y. Jin, K. Narukawa, and B. Sendhoff, "A multiobjective evolutionary algorithm using gaussian process-based inverse modeling," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 6, pp. 838–856, December. 2015.
- [38] X. Yang, J. Zou, S. Yang, J. Zheng, and Y. Liu, "A fuzzy decision variables framework for large-scale multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 27, no. 3, pp. 445–459, June. 2023.
- [39] X. Zhang, Y. Tian, R. Cheng, and Y. Jin, "A decision variable clustering-based evolutionary algorithm for large-scale many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 97–112, February. 2018.
- [40] X. Wang, K. Zhang, J. Wang, and Y. Jin, "An enhanced competitive swarm optimizer with strongly convex sparse operator for large-scale multi-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 5, pp. 859–871, October. 2022.
- [41] R. Cheng and Y. Jin, "A competitive swarm optimizer for large scale optimization," *IEEE Transactions on Cybernetics*, vol. 45, no. 2, pp. 191–204, February. 2015.
- [42] Q. Yang, W.-N. Chen, J. D. Deng, Y. Li, T. Gu, and J. Zhang, "A level-based learning swarm optimizer for large-scale optimization," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 4, pp. 578–594, August. 2018.
- [43] R. Lan, Y. Zhu, H. Lu, Z. Liu, and X. Luo, "A two-phase learning-based swarm optimizer for large-scale optimization," *IEEE Transactions on Cybernetics*, vol. 51, no. 12, pp. 6284–6293, December. 2021.
- [44] X. Gao, S. Song, H. Zhang, and Z. Wang, "A flexible ranking-based competitive swarm optimizer for large-scale continuous multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 29, no. 1, pp. 247–261, 2025.
- [45] Y. Tian, X. Zhang, C. Wang, and Y. Jin, "An evolutionary algorithm for large-scale sparse multiobjective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 2, pp. 380–393, April. 2020.
- [46] Y. Tian, L. Si, X. Zhang, R. Cheng, C. He, K. C. Tan, and Y. Jin, "Evolutionary large-scale multi-objective optimization: A survey," *ACM Computing Surveys (CSUR)*, vol. 54, no. 8, pp. 1–34, 2021.
- [47] Z. Wu, W. Lin, H. Tang, and Q. Lin, "A multiobjective multitask evolutionary algorithm based on decomposition and multivariate gaussian distribution," in *2021 IEEE 7th International Conference on Cloud Computing and Intelligent Systems (CCIS)*, 2021, pp. 104–108.
- [48] Q. Lin, Y. Ye, L. Ma, M. Jiang, and K. C. Tan, "Dynamic multiobjective evolutionary optimization via knowledge transfer and maintenance," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 54, no. 2, pp. 936–949, 2024.
- [49] Q. Lin, Q. Wang, B. Chen, Y. Ye, L. Ma, and K. C. Tan, "Multiobjective many-tasking evolutionary optimization using diversified gaussian-based knowledge transfer," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2024.
- [50] J. Zhang, W. Zhou, X. Chen, W. Yao, and L. Cao, "Multisource selective transfer framework in multiobjective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 3, pp. 424–438, 2020.
- [51] N. Khodadadi, E. Khodadadi, B. Abdollahzadeh, E.-S. M. El-Kenawy, P. Mardanpour, W. Zhao, F. S. Gharehchopogh, and S. Mirjalili, "Multi-objective generalized normal distribution optimization: A novel algorithm for multi-objective problems," *Cluster Computing*, vol. 27, no. 8, pp. 10589–10631, 2024.
- [52] A. Song, Q. Yang, W.-N. Chen, and J. Zhang, "A random-based dynamic grouping strategy for large scale multi-objective optimization," in *2016 IEEE Congress on Evolutionary Computation (CEC)*, 2016, pp. 468–475.
- [53] S. Liu, Q. Lin, K.-C. Wong, Q. Li, and K. C. Tan, "Evolutionary large-scale multiobjective optimization: Benchmarks and algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 27, no. 3, pp. 401–415, June. 2023.
- [54] 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.
- [55] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, December. 2007.
- [56] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 577–601, 2014.
- [57] K. Deb and M. Goyal, "A combined genetic adaptive search (GeneAS) for engineering design," *Computer Science and informatics*, vol. 26, pp. 30–45, 1996.
- [58] W. Chen, H. Ishibuchi, and K. Shang, "Fast greedy subset selection from large candidate solution sets in evolutionary multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 4, pp. 750–764, August. 2022.
- [59] K. Shang and H. Ishibuchi, "A new hypervolume-based evolutionary algorithm for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 5, pp. 839–852, October. 2020.
- [60] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "PlatEMO: A MATLAB platform for evolutionary multi-objective optimization [educational forum]," *IEEE Computational Intelligence Magazine*, vol. 12, no. 4, pp. 73–87, November. 2017.
- [61] B. Li, Y. Yang, P. Yang, G. Li, K. Tang, and A. Zhou, "Causal inference-based large-scale multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 29, no. 2, pp. 444–458, 2025.
- [62] Alcalá-FdezJ., SánchezL., GarcíaS., J. J. Del, VenturaS., M. Garrellj., OteroJ., RomeroC., BacarditJ., and M. Rivasv., "KEEL: a software tool to assess evolutionary algorithms for data mining problems," *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, vol. 13, no. 3, pp. 307–318, May. 2008.
- [63] B. Li, Y. Zhang, P. Yang, X. Yao, and A. Zhou, "A two-population algorithm for large-scale multiobjective optimization based on fitness-aware operator and adaptive environmental selection," *IEEE Transactions on Evolutionary Computation*, vol. 29, no. 3, pp. 631–645, 2025.
- [64] S. Liu, J. Li, Q. Lin, Y. Tian, J. Li, and K. C. Tan, "Evolutionary large-scale multiobjective optimization via autoencoder-based problem transformation," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 8, no. 4, pp. 2709–2722, 2024.
- [65] N. Verbiest, J. Derrac, C. Cornelis, S. García, and F. Herrera, "Evolutionary wrapper approaches for training set selection as preprocessing mechanism for support vector machines: Experimental evaluation and support vector analysis," *Applied Soft Computing*, vol. 38, pp. 10–22, 2016.
- [66] A. Asuncion and D. Newman, "Uci machine learning repository," 2007.