Coevolutionary Multitasking for Constrained Multiobjective Optimization

Songbai Liu, Zeyi Wang, Qiuzhen Lin, *Member, IEEE,* Jianyong Chen, and Kay Chen Tan, *Fellow, IEEE*

***Abstract*—Exploring the complex landscape of constrained multiobjective optimization problems (CMOPs) using evolutionary algorithms presents challenges of harmonizing constraints with optimization objectives. Coevolutionary multitasking (CEMT) emerges as a promising strategy to grapple with these complexities, capitalizing on the potential synergy achievable from distinct yet complementary tasks. Continuing in this research direction, this paper proposes an adaptive CEMT framework, referred to as ACEMT. The goal of ACEMT is to solve CMOPs effeciently by facilitating knowledge sharing from two adaptive auxiliary tasks that exhibit complementarity to the main task (i.e., the target CMOP). One auxiliary task adapts to the evolving landscape by gradually narrowing its constraint boundaries, thus effectively exploring regions with smaller feasible spaces. The second auxiliary task is intentionally structured to focus exclusively on specific constraints, undergoing continuous adaptation throughout the evolutionary process to expedite convergence and uncover potential regions. Moreover, to boost their efficiency, an adaptive constraint relaxation and an adaptive constraint selection strategy are customized for these two auxiliary tasks, respectively. To validate the ACEMT's performance, a comprehensive series of experiments is conducted, encompassing three benchmark suites and real-world applications. The experimental findings confirm the ACEMT's superiority, as it consistently outperforms or rivals other state-of-the-art related constrained evolutionary algorithms.**

***Index Terms*—Constrained Multiobjective Optimization, Coevolutionary Multitasking, Adaptive Auxiliary Tasks.**

I. Introduction

MULTIobjective optimization poses a profound challenge spanning diverse fields, including engineering, economics, and complex decision-making systems. This intricate task involves simultaneously optimizing multiple conflicting objectives, which adds complexity compared to traditional single-objective optimization. When constraints are incorporated, as is often the case in real-world scenarios, the complexity deepens. Constraints, such as resource limits and safety requirements [1, 2], necessitate the discovery of feasible Pareto-optimal solutions, which optimize multiple objectives while adhering to these constraints. This is of paramount importance in practical applications like robot gripper optimization [3], urban bus scheduling [4], and energy saving optimization [5].

Evolutionary algorithms (EAs), drawing inspiration from the principles of natural selection, have demonstrated their effectiveness in solving multiobjective optimization problems (MOPs). Nevertheless, traditional multiobjective EAs (MOEAs) often neglect the consideration of constraint handling, leading to suboptimal efficiency when tackling constrained MOPs (CMOPs) [6]. In this context, constrained MOEAs (CMOEAs) play a crucial role in developing specific techniques for handling constraints. These techniques are designed to significantly enhance the performance when solving CMOPs. Essentially, CMOEAs are charged with the task of customizing constraint-handling techniques (CHTs) tailored to the specific challenges presented by various CMOPs. Four main categories of CHTs find common usage, encompassing methods based on constraint violation penalty (CVP) [7–10], constrained dominance principle (CDP) [11–13], constraint-objective transformation (COT) [14–16], and constraint-oriented multi-stage (CMS) [17–20]. For an in-depth understanding of these CHTs, please refer to Section II.B, where a comprehensive explanation is provided.

Although the above existing CHTs have greatly improved the performance of CMOEAs, they do have their limitations, mainly centered around finding the right equilibrium between convergence, diversity, and feasibility [21]. In CVP-based CMOEAs, it is challenging to determine the appropriate penalty term and to seamlessly integrate constraints into the objective functions. For CDP-based CMOEAs, they often prioritize solution feasibility during non-dominated sorting or performance comparison [22]. This focus on feasibility can inadvertently diminish the contribution of infeasible solutions, leading to reduced population diversity and a higher risk of getting stuck in local optima. Conversely, COT-based CMOEAs treat constraints as optimization objectives, potentially making the problem more complex by turning it into a complex many-objective optimization task, especially when there are a lot of constraints [23]. CMS-based CMOEAs divide the evolutionary process into multiple stages, each emphasizing different priorities, such as population convergence, diversity, or feasibility [24, 25]. However, this kind of CHTs introduces new challenges, including properly allocating computational resources among these stages. Recognizing these limitations, it is essential to continuously explore refinements in CHTs to attain a trade-off in improving the convergence, diversity, and feasibility of candidate solutions.

In recent years, there has been a growing interest in the concept of employing multiple populations within a coevo-

Manuscript received Nov xx, 2023. This work is partially supported by the National Natural Science Foundation of China (NSFC) under No. U21A20512, and in part by the Research Grants Council of the Hong Kong SAR under Grant PolyU11211521 and Grant PolyU15218622, and in part by Natural Science Foundation of Guangdong Province under Grant No. 2023A1515011238.

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

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

lutionary multitasking (CEMT) framework to solve CMOPs [26]. In CEMT, each population is designated for a specific optimization task and works collaboratively to improve the quality of their respective populations through mutual cooperation. Each population can select the appropriate CHT based on its corresponding optimization task. Here, the original CMOP is treated as the main task. Complementary auxiliary tasks are then created to work alongside the main task, enabling the different populations to collectively address the challenges of achieving a harmonious balance between convergence, diversity, and feasibility. CEMT does come with its own set of challenges, such as devising appropriate auxiliary tasks, promoting knowledge exchange between various tasks, and ensuring diversity across multiple populations. Effectively addressing these challenges is pivotal for the advancement of CEMT-based CMOEAs.

Building on the insights gained from the above observations, this paper centers its efforts on designing an adaptive CEMT (ACEMT) framework, aiming to craft adaptive auxiliary tasks for CMOPs. Specifically, the main task is dedicated to the pursuit of optimal feasible solutions (i.e., focus on feasibility), while the two auxiliary tasks concentrate on enhancing diversity and promoting convergence, respectively. The main contributions of this paper can be summarized as follows:

1) We present the ACEMT framework, featuring three interconnected optimization tasks, each with its personalized population and distinct goals. Knowledge sharing between these tasks maintains a balanced interplay of diversity, convergence, and solution feasibility.
2) We design an auxiliary task focusing on adaptive constraint relaxation to preserve diversity. This task fine-tunes relaxation levels. It's complemented by an adaptive angle-based selection strategy, in which the solutions are normalized based on their distribution.
3) We create an auxiliary task for adaptive constraint selection, consciously omitting specific constraints or considering subsets. The goal is to prioritize the exploration of solutions with improved convergence.

The rest of this paper is arranged as follows. The related work on evolutionary multiobjective optimization and the motivation of this work are provided in Section II. A detailed description of ACEMT is presented in Section III. The experimental studies of ACEMT are given in Section IV. The conclusion of this work and the potential following future work are provided in Section V.

II. Related Work and Motivation

A. Constrained Multiobjective Optimization

In general, a CMOP can be defined as follows:

$$\begin{aligned} &\text{Minimize } \ F(x) = (f_1(x), ..., f_m(x)) \\ &s.t. \begin{cases} g_j(x) \leq 0, \ j = 1, ..., q \\ h_j(x) = 0, \ j = q+1, ..., l \end{cases} \end{aligned} \tag{1}$$

where $x = (x_1, x_2, ..., x_n)$ is a solution vector with n variables, $F(x)$ represents the set of m objective functions to be optimized. $g(x)$ and $h(x)$ are the q inequality constraints and $l - q$ equality constraints, respectively. Given two solutions $(x, y) \in \Omega$, if $f_i(x) \leq f_i(y)$ for $\forall i \in \{1, 2, ..., m\}$ and $\exists j \in \{1, 2, ..., m\} \Rightarrow f(x_j) < f(y_j)$, x is said to Pareto dominate y, expressed as $x \prec y$. A solution earns the label of "Pareto optimal" when there exists no other solution capable of dominating it. These Pareto optimal solutions collectively constitute what is termed the Pareto set (PS). This set is then projected onto the objective space, forming the Pareto front (PF). For clarity, this paper distinguishes between two variations of the PF: the unconstrained PF (UPF) represents the PF for the optimization task without any constraints considered, while the constrained PF (CPF) represents the PF for the target CMOP that encompasses all constraints.

The feasibility of a solution x can be assessed by its constraint violation (CV) degree $CV(x)$, defined as follows:

$$CV(x) = \sum_{j=1}^{l} cv_j(x) \tag{2}$$

where $cv_j(x)$ indicates the degree to which x violates the jth constraint, which can be calculated as follows:

$$cv_j(x) = \begin{cases} max(0, \ g_j(x)), & \text{if } j \leq q \\ max(0, \ |h_j(x) - \delta|), & \text{otherwise} \end{cases} \tag{3}$$

where δ is a boundary relaxation parameter that converts equality constraints into inequality constraints. In general, δ is a very small positive value, such as 10^{-4}. Solving CMOPs is a challenging task, demanding a delicate balance among three fundamental aspects: ensuring solutions are feasible, maintaining a variety of solutions, and achieving convergence. These aspects are essential in the pursuit of approximating the UPF using the obtained solutions.

B. Constraint-Handling Techniques

Most CMOEAs primarily focus on designing effective CHTs for solving CMOPs. These CHTs fall into four common categories: CVP-based, CDP-based, COT-based, and CMS-based CHTs, each of which is detailed below.

CVP-based CHTs incorporate $CV(x)$ into the objective function as a penalty term. This transformation aims to convert CMOPs into equivalent unconstrained MOPs. The critical aspect of this approach lies in determining the appropriate penalty coefficient, which significantly influences the algorithm's efficiency. Selecting the appropriate penalty coefficient is of paramount importance. If it's too small, it may fail to exert enough constraint pressure, causing the population to drift into the infeasible region. On the other hand, if the penalty coefficient is too large, it might trap the population in the local feasible region, preventing it from reaching the global optimum. Representative CVP-based CMOEAs include c-DPEA [7], ShiP [8], CRSDE [9], and TPDE [10].

CDP-based CHTs commonly prioritize solution feasibility during performance evaluation, and they are widely adopted because of their straightforward implementation, as in [11]. The criteria used for comparison in CDP are as follows:

$$x \prec_{CDP} y, if \begin{cases} CV(x) < CV(y) or \\ CV(x) = CV(y) \& x \prec y \end{cases} \tag{4}$$

where $x \prec_{CDP} y$ signifies that, according to CDP criteria, the performance of x surpasses that of y. Feasible solutions have a definite advantage over infeasible ones. Additionally, when evaluating two infeasible solutions, the solution with a lower level of CV is considered superior to the one with a higher CV. As a consequence, CDP prioritizes individuals based on their adherence to constraints, potentially leading to populations becoming concentrated in specific, locally feasible regions. To mitigate this drawback, the ε-level CDP is developed in [12], in which a parameter called ε is defined to relax the constraint conditions. When the $CV(x)$ of a solution x is less than ε, it is considered feasible, even if it's not precisely equal to 0. When ε is gradually reduced to 0, the ε-level CDP converges to the standard CDP. Hence, the tuning of ε holds paramount significance. If the initial value of ε is excessively large, it could lead to an extensive feasible region, potentially confining the population to local areas. Conversely, a rapid reduction in ε might abruptly render solutions near the feasible region as infeasible, resulting in the loss of potentially valuable solutions. Representative CDP-based CMOEAs include CRSDE [9], MFO [27], and MTCMO [28].

COT-based CHTs address constraints by treating them as supplementary objectives, transforming CMOPs into MOPs. This allows existing MOEAs to be used for problem solving. However, since the exact number of constraints is often uncertain, converting each constraint into an objective may lead to a many-objective optimization problem, increasing the problem's complexity. Therefore, the more common practice is to adopt the minimization of $CV(x)$ as an optimization objective. Representative CDP-based CMOEAs include ToR [14], NRC [15], and CMOEA-MS [16].

CMS-based CHTs partition the optimization process into distinct stages, each aimed at achieving specific goals. One of the most classical representatives is the push-pull search (PPS) framework [17]. During the "push" phase, no constraints are taken into account, and the goal is to propel the population toward the UPF. In contrast, the "pull" phase considers all constraints, with the aim of guiding the population back to the CPF. Notable advancements in PPS algorithms encompass CMOES [18], URCMO [19], and MSCMO [20].

C. Coevolutionary Multitasking with Multi-Population

CEMT-based CMOEAs are designed to co-evolve distinct populations or archives, each with specific roles aimed at enhancing efficiency and optimization performance through collaborative interactions. There are two primary design approaches. The first approach, rooted in evolutionary multitasking (EMT), involves generating a range of optimization tasks, encompassing both constrained and unconstrained variations derived from the original CMOP. Each of these tasks is managed by dedicated populations that produce offspring solutions and share them with each other. The second approach is archive-centered, where auxiliary archives are tasked with the storage of solutions aligned with various preferences, including diversity-based and convergence-based archives. The primary evolutionary population holds the responsibility for generating offspring solutions and collaborates with the archives. These coevolutionary multitasking strategies are thoughtfully designed to collectively enhance exploration and optimization capabilities within CMOEAs.

Various CEMT frameworks have been proposed to boost optimization performance in solving CMOPs. BiCo [29] uses an archive that collaborates with the main population in generating offspring. By incorporating $CV(x)$ as an extra objective, it excels at preserving highly convergent infeasible solutions. Angular distance is used to select solutions with outstanding diversity. TPEA [30] uses two archives, integrating $CV(x)$ to create $m + 1$ optimization objectives, resulting in a non-dominated solution set. Subsequently, unconstrained non-dominated sorting was applied to this set, with one archive dedicated to convergent solutions and the other preserving reverse non-dominated solutions. IDW [31] applied a transition phase to categorize the population into diversity and convergence archives. The former selects solutions using decomposition, while the latter filters solutions based on the CDP. In addition, the EMT-based frameworks, like CMOEMT [32], incorporate three tasks involving the target CMOP, a relaxed CMOP, and an unconstrained MOP. MCCMO [33] assign a task for each constraint, implementing a sleep/active state transition strategy for each. In contrast, CMOEAPP [34], DDCMOEA [35], and EMCMO [36] featured two tasks: one emphasizing convergence over feasibility, and the other giving priority to feasibility. Lastly, CCMO [37] and MTCMO [28] each included the target CMOP and a relaxed CMOP tasks.

D. Motivation

CMOEAs aim to identify a feasible population that closely approximate the true CPF. However, this endeavor is rife with difficulties. Constraints often render a significant portion of the search space as infeasible regions, significantly complicating the search for feasible solutions. Constraints can create extensive and continuous infeasible regions, further heightening the challenge of discovering the global optimum. Constraints can partition the search space into numerous small feasible regions, exacerbating the difficulty of identifying optimal solutions for each one. Essentially, the solving of CMOPs demand the simultaneous consideration of feasibility, diversity, and convergence of the evolutionary population, necessitating a delicate balance among these aspects. However, many CMOEAs struggle to effectively strike this balance.

Achieving this balance by a single population poses a complex challenge, driving the exploration of CMOEAs based on coevolutionary multitasking. CEMT leverages multiple populations to store solutions with distinct qualities, fostering a harmonious interplay between feasibility, diversity, and convergence. As populations evolve, the transfer of knowledge between them plays a pivotal role in maintaining this equilibrium. Updating multiple populations is crucial, requiring specific goal-aligned environment selection strategies. The creation of auxiliary tasks is equally vital. Unconstrained auxiliary tasks are beneficial for navigating extensive infeasible regions and enhancing convergence. However, in cases where CPF and UPF are far apart, unconstrained auxiliary tasks may fall short [36]. Relaxed CMOP tasks broaden the feasible region

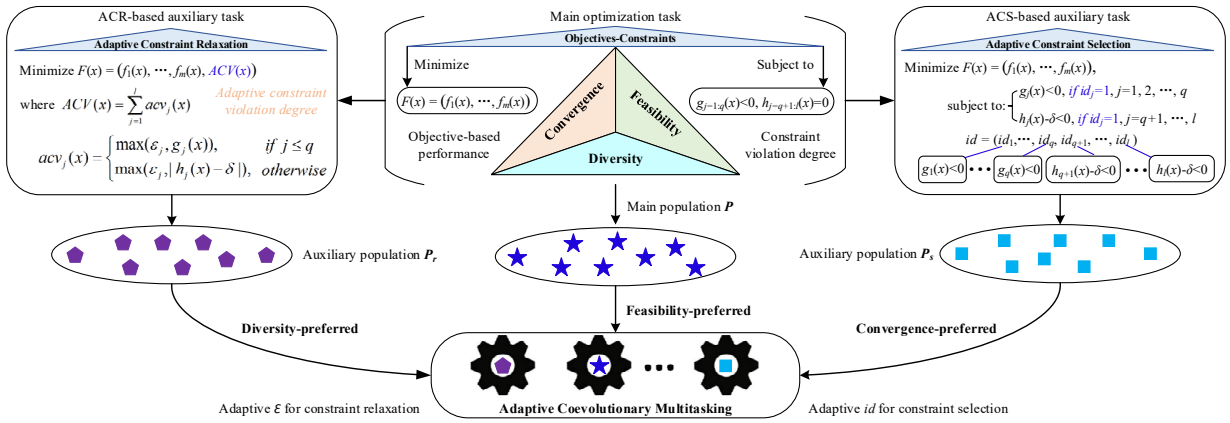

Fig. 1. An intuitive overview of ACEMT, mainly including the specific composition of the three optimization tasks.

by adjusting constraint relaxation through the parameter ε. Managing the update of ε is a delicate balance: a slow change rate may delay convergence to the true feasible region, while rapid changes risk discarding promising solutions [28]. Single-constraint-based tasks offer the potential to discover superior solutions but may demand more computational resources [33].

As previously discussed, current methods for constructing auxiliary tasks struggle to effectively address a wide range of constraints. In response to this challenge, this paper develops an adaptable CEMT framework. ACEMT aims to strike a balance between convergence and diversity by implementing two complementary adaptive auxiliary tasks, while the main population focuses on a task that prioritizes feasibility. In the following section, we provide a detailed explanation of the ACEMT framework. Subsequently, we delve into the specifics of the populations assigned to the two auxiliary tasks, presenting them in a coherent sequence.

III. The Proposed Algorithm

This section provides a detailed explanation of the proposed ACEMT for solving CMOPs. We begin by providing an intuitive overview of ACEMT in Section III.A. Following that, in Section III.B, we present the general framework of ACEMT. Subsequently, we dive into the primary components of ACEMT in the next two subsections, concentrating on two optimization tasks rooted in adaptive constraint relaxation (ACR) and adaptive constraint selection (ACS), respectively.

A. An Intuitive Overview of ACEMT

As depicted in Fig.1, ACEMT concurrently handles three distinct optimization tasks. Alongside the main task defined in (1), it incorporates two adaptive auxiliary tasks. The ACR-based task is formulated as an MOP with $(m+1)$ objectives, defined as follows:

$$\text{Minimize}\ F(x)=(f_1(x),\cdots,f_m(x),ACV(x)) \tag{5}$$

where the newly joined objective $ACV(x)$ signifies the degree of adaptive constraint violation (ACV) for solution x, which is defined as follows:

$$ACV(x)=\sum_{j=1}^{l} acv_j(x) \tag{6}$$

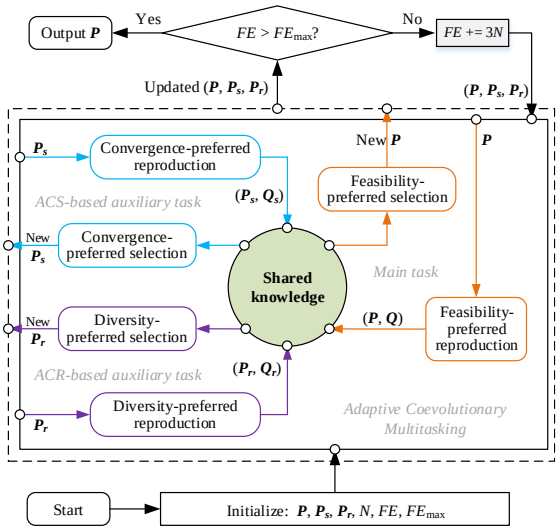

Fig. 2. Illustration of the general framework of ACEMT.

where $acv_j(x)$ quantifies the adaptive violation level of the jth constraint, computed as follows:

$$acv_j(x)=\begin{cases}\max\left(\varepsilon_j,g_j(x)\right), & \text{if } j\le q\\ \max\left(\varepsilon_j,|h_j(x)-\delta|\right), & \text{otherwise}\end{cases} \tag{7}$$

where ε_j represents the constraint relaxation level, and its value dynamically adapts as evolution proceeds. The ACS-based task dynamically selects which constraints to prioritize for satisfaction, as defined below:

$$\begin{aligned}&\text{Minimize}\ \ F(x)=(f_1(x),f_2(x),...,f_m(x))\\ &s.t.\begin{cases}id_j*g_j(x)\le 0,\ \ j=1,...,q\\ id_j*(h_j(x)-\delta)\le 0,\ \ j=q+1,...,l\end{cases}\end{aligned} \tag{8}$$

where $id=(id_1,id_2,\cdots,id_l)$ is a binary vector. A value of 0 for id_j denotes that the corresponding constraint with index j is not considered within this task.

Upon identifying the optimization tasks, ACEMT assigns a separate population to each task, each of which has its distinct preferences and personalized population evolves with the aim of enhancing a negotiated performance indicator. To be

specific, the population P dedicated to the main task prioritizes solutions with superior feasibility. The population P_r for the ACR-based task adaptively relaxes constraints to maintain a more diverse set of solutions. Meanwhile, the population P_s for the ACS-based task either disregards constraints entirely or selectively incorporates specific constraints to preserve solutions that demonstrate improved convergence in terms of objective-based performance. Thus, three task-specific populations collaborate by sharing knowledge during the subsequent evolutionary process, i.e., coevolutionary multitasking.

B. The General Framework of ACEMT

ACEMT is designed to strike a harmonious balance between solution convergence, diversity, and feasibility when tackling a given CMOP. ACEMT, resembling the standard MOEA structure, encompasses three main components: initialization, reproduction, and environment selection, as shown in Fig. 2. In the initialization, three separate populations (P, P_r, P_s) are randomly created, with each commencing with N initial solutions. This process consumes $3N$ function evaluations, and the function evaluation counter FE starts from $FE = 3N$. With each subsequent evolutionary generation, an additional $3N$ function evaluations are expended. The evolutionary process continues until it reaches the predefined maximum number of function evaluations $FE_{\max}$. In each generation, ACEMT employs three parallel pipelines to execute the processes of reproduction and environmental selection for the main task and the two auxiliary tasks, respectively.

In the reproduction, ACEMT applies a uniform evolutionary search on all tasks to generate offspring populations (i.e., Q, Q_r, Q_s). Specifically, ACEMT leverages a hybrid search strategy that integrates simulated binary crossover (SBX) and differential evolution (DE), with SBX being responsible for creating half of the offspring solutions, and DE handling the remaining half (the DE/best/1 operator is used). This is inspired by prior research indicating that different evolutionary searches exhibit varying effectiveness in addressing different CMOPs. For instance, SBX tends to excel in solving DASCMOPs [38], while DE is better suited for tackling LIRCMOPs [39]. As such, the hybrid strategy provides enhanced generalization capabilities. In particular, task-specific reproduction is primarily evident in their mating selection processes, encompassing the determination of mating pools using tournaments in SBX and the selection of parents with superior performance in DE. Each task conducts mating selection based on its preferred performance indicators. The criteria for mating selection depend on the adopted environmental selection strategy. The main task in ACEMT employs the widely-used feasibility-preferred selection method. At first, all solutions undergo CDP-based sorting, with solutions holding higher CDP rankings given precedence. In situations where solutions share the same ranking, their selection is further determined by their respective crowding distances, following the same process as in NSAG-II [11]. The identical selection criterion is used for mating selection. Moreover, ACEMT's two auxiliary tasks utilize diversity-preferred and convergence-preferred selection strategies, which will be elaborated in Section III.C and Section III.D, respectively.

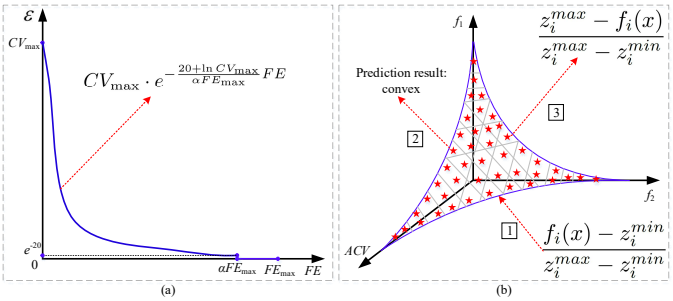

Fig. 3. Illustration of the change curve of ε and the adaptive normalization for the ACR-based auxiliary task.

Moreover, a distinctive aspect of ACEMT is its knowledge sharing. Before environment selection, all solutions from the three tasks (including P, Q, P_r, Q_r, P_s, Q_s) are shared to form an union population P_u without repeated solutions. In prior research [19, 36], the typical practice is to investigate the relationship between CPF and UPF to gauge the appropriateness of knowledge sharing. For example, when a distinct separation between CPF and UPF is detected, the transfer of solutions from P_s to P may not enhance the performance of P and could potentially be counterproductive. However, accurately assessing the CPF-UPF relationship poses a challenge, prompting us to avoid making such determinations. Instead, we opt for the direct combination of all parents and offspring. This unreserved knowledge sharing among tasks allows the environmental selection for each task to adapt to varying CPF-UPF relationships and select suitable solutions accordingly.

C. Adaptive Constraint Relaxation

The key to the ACR-based auxiliary task lies in the adaptive adjustment of the parameter ε in (7). It's important to consider that the improvement in population quality during the early stages of evolution is typically more pronounced, indicating a rapid convergence of the population. Conversely, as the evolution progresses, population quality improvement tends to plateau. Therefore, it is essential to rapidly decrease the value of ε in the early stages while allowing it to decrease more gradually in the later stages to facilitate the exploration of potential feasible regions. In ACEMT, the ε is dynamically reduced from an initially large value to zero during the course of evolution. To achieve this, an exponential decay function is employed to model the change in ε, defined as follows:

$$\varepsilon = \begin{cases} CV_{\max} \cdot e^{-\frac{20+\ln CV_{\max}}{\alpha FE_{\max}} FE} & \text{if } FE \le \alpha FE_{\max} \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

where α is a hyperparameter with values between 0 and 1. A sensitivity analysis on the α is conducted in our experimental studies. Besides, $CV_{\max}$ indicates the initial value of ε, which is defined as the maximum CV degree among the initial solutions at the first generation, computed as follows:

$$CV_{\max} = \max\{CV(x) \mid x \in P \cup P_r \cup P_s\} \tag{10}$$

Fig. 3(a) shows the change curve of ε to enhance clarity. Dynamic constraint relaxation, governed by the adaptive ad-

Algorithm 1 AdaptiveAngleBasedDeletion

Input: union population P_u, population size N, current ε
Output: the updated P_r for the ACR-based task

initialize i = 1, $P_u^t = P_r = \emptyset$;
calculate $ACV(x)$ by (6) for each $x \in P_u$;
divide P_u into $(S_1, S_2, ..., S_L)$ by non-dominated sorting;
while $|P_u^t| < N$ **do**
    P_u^t = $P_u^t \cup S_i$, i = i + 1;
end while
normalize all solutions in P_u^t by (11);
predict the convexity of the shape of P_u^t;
if P_u^t is convex, re-normalize solutions in P_u^t by (12);
calculate the angle between solution-pairs in P_u^t by (13);
while $|P_u^t| > N$ **do**
    find the solution pair (x, y) with the minimum angle;
    delete the solution with the worse ACV from P_u^t;
end while
add all solutions of P_u^t into P_r;
return P_r

justment of ε, is vital for maintaining diversity in P_r and safeguarding valuable infeasible solutions. By integrating the optimization objective $ACV(x)$ into the ACR-based task, we fine-tune constraint relaxation as ε changes. This empowers P_r to explore a broader solution space while retaining infeasible solutions that might become feasible with further adaptations, thus promoting diversity and expanding exploration. Treating $ACV(x)$ as an optimization objective enhances this strategy, motivating P_r to recognize the potential of infeasible solutions and actively seek improvements. The rationale stems from the recognition that the feasible solution space may be quite limited, making it challenging to directly identify. Promising infeasible solutions located in proximity to the feasible regions can survive by relaxing constraints and guide P_r toward gradual convergence to these feasible regions, thus preserving a diverse set of promising solutions.

To promote and maintain diversity, ACEMT employs an adaptive angle-based deletion (AAD) strategy to update the population P_r by selecting N solutions from the union population P_u. The pseudo-code for implementing AAD is presented in Algorithm 1. Initially, AAD trims P_u to keep only $2N$ solutions. This pruning process involves non-dominated sorting of P_u based on the $(m+1)$ objectives in the ACR-based task, dividing P_u into multiple subsets, and then employing the pruning strategy from NSGA-II to ensure that the number of solutions in the trimmed P_u (referred to as P_u^t) does not exceed N. Subsequently, AAD iteratively removes solutions from P_u^t targeting those with the smallest angles within its respective solution pairs until the total number of solutions in P_u^t is reduced to N. In each iteration, AAD identifies the solution pair within P_u^t with the smallest angle value and eliminates the solution with the less favorable ACV degree, resulting in a more diverse and refined population in P_r.

The crux of AAD lies in the adaptive computation of the angle $\theta(x, y)$ between two solutions (x, y) in P_u^t. This necessitates the normalization of solutions due to differing scales among various objectives. Consequently, AAD normalizes each objective of solution x using the formula below:

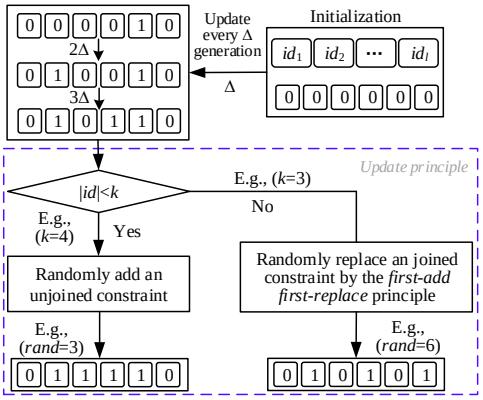

Fig. 4. Illustration of the id update process (strating from a zero vertor) to determine the ACS-based auxiliary task.

$$f_i'(x) = \frac{f_i(x) - z_i^{min}}{z_i^{max} - z_i^{min}} \tag{11}$$

where $i = 1, 2, ..., m+1$ and $f'_{m+1}(x)$ indicates the normilized $ACV(x)$. Besides, z_i^{min} and z_i^{max} denote the minimum and maximum values of the ith objective for all solutions in P_u^t, respectively. Moreover, the distribution of P_u^t also significantly influences the angle between solutions. AAD has the capability to predict whether P_u^t in the current normalized objective space exhibits a concave or convex shape. In particular, if a majority of the non-dominanted solutions in P_u^t are found to be situated below the unit hyperplane, AAD predicts the distribution of P_u^t as convex (please refer to our previous work [40, 41] for a detailed explanation of the prediction). In such case, it proceeds with re-normalization of the solutions in P_u^t as follows:

$$f_i'(x) = \frac{z_i^{max} - f_i(x)}{z_i^{max} - z_i^{min}} \tag{12}$$

Fig. 3(b) shows this adaptive normalization to enhance clarity. This way, the $\theta(x, y)$ can be computed as follows:

$$\theta(x, y) = arccos \frac{\mid F'(x) \cdot F'(y) \mid}{\parallel F'(x) \parallel \cdot \parallel F'(y) \parallel} \tag{13}$$

D. Adaptive Constraint Selection

A pivotal aspect within the ACS-based task involves the thoughtful configuration of the id vector in (8). This entails making strategic decisions on how to adapt the selection of constraints for consideration. The primary goal of the ACS-based task is to guide the population P_r towards more effective convergence across perhaps expansive infeasible regions while simultaneously uncovering promising regions by selectively adjusting the considered constraint set. In several existing work, the id is routinely set to a zero vector, essentially implying that no constraints are factored into this auxiliary task. This common setting rapidly propels the P_r toward the UPF during evolution, providing impetus to the population P for the main task in their pursuit of the UPF. However, once the UPF and CPF are separated far away, this setup can impede P in progressing towards the CPF, especially in the later stages of evolution, leading to detrimental effects.

Algorithm 2 ClusteringBasedSelection

Input: union population P_u, population size N, current id
Output: the updated P_s for the ACS-based task

initialize $i = 1$, $P_u^t = P_r = \emptyset$;
calculate $CV(x)$ based on (2) and (8) for each $x \in P_u$;
divide P_u into $(S_1, S_2, ..., S_L)$ by CDP sorting;
while $|P_u^t| < N$ **do**
    $P_u^t = P_u^t \cup S_i$, $i = i + 1$;
end while
divid P_u^t into N non-empty clusters;
add the fittest solution from each cluster into P_r;
return P_r;

In response to this insight, ACEMT introduces an adaptive mechanism to update the id vector every Δ generations. Initially, id starts as a zero vector. Subsequently, after each interval of Δ generations, one of the zero elements within id is randomly selected and toggled to 1, signifying the activation of the corresponding constraint in the ACS-based task. This process of constraint selection continues until the magnitude of id reaches k (i.e., $|id| = k$), where k serves as a hyper-parameter ranging from 0 to l, and l represents the number of constraints in (1) (typically set to $k = \lfloor 0.5l \rfloor$). Once k constraints are activated in the ACS-based task, id is randomly updated based on the "first-add first-replaced" principle. To be spicific, the constraint that is currently activated first is replaced by a randomly selected, unjoined constraint. A visual representation of the id updates can be found in Fig. 4. A detailed examination of the hyperparameters Δ and k will be conducted in subsequent experiments.

To improve the population quality, ACEMT employs a clustering-based selection strategy for updating the P_s by selecting N solutions from P_u. The procedure for implementing this selection is detailed in Algorithm 2. Initially, P_u is pruned to create P_u^t, which contains no more than N solutions. The pruning strategy applied here is akin to the CDP-based sorting as proposed in NSGA-II. The key distinction is that the sorting here exclusively involves constraints that correspond to elements with a value of 1 in the current id vector. After that, P_u^t is partitioned into N non-empty clusters utilizing a hierarchical clustering method (please consult our prior work [42, 43] for a more in-depth understanding). During the clustering, the similarity between different solutions is evaluated based on their Euclidean distance. Subsequently, within each cluster, the solution with the best fitness is selected, determined by the sum of its objective values. These selected solutions are then added to the population P_r.

IV. Experimental Studies

A. Experiment Settings

In this section, we conduct experiments to validate ACEMT's efficacy in solving diverse CMOPs. Initially, we compare ACEMT with fourteen competitive CMOEAs across three test suites (CF [44], DASCMOP [38], and LIRCMOP [39]). Subsequently, we perform sensitivity analysis on all parameters of ACEMT and assess the effectiveness of its auxiliary tasks through ablation studies. Finally, we evaluate ACEMT's performance using a real-world test suite.

To facilitate a comprehensive evaluation of ACEMT, we chose fourteen competitive CMOEAs, which can be roughly categorized into two groups. The first group comprises CMOEAs following the CEMT framework, including BiCo [29], CMEGL [45], CMOEMT [32], MCCMO [33], MFO-MOEA/D [27], MTCMO [28], and TPEA [30]. The second group encompasses CMOEAs that do not adhere to the CEMT framework, including c-DPEA [7], CMME [46], CMOEA-MS [16], DSPCMDE [47], MSCMO [20], ShiP$^+$ [8], and URCMO [19]. To ensure fair comparisons, we set the parameters of these CEMT-based and Non-CEMT-based solvers according to their original papers. The three parameters in ACEMT are set to $\alpha = 0.5$, $\Delta = 60$, and $k = \lfloor 0.7l \rfloor$.

For a comprehensive assessment, we utilize three widely recognized metrics: inverted generational distance (IGD [48]), IGD plus version (IGD+ [49]), and hypervolume (HV [50]). These metrics jointly measure both convergence and diversity of the obtained population. Please refer to the supplementary file for the HV and IGD+ results due to space limitation. All experiments are carried out on the PlatEMO platform [51]. As recommended by prior work [11, 52], we set the population size (N) to 100, and the maximum number of function evaluations (FE_{max}) is fixed at 200,000. To ensure statistical significance, each algorithm is executed 30 times independently. We employ the Wilcoxon test at a 0.05 significance level to gauge the statistical significance of differences between the two methods. In the results, symbols like "+", "-", or "=" denote whether the compared algorithm performs better, worse, or similarly to ACEMT, respectively.

B. Results on Three Test Benchmark Suites

Table I shows the mean IGD results for ACEMT and its seven CEMT-based counterparts, while Table II exhibits the mean IGD results for ACEMT and seven Non-CEMT-based competitors. In both tables, the best result on each test CMOP is highlighted in bold. In cases where the optimizer fails to discover any feasible solutions, the result is denoted as "NaN".

In the CF test suite,various types of constraints are simulated by confining regions on the hypersphere's surface for the use of positional variables. Comparing ACEMT with seven non-CEMT-based competitors, ACEMT outperforms them by achieving the best IGD results in 7 out of the 10 CF problems. It only falls behind DSPCMDE in CF3 and ShiP$^+$ in CF8. When assessed against seven CEMT-based competitors, ACEMT exhibits its strength, securing the best IGD results across all CF problems, except for CF1, where its performance only falls behind CMOEMT. The efficacy of ACEMT is evident in Figs. 1-10 of the supplementary file, showcasing its robust convergence and diversity exploration capabilities across the CF test suite. The adaptive auxiliary tasks play a pivotal role in addressing convergence, feasibility, and diversity needs, contributing to ACEMT's strong performance.

In DASCMOP, the first constraint controls feasibility complexity by managing the feasible region's size, the second enhances diversity challenges by regulating the region's width,

TABLE I
THE AVERAGE IGD RESULTS OBTAINED BY ACEMT AND ITS SEVEN CEMT-BASED COMPETITORS ON THREE TEST BENCHMARK SUITES

Problem	M	D	BiCo	CMEGL	CMOEMT	MCCMO	MFO-MOEA/D	MTCMO	TPEA	ACEMT
CF1	2	10	9.0423e-3 (1.24e-3) -	2.6842e-3 (2.47e-4) -	**8.2219e-4 (1.90e-4) +**	4.5769e-3 (4.81e-4) -	3.0131e-3 (7.51e-4) -	3.6902e-3 (6.93e-4) -	4.0723e-3 (4.78e-4) -	1.3143e-3 (1.92e-4)
CF2	2	10	3.5017e-2 (1.30e-2) -	2.4868e-2 (9.55e-3) -	3.7367e-3 (1.35e-3) -	2.3980e-2 (8.83e-3) -	1.2991e-1 (5.34e-2) -	2.9725e-2 (1.46e-2) -	1.8692e-2 (8.06e-3) -	**2.6967e-3 (8.37e-5)**
CF3	2	10	2.5890e-1 (1.12e-1) -	2.0577e-1 (8.28e-2) =	2.2197e-1 (9.30e-2) -	2.3406e-1 (6.96e-2) -	5.1104e-1 (2.56e-1) -	2.6523e-1 (8.09e-2) -	2.3148e-1 (9.13e-2) -	**1.6969e-1 (8.70e-2)**
CF4	2	10	9.2684e-2 (4.52e-2) -	9.0853e-2 (3.35e-2) -	2.9876e-2 (9.33e-3) -	7.5593e-2 (2.62e-2) -	2.2497e-1 (1.17e-1) -	9.7794e-2 (3.44e-2) -	6.5929e-2 (2.71e-2) -	**1.7350e-2 (6.72e-3)**
CF5	2	10	2.8331e-1 (1.28e-1) -	2.6464e-1 (9.60e-2) -	1.5754e-1 (7.62e-2) -	2.5989e-1 (1.16e-1) -	4.2346e-1 (9.42e-2) -	3.1854e-1 (1.41e-1) -	2.2404e-1 (8.38e-2) -	**9.1006e-2 (3.03e-2)**
CF6	2	10	8.0135e-2 (3.82e-2) -	4.7259e-2 (1.78e-2) -	2.5032e-2 (7.61e-3) -	5.3763e-2 (2.18e-2) -	2.1103e-1 (7.66e-2) -	6.0705e-2 (2.58e-2) -	3.1146e-2 (1.05e-2) -	**1.3571e-2 (2.02e-3)**
CF7	2	10	2.5300e-1 (9.44e-2) -	2.6514e-1 (1.16e-1) -	2.2456e-1 (8.99e-2) -	2.5307e-1 (1.14e-1) -	4.0920e-1 (1.25e-1) -	2.7036e-1 (1.03e-1) -	1.6415e-1 (7.05e-2) -	**8.4785e-2 (5.77e-2)**
CF8	3	10	1.9854e-1 (9.05e-2) -	1.9929e-1 (7.05e-2) -	3.1173e-1 (5.84e-2) -	1.8012e-1 (5.06e-2) -	2.2697e-1 (2.12e-1) =	2.3020e-1 (1.17e-1) -	1.7958e-1 (3.04e-2) -	**1.4527e-1 (3.14e-2)**
CF9	3	10	1.0145e-1 (2.86e-2) -	9.6110e-2 (5.30e-3) -	1.8662e-1 (5.18e-2) -	9.2043e-2 (4.79e-3) -	1.3922e-1 (1.53e-1) -	1.0026e-1 (3.43e-2) -	9.8522e-2 (8.01e-3) -	**7.2146e-2 (9.87e-3)**
CF10	3	10	4.4565e-1 (2.48e-1) =	3.4410e-1 (6.22e-2) -	4.4758e-1 (1.73e-1) -	3.4787e-1 (8.13e-2) -	4.9938e-1 (1.99e-1) -	3.8698e-1 (1.00e-1) -	2.6225e-1 (1.22e-1) =	**2.2508e-1 (8.96e-2)**
DASCMOP1	2	30	7.1117e-1 (4.75e-2) -	6.3327e-1 (1.13e-1) -	4.3851e-1 (3.27e-1) -	2.9966e-3 (2.37e-4) =	7.3925e-1 (5.26e-2) -	6.7811e-1 (2.95e-2) -	6.7281e-1 (5.36e-2) -	**2.9136e-3 (2.60e-4)**
DASCMOP2	2	30	2.3183e-1 (1.55e-2) -	2.2034e-1 (3.95e-2) -	1.8338e-1 (7.00e-2) -	4.2292e-3 (1.35e-4) -	5.5572e-1 (2.25e-1) -	2.2414e-1 (2.47e-2) -	2.0557e-1 (2.74e-2) -	**3.9413e-3 (6.58e-5)**
DASCMOP3	2	30	3.0729e-1 (6.26e-2) -	2.6494e-1 (4.16e-2) -	3.3057e-1 (3.08e-2) -	1.9446e-2 (1.94e-4) =	4.8656e-1 (1.76e-1) -	2.6257e-1 (2.34e-2) -	2.2695e-1 (5.87e-2) -	**1.6759e-2 (3.40e-3)**
DASCMOP4	2	30	3.7381e-3 (1.20e-2) -	1.1644e-3 (2.36e-5) +	1.2640e-3 (5.99e-5) -	1.7634e-3 (1.22e-3) -	6.3630e-2 (1.91e-1) -	**1.1589e-3 (1.75e-5) +**	1.4164e-3 (1.15e-4) -	1.2426e-3 (1.34e-4)
DASCMOP5	2	30	1.6090e-2 (7.23e-2) -	2.7068e-3 (4.86e-5) +	2.8127e-3 (8.91e-5) +	4.6061e-2 (1.60e-1) -	1.9024e-2 (1.57e-2) -	**2.7002e-3 (4.76e-5) +**	2.7905e-3 (8.79e-5) +	2.8697e-3 (1.09e-4)
DASCMOP6	2	30	9.4753e-2 (1.32e-1) =	2.5123e-2 (3.06e-2) -	1.8947e-2 (1.75e-3) =	7.2825e-2 (1.73e-1) -	1.0506e-1 (6.91e-2) -	2.3799e-2 (2.51e-2) =	2.6799e-2 (6.80e-2) -	**1.7548e-2 (3.18e-3)**
DASCMOP7	3	30	3.1667e-2 (8.41e-4) -	3.0963e-2 (7.13e-4) -	3.1915e-2 (7.61e-4) -	5.5675e-2 (6.65e-2) -	3.6019e-2 (6.60e-4) -	3.0870e-2 (5.66e-4) -	3.2106e-2 (9.30e-4) -	**3.0332e-2 (8.09e-4)**
DASCMOP8	3	30	4.1232e-2 (8.37e-4) -	4.0512e-2 (1.17e-3) -	4.1334e-2 (9.92e-4) -	1.4747e-1 (1.98e-1) -	6.7543e-1 (3.90e-1) -	3.9631e-2 (1.06e-3) =	4.0110e-2 (1.12e-3) -	**3.9242e-2 (8.30e-4)**
DASCMOP9	3	30	3.3213e-1 (4.79e-2) -	1.0920e-1 (6.04e-2) -	6.8189e-2 (7.78e-2) -	4.0719e-2 (1.29e-3) -	7.0251e-1 (2.98e-1) -	2.6729e-1 (7.21e-2) -	2.7265e-1 (6.30e-2) -	**3.9355e-2 (7.89e-4)**
LIRCMOP1	2	30	1.8786e-1 (1.39e-2) -	9.6607e-2 (2.42e-2) -	2.1606e-1 (5.83e-2) -	8.3560e-2 (5.47e-2) -	2.8665e-1 (3.05e-2) -	1.0502e-1 (2.07e-2) -	1.1922e-1 (1.99e-2) -	**8.2244e-3 (8.14e-4)**
LIRCMOP2	2	30	1.6313e-1 (1.55e-2) -	9.3104e-2 (1.78e-2) -	1.8566e-1 (3.51e-2) -	4.9712e-2 (4.51e-2) -	2.4543e-1 (1.90e-2) -	1.0727e-1 (1.50e-2) -	6.5311e-2 (1.44e-2) -	**7.3765e-3 (7.46e-4)**
LIRCMOP3	2	30	1.8660e-1 (2.33e-2) -	1.6973e-1 (3.53e-2) -	2.5593e-1 (4.83e-2) -	9.2874e-2 (5.84e-2) -	2.9114e-1 (3.61e-2) -	1.2586e-1 (2.70e-2) -	7.0302e-2 (2.43e-2) -	**4.8759e-3 (8.80e-4)**
LIRCMOP4	2	30	1.9175e-1 (2.35e-2) -	1.6909e-1 (3.36e-2) -	2.5018e-1 (4.63e-2) -	1.1555e-1 (5.83e-2) -	2.6053e-1 (3.64e-2) -	1.3714e-1 (2.21e-2) -	8.9001e-2 (1.99e-2) -	**4.6434e-3 (7.92e-4)**
LIRCMOP5	2	30	1.2196e+0 (6.48e-3) -	3.0382e-1 (8.39e-2) -	2.0202e-2 (3.14e-2) -	9.0950e-3 (1.99e-3) -	3.3981e-1 (5.75e-2) -	4.1185e-1 (2.78e-1) -	2.4010e-1 (7.42e-2) -	**6.7303e-3 (5.19e-4)**
LIRCMOP6	2	30	1.3453e+0 (1.51e-4) -	3.6808e-1 (8.87e-2) -	4.9662e-2 (8.17e-2) -	7.5528e-3 (8.24e-4) -	4.0644e-1 (8.84e-2) -	5.7099e-1 (4.37e-1) -	2.5980e-1 (2.16e-1) -	**6.4615e-3 (3.27e-4)**
LIRCMOP7	2	30	3.3616e-1 (5.37e-1) -	1.1859e-1 (3.57e-2) -	5.5679e-2 (4.91e-2) -	1.0359e-2 (1.19e-2) -	1.5307e-1 (2.79e-2) -	1.2320e-1 (3.02e-2) -	9.0236e-2 (2.95e-2) -	**7.1721e-3 (5.58e-4)**
LIRCMOP8	2	30	8.6951e-1 (7.25e-1) -	1.7393e-1 (6.50e-2) -	2.0695e-2 (3.30e-2) -	7.4548e-3 (3.04e-4) -	2.6961e-1 (4.34e-2) -	1.7728e-1 (4.95e-2) -	1.2470e-1 (5.28e-2) -	**7.0641e-3 (2.75e-4)**
LIRCMOP9	2	30	9.1576e-1 (1.50e-1) -	3.4683e-1 (1.06e-1) -	4.3693e-1 (7.05e-2) -	**1.8689e-2 (2.21e-2) +**	1.1684e+0 (2.30e-1) -	6.6272e-1 (1.40e-1) -	3.9999e-1 (4.77e-2) -	6.0417e-2 (3.13e-2)
LIRCMOP10	2	30	9.0233e-1 (4.92e-2) -	1.7952e-1 (4.22e-2) -	2.1373e-1 (7.49e-2) -	7.0878e-3 (6.15e-4) -	5.7217e-1 (1.20e-1) -	2.4436e-1 (1.81e-1) -	1.5165e-1 (3.45e-2) -	**6.1379e-3 (2.99e-4)**
LIRCMOP11	2	30	4.5570e-1 (2.15e-1) -	5.0520e-2 (4.01e-2) -	8.2575e-2 (6.15e-2) -	3.1009e-3 (3.03e-4) -	1.0100e+0 (2.25e-1) -	2.4919e-1 (1.93e-1) -	9.4366e-2 (3.04e-2) -	**2.5686e-3 (2.59e-4)**
LIRCMOP12	2	30	3.8195e-1 (1.85e-1) -	1.2587e-1 (6.78e-2) -	1.9929e-1 (8.19e-2) -	4.7427e-3 (1.01e-3) -	9.6673e-1 (1.12e-1) -	2.9191e-1 (1.13e-1) -	1.7573e-1 (4.30e-2) -	**3.7358e-3 (5.29e-4)**
LIRCMOP13	3	30	1.3194e+0 (2.13e-3) -	**9.2533e-2 (9.30e-4) +**	9.7055e-2 (1.14e-3) +	1.1958e-1 (1.81e-3) -	1.3019e+0 (9.90e-5) -	1.3156e+0 (1.98e-3) -	9.4047e-2 (1.15e-3) +	1.0148e-1 (1.46e-3)
LIRCMOP14	3	30	1.2752e+0 (1.90e-3) -	9.5228e-2 (8.40e-4) +	1.0394e-1 (1.38e-3) +	1.0309e-1 (1.36e-3) +	1.2580e+0 (1.07e-4) -	1.2719e+0 (1.65e-3) -	**9.4483e-2 (9.10e-4) +**	1.7562e-1 (2.96e-1)
+/-/=			0/31/2	4/28/1	4/28/1	2/29/2	0/32/1	2/29/2	3/29/1	

TABLE II
THE AVERAGE IGD RESULTS OBTAINED BY ACEMT AND ITS SEVEN NON-CEMT-BASED COMPETITORS ON THREE TEST BENCHMARK SUITES

Problem	M	D	c-DPEA	CMME	CMOEA-MS	DSPCMDE	MSCMO	ShiP$^+$	URCMO	ACEMT
CF1	2	10	3.7721e-3 (4.30e-4) -	1.0534e-2 (1.96e-3) -	1.4458e-2 (1.16e-2) -	3.3215e-3 (7.60e-4) -	6.8968e-3 (1.14e-3) -	5.3100e-3 (6.39e-4) -	2.3305e-3 (3.10e-4) -	**1.3143e-3 (1.92e-4)**
CF2	2	10	2.5394e-2 (1.30e-2) -	3.8946e-2 (1.83e-2) -	6.1111e-2 (4.15e-2) -	5.8919e-3 (2.24e-3) -	3.0497e-2 (1.59e-2) -	3.4913e-2 (1.06e-2) -	4.8434e-3 (4.92e-4) -	**2.6967e-3 (8.37e-5)**
CF3	2	10	2.5016e-1 (9.57e-2) -	2.9464e-1 (1.11e-1) -	2.5782e-1 (1.01e-1) -	1.0130e-1 (1.64e-2) +	2.2656e-1 (7.86e-2) -	2.5387e-1 (1.48e-1) -	2.0798e-1 (1.09e-1) =	1.6969e-1 (8.70e-2)
CF4	2	10	7.8983e-2 (2.55e-2) -	1.2174e-1 (7.10e-2) -	1.4028e-1 (6.85e-2) -	5.0115e-2 (7.40e-3) -	1.0400e-1 (3.78e-2) -	8.0539e-2 (2.23e-2) -	2.6335e-2 (9.35e-3) -	**1.7350e-2 (6.72e-3)**
CF5	2	10	3.4059e-1 (1.35e-1) -	4.0818e-1 (1.18e-1) -	3.2859e-1 (1.19e-1) -	2.3588e-1 (1.51e-1) -	2.6298e-1 (8.90e-2) -	2.8924e-1 (1.51e-1) -	1.9255e-1 (1.13e-1) -	**9.1006e-2 (3.03e-2)**
CF6	2	10	4.7460e-2 (2.25e-2) -	6.2930e-2 (2.17e-2) -	9.8892e-2 (2.48e-2) -	2.3906e-2 (4.97e-3) -	4.6095e-2 (1.68e-2) -	4.8197e-2 (3.27e-2) -	2.2280e-2 (7.41e-3) -	**1.3571e-2 (2.02e-3)**
CF7	2	10	2.5270e-1 (1.23e-1) -	3.6060e-1 (1.77e-1) -	3.4419e-1 (9.83e-2) -	1.7729e-1 (1.29e-1) -	2.5081e-1 (7.43e-2) -	2.3604e-1 (1.12e-1) -	1.5980e-1 (1.01e-1) -	**8.4785e-2 (5.77e-2)**
CF8	3	10	1.8022e-1 (5.46e-2) -	2.7800e-1 (1.37e-1) -	1.6721e-1 (3.70e-2) -	1.6383e-1 (1.28e-2) -	3.0079e-1 (2.35e-1) -	**1.3832e-1 (3.20e-2) +**	1.4600e-1 (8.96e-3) =	1.4527e-1 (3.14e-2)
CF9	3	10	8.9601e-2 (4.59e-3) -	9.6116e-2 (2.31e-2) -	9.2123e-2 (1.93e-2) -	1.0665e-1 (9.84e-3) -	1.1502e-1 (4.58e-2) -	**6.8981e-2 (3.15e-3) =**	8.5790e-2 (3.77e-3) -	7.2146e-2 (9.87e-3)
CF10	3	10	3.7784e-1 (7.98e-2) -	3.8847e-1 (1.39e-1) -	3.0790e-1 (1.67e-1) =	3.4061e-1 (1.10e-1) -	3.8576e-1 (1.81e-1) -	3.8953e-1 (1.23e-1) -	2.5605e-1 (1.06e-1) =	**2.2508e-1 (8.96e-2)**
DASCMOP1	2	30	6.4316e-1 (8.73e-2) -	6.9355e-1 (7.81e-2) -	7.2016e-1 (3.50e-2) -	4.0756e-3 (6.96e-4) -	6.6153e-1 (8.00e-2) -	3.5429e-3 (2.94e-4) -	**2.7604e-3 (1.36e-4) +**	2.9136e-3 (2.60e-4)
DASCMOP2	2	30	2.2739e-1 (2.75e-2) -	2.5638e-1 (3.48e-2) -	2.4891e-1 (3.15e-2) -	4.8130e-3 (8.88e-5) -	2.2354e-1 (2.82e-2) -	1.7531e-2 (1.54e-2) -	4.3426e-3 (1.22e-4) -	**3.9413e-3 (6.58e-5)**
DASCMOP3	2	30	2.7713e-1 (4.99e-2) -	3.3574e-1 (2.64e-2) -	3.4812e-1 (3.09e-2) -	1.9312e-2 (1.61e-3) -	2.7340e-1 (4.81e-2) -	1.6086e-1 (2.81e-2) -	2.4916e-1 (1.43e-1) -	**1.6759e-2 (3.40e-3)**
DASCMOP4	2	30	1.4391e-3 (8.04e-4) =	6.8809e-2 (1.12e-1) -	1.7311e-2 (6.11e-2) -	6.7397e-1 (4.19e-1) -	**1.1519e-3 (2.03e-5) +**	1.8801e-3 (7.62e-4) -	1.3931e-3 (1.91e-4) -	1.2426e-3 (1.34e-4)
DASCMOP5	2	30	3.0351e-3 (2.82e-4) -	3.6461e-2 (1.10e-1) -	**2.6694e-3 (1.48e-4) +**	3.7063e-1 (3.91e-1) -	2.7608e-3 (6.87e-5) +	3.1941e-3 (2.06e-4) -	2.8725e-3 (1.31e-4) =	2.8697e-3 (1.09e-4)
DASCMOP6	2	30	1.8769e-2 (2.10e-3) =	3.2936e-1 (1.58e-1) -	1.6054e-1 (1.90e-1) -	5.0704e-1 (3.21e-1) -	3.0923e-2 (2.91e-2) =	**1.4867e-2 (3.23e-3) +**	1.4076e-1 (1.76e-1) -	1.7548e-2 (3.18e-3)
DASCMOP7	3	30	3.1655e-2 (7.82e-4) -	3.9661e-2 (1.11e-3) -	5.4122e-2 (9.87e-2) -	5.1737e-1 (4.49e-1) -	3.0559e-2 (6.68e-4) =	4.5828e-2 (1.96e-3) -	3.1410e-2 (8.62e-4) -	**3.0332e-2 (8.09e-4)**
DASCMOP8	3	30	4.1131e-2 (1.79e-3) -	5.5499e-2 (1.66e-3) -	3.9631e-2 (1.05e-3) =	3.6146e-1 (3.56e-1) -	4.2020e-2 (1.18e-2) -	5.4344e-2 (3.15e-3) -	4.0065e-2 (1.04e-3) -	**3.9242e-2 (8.30e-4)**
DASCMOP9	3	30	2.0756e-1 (8.63e-2) -	4.1406e-1 (5.68e-2) -	3.6441e-1 (6.97e-2) -	5.6346e-2 (2.99e-3) -	1.9747e-1 (6.43e-2) -	1.0506e-1 (5.86e-2) -	**3.9351e-2 (8.75e-4) =**	3.9355e-2 (7.89e-4)
LIRCMOP1	2	30	1.4651e-1 (4.12e-2) -	2.6093e-1 (4.85e-2) -	3.3180e-1 (4.86e-2) -	5.6899e-2 (2.65e-2) -	1.7700e-1 (7.26e-2) -	1.4342e-2 (1.38e-3) -	8.9616e-2 (3.80e-2) -	**8.2244e-3 (8.14e-4)**
LIRCMOP2	2	30	1.3386e-1 (3.75e-2) -	2.2658e-1 (2.41e-2) -	2.5965e-1 (4.15e-2) -	3.7964e-2 (1.64e-2) -	1.6231e-1 (4.37e-2) -	1.4263e-2 (9.53e-4) -	2.7711e-2 (5.44e-2) -	**7.3765e-3 (7.46e-4)**
LIRCMOP3	2	30	1.5561e-1 (5.69e-2) -	2.7579e-1 (3.58e-2) -	3.1479e-1 (4.26e-2) -	8.0724e-2 (5.30e-2) -	1.5519e-1 (6.07e-2) -	3.0927e-2 (1.59e-2) -	5.5882e-2 (5.83e-2) =	**4.8759e-3 (8.80e-4)**
LIRCMOP4	2	30	1.6106e-1 (4.62e-2) -	2.6895e-1 (3.40e-2) -	2.9325e-1 (2.89e-2) -	6.3130e-2 (4.89e-2) -	1.8236e-1 (6.85e-2) -	3.5660e-2 (1.95e-2) -	6.3775e-2 (7.63e-2) -	**4.6434e-3 (7.92e-4)**
LIRCMOP5	2	30	3.0991e-1 (4.94e-2) -	1.2241e+0 (3.96e-3) -	3.0733e-1 (5.69e-2) -	1.2473e-2 (8.09e-3) -	5.7339e-1 (4.29e-1) -	3.0521e-1 (4.44e-2) -	1.2796e-2 (3.36e-3) -	**6.7303e-3 (5.19e-4)**
LIRCMOP6	2	30	3.4927e-1 (8.93e-2) -	1.3449e+0 (1.03e-4) -	3.1185e-1 (6.10e-2) -	9.3536e-3 (1.16e-3) -	7.4078e-1 (5.05e-1) -	3.5389e-1 (8.99e-2) -	1.1921e-2 (1.08e-2) -	**6.4615e-3 (3.27e-4)**
LIRCMOP7	2	30	1.2784e-1 (3.25e-2) -	3.6429e-1 (5.26e-1) -	1.1747e-1 (2.62e-2) -	8.2905e-3 (2.92e-4) -	1.2795e-1 (4.17e-2) -	1.1964e-1 (2.56e-2) -	7.6623e-3 (3.20e-4) -	**7.1721e-3 (5.58e-4)**
LIRCMOP8	2	30	1.6935e-1 (4.06e-2) -	1.0207e+0 (7.18e-1) -	1.8333e-1 (4.53e-2) -	8.2702e-3 (2.75e-4) -	1.8153e-1 (5.00e-2) -	1.7585e-1 (5.23e-2) -	7.6086e-3 (2.81e-4) -	**7.0641e-3 (2.75e-4)**
LIRCMOP9	2	30	3.5242e-1 (9.55e-2) -	7.9350e-1 (1.21e-1) -	6.4547e-1 (2.70e-1) -	2.8843e-1 (1.67e-1) -	5.7035e-1 (1.63e-1) -	4.0224e-1 (4.42e-2) -	2.3953e-1 (1.57e-1) -	**6.0417e-2 (3.13e-2)**
LIRCMOP10	2	30	1.6125e-1 (6.44e-2) -	7.9258e-1 (1.51e-1) -	3.2373e-1 (1.52e-1) -	1.1306e-2 (2.20e-3) -	1.7869e-1 (1.01e-1) -	3.8656e-1 (5.87e-2) -	2.8945e-2 (4.83e-2) -	**6.1379e-3 (2.99e-4)**
LIRCMOP11	2	30	6.6518e-2 (3.33e-2) -	7.0388e-1 (1.02e-1) -	2.4829e-1 (8.59e-2) -	1.3416e-2 (3.09e-2) -	8.7459e-2 (8.04e-2) -	1.7764e-1 (6.95e-2) -	5.2265e-2 (6.56e-2) -	**2.5686e-3 (2.59e-4)**
LIRCMOP12	2	30	1.2562e-1 (3.49e-2) -	5.6766e-1 (1.19e-1) -	2.7409e-1 (9.65e-2) -	6.8410e-2 (1.05e-1) -	2.3143e-1 (1.46e-1) -	2.2826e-1 (4.89e-2) -	1.5152e-1 (1.14e-1) -	**3.7358e-3 (5.29e-4)**
LIRCMOP13	3	30	9.3505e-2 (1.15e-3) +	1.3148e+0 (2.38e-3) -	9.2445e-2 (6.35e-4) +	1.3340e-1 (3.66e-3) -	9.3229e-2 (6.89e-4) +	**8.6482e-2 (7.28e-4) +**	1.0241e-1 (2.01e-3) -	1.0148e-1 (1.46e-3)
LIRCMOP14	3	30	9.5119e-2 (8.93e-4) +	1.2712e+0 (1.94e-3) -	**9.4820e-2 (9.38e-4) +**	1.1777e-1 (4.24e-3) +	9.5550e-2 (9.89e-4) +	1.0086e-1 (2.03e-3) +	9.7693e-2 (1.05e-3) =	1.7562e-1 (2.96e-1)
+/-/=			2/29/2	0/33/0	3/28/2	2/31/0	4/27/2	4/28/1	1/25/7	

and the third governs convergence challenges by controlling the size of infeasible regions. ACEMT excels and secures the best IGD results in 6 out of the 9 test problems. ACEMT stands out on DASCMOP2, 3, and 7-8, and achieves the second position on DASCMOP1. However, for DASCMOP4-5, MTCMO takes the lead, and ACEMT falls behind CMEGL, CMOEMT, and MTCMO. Additionally, on DASCMOP4, ACEMT secures the second position, following MSCMO, and on DASCMOP6, it performs less effectively than ShiP$^+$. While mean IGD values indicate subtle differences among solvers in DASCMOP4-5, and CPFs are reachable (as the results in the supplementary file), ACEMT, even in cases of similar performance, remains competitive, not far from optimal results. In pairwise comparisons, ACEMT consistently demonstrates its superiority or comparability to competitors in at least half of the DASCMOP problems. On the LIRCMOP test suite, this suite is characterized by having many extensive infeasible regions and relatively confined feasible regions. ACEMT excels in these challenging benchmarks. When compared to seven competitors following the CEMT-based framework, ACEMT consistently achieves the top position in terms of IGD results for most cases. There are exceptions in LIRCMOP9, 13, and 14, where ACEMT

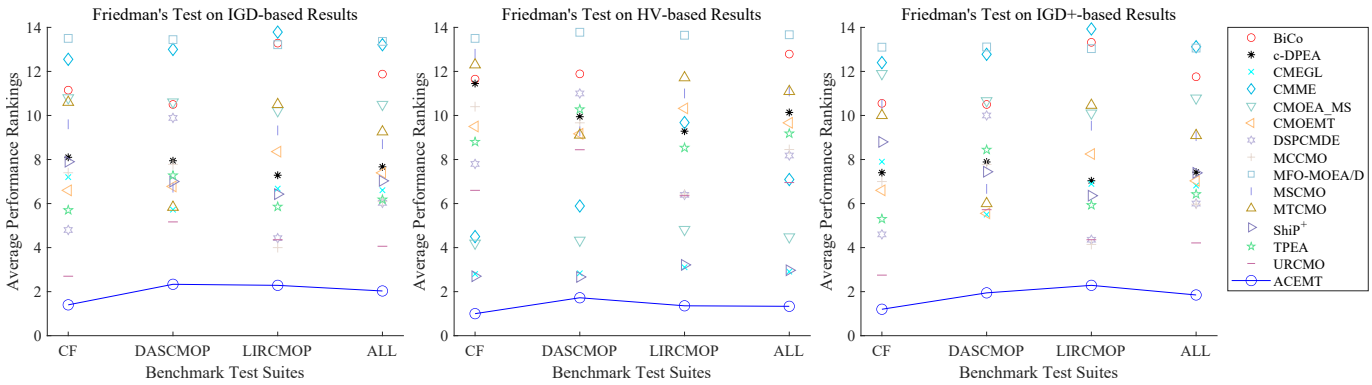

Fig. 5. The rankings of Friedman's test obtained by ACEMT and comparison algorithms on all test suites respectively over 30 runs.

performs less effectively than MCCMO on LIRCMOP9 and underperforms on LIRCMOP13-14 compared to CMEGL and TPEA. When assessing ACEMT against non-CEMT-based competitors, ACEMT consistently leads in terms of IGD for most cases, except for LIRCMOP13 and 14. ACEMT surpasses CMME, DSPCMDE and URCMO on LIRCMOP13 and only outperforms CMME on LIRCMOP14.

The comprehensive examination of ACEMT across these benchmark suites underscores its significant advantages. Unlike its CEMT-based counterparts, which often rely on a single fixed auxiliary task, ACEMT introduces innovation with two adaptive auxiliary tasks: the ACR-based task and the ACS-based task. Notably, MTCMO also demonstrates promising results, particularly in addressing DASCMOP1-3 with 11 constraints. Recognizing this, we enhance the construction method by introducing ACS-based auxiliary tasks, maintaining processing efficiency even in scenarios with numerous constraints. The experimental validation confirms the effectiveness of the ACS-based auxiliary task. In instances like solving LIRCMOP1-4, with only 2-3 constraints and a highly confined feasible region, the ACS-based auxiliary task may redundantly explore limited spaces. In such scenarios, the ACR-based auxiliary task proves more beneficial. The outstanding performance achieved in solving these test problems highlights the effectiveness of the ACEMT framework.

Moreover, in comparison to non-CEMT-based competitors, ACEMT consistently exhibits superior performance in addressing these problems. To wrap up our evaluation, we utilize the KEEL tool [53] to assess and rank the performance of all participating solvers through the Friedman test, considering their IGD results. The results are depicted in Fig. 5, showing the average performance rankings for ACEMT and its competitors. ACEMT consistently shows the top rank across all three test benchmark suites. Furthermore, as depicted in Fig. 6, we present the final solutions obtained by ACEMT and its competitors in solving the LIRCMOP4. It is apparent that LIRCMOP4 comprises a disconnected true CPF, consisting of multiple small feasible regions. Among the competitors, such as CMME, CMOEA-MS, CMOEMT, and MFO-MOEA/D, they can only identify one or two regions, typically the easiest to locate. In stark contrast, ACEMT stands out by successfully discovering all feasible regions, providing a comprehensive coverage of the CPF (The figures for other test problems are available in the supplementary file).

C. Parameter Sensitivity Analysis

In ACEMT, three parameters—α, k, and Δ—play crucial roles. To evaluate the impact of varying parameter values on ACEMT, we carried out parameter sensitivity analysis experiments in solving CF, DASCMOP, and LIRCMOP problems. The basic settings for the three parameters in ACEMT are $\alpha = 0.7$, $\Delta = 50$, and $k = \lfloor 0.5l \rfloor$.

The parameter Δ controls how frequently the id is updated in the ACS-based auxiliary task. A larger Δ can lead to rapid updates, potentially limiting the exploration of the auxiliary population P_s within the current feasible regions. Conversely, a smaller Δ may slow down updates, potentially causing the P_s to converge stagnate. To assess the sensitivity of this parameter, we selected 10 values evenly distributed between 10 and 100, i.e., Δ = 10, 20, 30, 40, 50, 60, 70, 80, 90, and 100 generations. The results obtained by ACEMT with different Δ values across all 33 benchmarks can be found in the supplementary file. Fig. 7.(a) presents the ranks of ACEMT under varying Δ values based on the IGD results. Notably, ACEMT's performance is weakest on the CF test suite when Δ = 40. On DASCMOP with a substantial number of constraints, ACEMT performs sub-optimally when Δ exceeds 60. This is due to certain constraints may lead P_s fall into local regions. When Δ = 60 or 70, ACEMT demonstrates significant improvements in solving the LIRCMOP test suite. After evaluating the performance across all test suites with varying Δ values, it becomes apparent that setting Δ to 60 consistently yields the best results.

The parameter k determines the maximum number of constraints considered in the ACS-based task, and it plays a vital role in updating the id vector. When $|id|$ exceeds k, the id is updated following the "first-add first-replaced" principle. To investigate its impact, we vary k as a fraction of l, taking values of $\lfloor 0.1l \rfloor, \lfloor 0.3l \rfloor, \lfloor 0.5l \rfloor, \lfloor 0.7l \rfloor, \lfloor 0.9l \rfloor$, and even $\lfloor \sqrt{l} \rfloor$. The results obtained by ACEMT with different k values across all benchmarks can be found in the supplementary file. Fig. 7(b) displays the Friedman's rankings of ACEMT under these different k values. Notably, ACEMT performs optimally on the CF and LIRCMOP test suites when k is set to $\lfloor 0.7l \rfloor$, while it

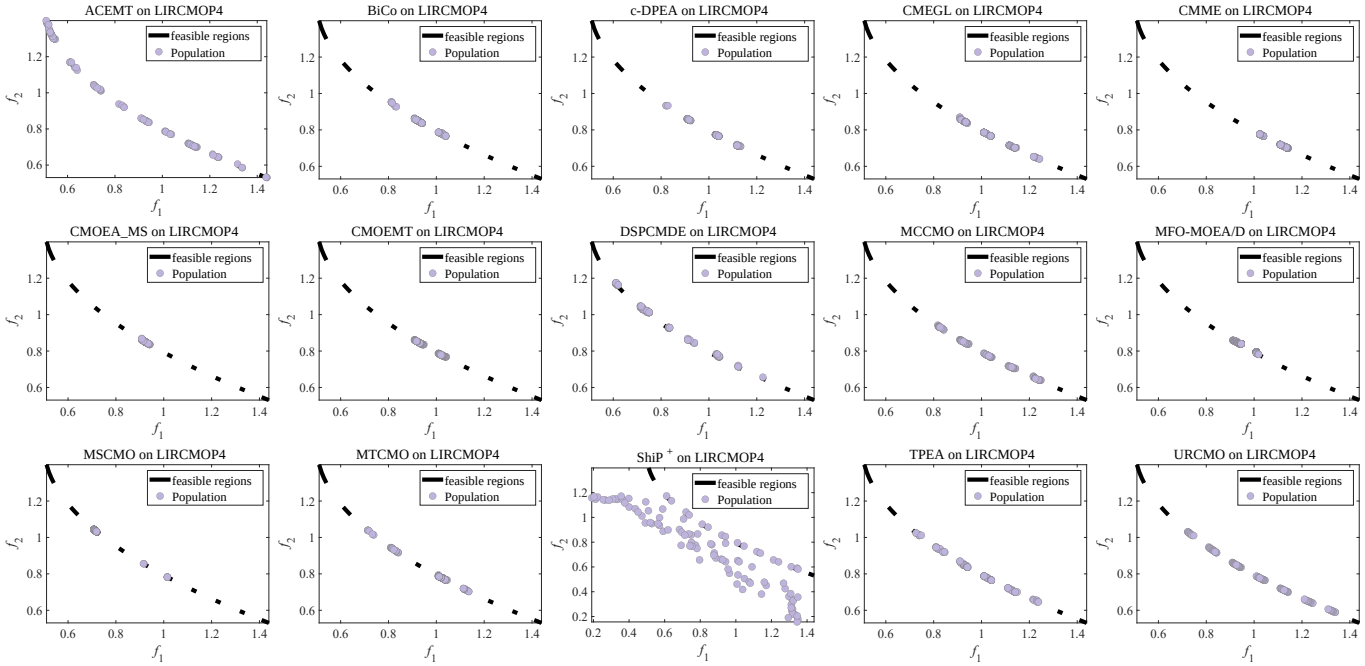

Fig. 6. The final population of ACEMT and comparison algorithm on LIRCMOP4

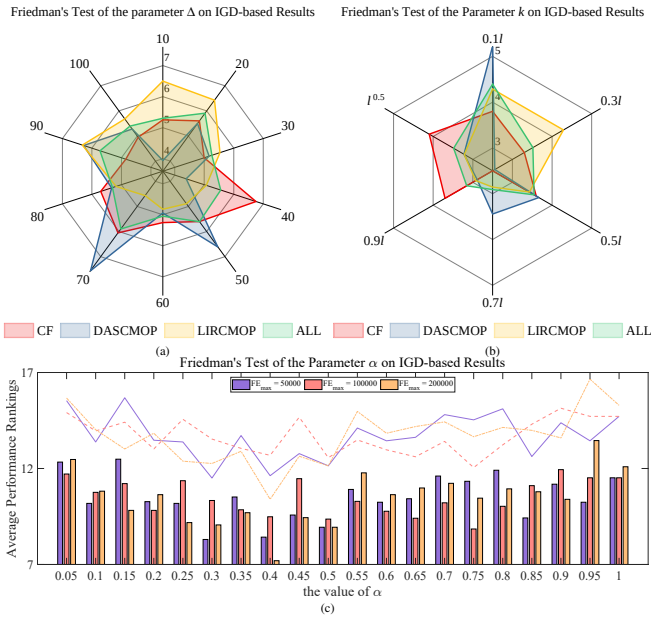

Fig. 7. The Rankings of Friedman's test of all parameters on test suites.

achieves the best results on the DASCMOP test suite with k equal to $\lfloor 0.3l \rfloor$. In summary, setting k to $\lfloor 0.7l \rfloor$ consistently yields the best results across all test suites.

The parameter α determines when the relaxation coefficient ε reaches zero in the ACR-based task. To investigate its influence, we explore different values for α, ranging from 0 to 1 with an interval of 0.05. All the IGD results obtained by ACEMT with varying α on all benchmarks are included in the supplementary file. Figure 7(c) illustrates the Friedman's ranks of ACEMT under different α values. Intriguingly, setting

TABLE III
THE WILCOXON TEST AND FRIEDMAN RANK OF THE MEAN IGD BY ACEMT AND ITS VARIANTS ON TEST SUITES OVER 30 RUNS.

Algorithm	CF		DASCMOP		LIRCMOP		ALL	
	+/-/=	Rank	+/-/=	Rank	+/-/=	Rank	+/-/=	Rank
ACEMT-ACS	2/3/5	2	1/3/5	2.44	2/6/6	2.14	5/12/16	2.18
ACEMT-ACR	1/4/5	2.3	1/0/8	1.39	0/7/7	2.29	2/11/20	2.03
ACEMT	-/-/-	1.7	-/-/-	2.17	-/-/-	1.57	-/-/-	1.79

α to values that are either too small (e.g., $\alpha < 0.25$) or too large (e.g., $\alpha > 0.9$) led to a deterioration in ACEMT's performance. When α is in the range of 0.3 to 0.5, ACEMT can achieve outstanding performance. When $\epsilon = 0$, the ACR-based auxiliary task will become the same as the main task, and we expect that ϵ becomes 0 as late as possible. This can be attributed to the larger setting of FE_{max}, which allows the population to enter the feasible regions early, resulting in $\alpha < 0.5$ being better than $\alpha > 0.5$. In order to further determine the optimal value of α, we conduct a separate analysis. By reducing the value of FE_{max} and performing Friedman's ranking based on the mean IGD obtained on the same test suites, the performance ranking have changed, as shown in the red and orange part of Figure 7(c). This observation suggests that smaller FE_{max} does have an effect on α. When $\alpha = 0.5$, FE_{max} has the least impact on α. Therefore, setting α to 0.5 is recommended.

D. Ablation Studies on ACEMT

In this section, we perform ablation studies on ACEMT to assess the effectiveness of its two auxiliary tasks. We create two variants, ACEMT-ACR and ACEMT-ACS, each focusing on one auxiliary task while excluding the other. ACEMT-ACR evaluates the ACR-based task by omitting the ACS-based task,

while ACEMT-ACS assesses the ACS-based task by excluding the ACR-based task. We then assess the performance of ACEMT and its two variants on three benchmark suites: DASCMOP, LIRCMOP, and CF, with detailed results available in the supplementary file. Table III presents the findings from Friedman and Wilcoxon tests on the IGD results obtained in solving these test suites, shedding light on the relative effectiveness of ACEMT and its ablated versions.

In the context of solving CF and LIRCMOPs, ACEMT-ACS outperforms ACEMT-ACR and is only ranked lower than ACEMT. Conversely, when addressing the DASCMOPs, ACEMT-ACR excels over ACEMT-ACS and even outperforms ACEMT in the rankings. This observation underscores the adaptability of these two auxiliary tasks to different types of benchmarks. Except for DASCMOP, ACEMT consistently outperforms its two variants, reaffirming the complementary nature of these auxiliary tasks. Their combined use offers more robust support to the main task. From the mean IGD in the Table XIV, it can be seen that ACEMT-ACR is not superior to ACEMT in solving DASCMOs.

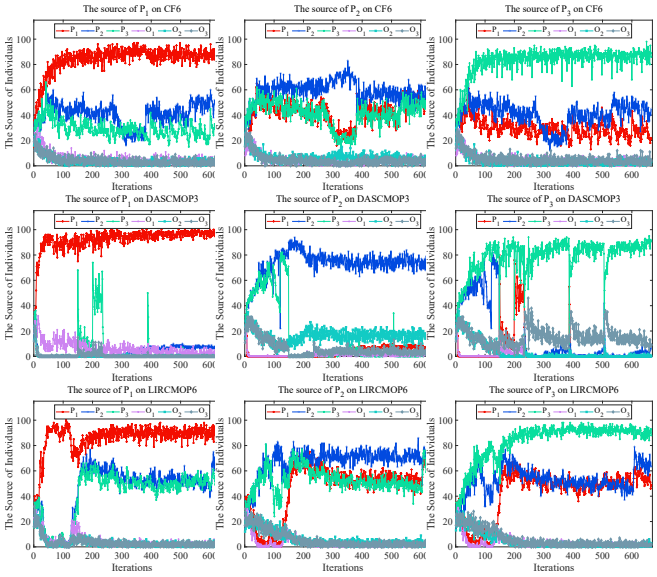

Fig. 8. The sources of individuals in three tasks in environmental selection strategies on CF6, DASCMOP3, and LIRCMOP6.

E. Study on the Knowledge Sharing

In ACEMT, there's a unique feature that distinguishes it from existing CEMT-based CMOEA: all three tasks share both parents and offspring for environmental selection. This departure from conventional practices in the field raises questions about how this arrangement impacts the solutions obtained in the updated populations (P, P_r, and P_s) for these three tasks. To understand this, we conducted a thorough analysis. In Figure 8, we present the statistical results for CF6, DASCMOP3, and LIRCMOP3, involving the counting of surviving solutions from each population, including P, Q, P_r, Q_r, P_s, and Q_s, separately. The results provide insights into the dynamics of the environmental selection process. In the early stages of evolution, the offspring population plays a more substantial role. However, as evolution progresses, a shift occurs, with the majority of surviving solutions originating from the parent populations of the three tasks, and fewer offspring solutions being selected. This means that parent populations from each task make a more significant contribution to help other tasks as evolution advances, particularly in the later stages. Nevertheless, it's essential to recognize that the newly generated offspring acts as a driving force for advancing the next-generation population and also contributes to diversity maintenance. This intricate interplay between parents and offspring ultimately led to our decision to share knowledge among all three tasks, a choice that has proven highly effective.

V. Conclusions

This paper has proposed an adaptive coevolutionary multitasking (ACEMT) framework to address the challenges of solving CMOPs. ACEMT utilizes three interconnected optimization tasks, each with its dedicated population and distinct goals, fostering knowledge transfer among them. The framework introduces two adaptive auxiliary tasks: one focuses on constraint relaxation to enhance diversity, accompanied by an adaptive angle-based selection strategy. The other task centers on adaptive constraint selection, intentionally omitting or focusing on specific constraints to emphasize improved convergence. Extensive experiments demonstrate ACEMT's superiority, consistently outperforming or rivaling state-of-the-art CMOEAs. This research contributes valuable insights into effectively tackling the complexities of CMOPs through ACEMT. Future improvements for the ACEMT could focus on more diverse auxiliary task design for increased adaptability and hybridization with other learning techniques to harness complementary strengths.

References

[1] Q. Xu, S. Zeng, F. Zhao, R. Jiao, and C. Li, "On formulating and designing antenna arrays by evolutionary algorithms," *IEEE Transactions on Antennas and Propagation*, vol. 69, no. 2, pp. 1118–1129, 2021.

[2] Z. Su, G. Zhang, F. Yue, D. Zhan, M. Li, B. Li, and X. Yao, "Enhanced constraint handling for reliability-constrained multiobjective testing resource allocation," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 3, pp. 537–551, 2021.

[3] R. Saravanan, S. Ramabalan, N. G. R. Ebenezer, and C. Dharmaraja, "Evolutionary multi criteria design optimization of robot grippers," *Applied Soft Computing*, vol. 9, no. 1, pp. 159–172, 2009.

[4] X. Zuo, C. Chen, W. Tan, and M. Zhou, "Vehicle scheduling of an urban bus line via an improved multiobjective genetic algorithm," *IEEE Transactions on intelligent transportation systems*, vol. 16, no. 2, pp. 1030–1041, 2014.

[5] Y. Cui, Z. Geng, Q. Zhu, and Y. Han, "Multi-objective optimization methods and application in energy saving," *Energy*, vol. 125, pp. 681–704, 2017.

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

[7] M. Ming, A. Trivedi, R. Wang, D. Srinivasan, and T. Zhang, "A dual-population-based evolutionary algorithm for constrained multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 4, pp. 739–753, 2021.

[8] Z. Ma and Y. Wang, "Shift-based penalty for evolutionary constrained multiobjective optimization and its application," *IEEE Transactions on Cybernetics*, vol. 53, no. 1, pp. 18–30, 2023.

[9] Z. Sun, H. Ren, G. G. Yen, T. Chen, J. Wu, H. An, and 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.

[10] Y. Yuan, W. Gao, L. Huang, H. Li, and J. Xie, "A two-phase constraint-handling technique for constrained optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–10, 2023.

[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] F. Qian, B. Xu, R. Qi, and H. Tianfield, "Self-adaptive differential evolution algorithm with α-constrained-domination principle for constrained multi-objective optimization," *Soft computing*, vol. 16, pp. 1353–1372, 2012.

[13] C. Zhang, A. K. Qin, W. Shen, L. Gao, K. C. Tan, and X. Li, "ϵ-constrained differential evolution using an adaptive ϵ-level control method," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 2, pp. 769–785, 2022.

[14] Z. Ma, Y. Wang, and W. Song, "A new fitness function with two rankings for evolutionary constrained multiobjective optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 8, pp. 5005–5016, 2021.

[15] Z.-Z. Liu, Y. Qin, W. Song, J. Zhang, and K. Li, "Multiobjective-based constraint-handling technique for evolutionary constrained multiobjective optimization: A new perspective," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2022.

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

[17] Z. Fan, W. Li, X. Cai, H. Li, C. Wei, Q. Zhang, K. Deb, and E. D. Goodman, "Push and pull search for solving constrained multi-objective optimization problems," *Swarm and Evolutionary Computation*, vol. 44, 2017.

[18] K. Zhang, Z. Xu, G. G. Yen, and L. Zhang, "Two-stage multi-objective evolution strategy for constrained multi-objective optimization," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2022.

[19] 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*, vol. 53, no. 6, pp. 3873–3886, 2023.

[20] J. Li, J. Chen, B. Xin, and L. Chen, "Efficient multi-objective evolutionary algorithms for solving the multi-stage weapon target assignment problem: A comparison study," in *2017 IEEE Congress on Evolutionary Computation (CEC)*, 2017, pp. 435–442.

[21] 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, 2020.

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

[23] T. Xu, J. He, and C. Shang, "Helper and equivalent objectives: Efficient approach for constrained optimization," *IEEE Transactions on Cybernetics*, vol. 52, no. 1, pp. 240–251, 2022.

[24] 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.

[25] Y. Hou, Y. Wu, and H. Han, "Multistate-constrained multiobjective differential evolution algorithm with variable neighborhood strategy," *IEEE Transactions on Cybernetics*, vol. 53, no. 7, pp. 4459–4472, 2023.

[26] J. Wang, G. Liang, and J. Zhang, "Cooperative differential evolution framework for constrained multiobjective optimization," *IEEE Transactions on Cybernetics*, vol. 49, no. 6, pp. 2060–2072, 2019.

[27] R. Jiao, B. Xue, and M. Zhang, "A multiform optimization framework for constrained multiobjective optimization," *IEEE Transactions on Cybernetics*, vol. 53, no. 8, pp. 5165–5177, 2023.

[28] 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.

[29] Z.-Z. Liu, B.-C. Wang, and K. Tang, "Handling constrained multiobjective optimization problems via bidirectional coevolution," *IEEE Transactions on Cybernetics*, vol. 52, no. 10, pp. 10 163–10 176, 2022.

[30] Z.-Z. Liu, F. Wu, J. Liu, Y. Qin, and K. Li, "Constrained multiobjective optimization with escape and expansion forces," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2023.

[31] C. Peng, H.-L. Liu, and E. D. Goodman, "A cooperative evolutionary framework based on an improved version of directed weight vectors for constrained multiobjective optimization with deceptive constraints," *IEEE Transactions on Cybernetics*, vol. 51, no. 11, pp. 5546–5558, 2021.

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

[33] J. Zou, R. Sun, Y. Liu, Y. Hu, S. Yang, J. Zheng, and K. Li, "A multi-population evolutionary algorithm using new cooperative mechanism for solving multi-objective problems with multi-constraint," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2023.

[34] J. Wang, Y. Li, Q. Zhang, Z. Zhang, and S. Gao, "Cooperative multiobjective evolutionary algorithm with propulsive population for constrained multiobjective optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 6, pp. 3476–3491, 2022.

[35] M. Ming, R. Wang, H. Ishibuchi, and 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.

[36] 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.

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

[38] Z. Fan, W. Li, X. Cai, H. Li, C. Wei, Q. Zhang, K. Deb, and E. Goodman, "Difficulty adjustable and scalable constrained multiobjective test problem toolkit," *Evolutionary computation*, vol. 28, no. 3, pp. 339–378, 2020.

[39] 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. 12 491–12 510, 2019.

[40] S. Liu, J. Zheng, Q. Lin, and K. C. Tan, "Evolutionary multi and many-objective optimization via clustering for environmental selection," *Information Sciences*, vol. 578, pp. 930–949, 2021.

[41] S. Liu, Q. Lin, K. C. Tan, M. Gong, and C. A. Coello Coello, "A fuzzy decomposition-based multi/many-objective evolutionary algorithm," *IEEE Transactions on Cybernetics*, vol. 52, no. 5, pp. 3495–3509, 2022.

[42] Q. Lin, S. Liu, K.-C. Wong, M. Gong, C. A. Coello Coello, J. Chen, and J. Zhang, "A clustering-based evolutionary algorithm for many-objective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 3, pp. 391–405, 2019.

[43] S. Liu, Q. Lin, K.-C. Wong, C. A. Coello Coello, J. Li, Z. Ming, and J. Zhang, "A self-guided reference vector strategy for many-objective optimization," *IEEE Transactions on Cybernetics*, vol. 52, no. 2, pp. 1164–1178, 2022.

[44] 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.

[45] K. Qiao, J. Liang, Z. Liu, K. Yu, C. Yue, and B. Qu, "Evolutionary multitasking with global and local auxiliary tasks for constrained multi-objective optimization," *IEEE/CAA Journal of Automatica Sinica*, vol. 10, pp. 1–14, 2023.

[46] F. Ming, W. Gong, L. Wang, and L. Gao, "A constrained many-objective optimization evolutionary algorithm with enhanced mating and environmental selections," *IEEE Transactions on Cybernetics*, 2022.

[47] 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, 2021.

[48] P. Bosman and D. Thierens, "The balance between proximity and diversity in multiobjective evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 174–188, 2003.

[49] H. Ishibuchi, H. Masuda, Y. Tanigaki, and Y. Nojima, "Modified distance calculation in generational distance and inverted generational distance," in *Evolutionary Multi-Criterion Optimization: 8th International Conference, EMO 2015, Guimarães, Portugal, March 29–April 1, 2015. Proceedings, Part II 8*. Springer, 2015, pp. 110–125.

[50] E. Zitzler and 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.

[51] 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, 2017.

[52] 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.

[53] J. Alcala-Fdez, S. Garcia, F. J. Berlanga, A. Fernandez, L. Sanchez, M. del Jesus, and F. Herrera, "Keel: A data mining software tool integrating genetic fuzzy systems," in *2008 3rd International Workshop on Genetic and Evolving Systems*, 2008, pp. 83–88.