An Interval-Constrained MultiObjective Evolutionary Algorithm Integrating Interval Constrained Clustering and Transfer Learning

***Abstract*—Most practical engineering applications are often formulated as interval constrained multiobjective optimization problems (ICMOPs). In such problems, at least one objective or constraint contains interval parameters. With increasing constraints, the feasible region becomes narrow and fragmented. This greatly raises the difficulty of finding feasible solutions. Therefore, this paper proposes an interval-constrained multiobjective evolutionary algorithm integrating interval constraint clustering and transfer learning (ICCTL-ICMOEA). First, an interval constraint overlap degree is introduced to quantify constraint similarity, the Louvain algorithm is employed to cluster constraints into subclasses. Each constraint class is then combined with interval objectives to construct multiple subproblems. Next, within a multi-population evolutionary framework, a population state-aware search strategy is designed to independently evolve the main population and subpopulations. Following that, an interval constraint relaxation-guided knowledge transfer strategy is developed to migrate selected individuals from subpopulations to the main population, accelerating feasible region exploration. Finally, the proposed algorithm was tested on constructed benchmark ICMOPs and compared with a state-of-the-art interval-constrained multiobjective evolutionary algorithm and a constrained multiobjective evolutionary algorithm. Experimental results demonstrate the strong competitiveness of the proposed algorithm.**

***Index Terms*—Interval-constrained multiobjective optimization problems, interval constraint clustering, transfer learning, evolutionary algorithms**

I. Introduction

Many practical engineering problems involve multiple conflicting objectives and multiple constraints, and are typically modeled as constrained multiobjective optimization problems (CMOPs). In practice, prediction errors and equipment performance fluctuations lead to parameter uncertainty. Such parameters can be represented using random variables, fuzzy numbers, interval, etc. Intervals are often more intuitive and concise than the other representations. Furthermore, random numbers and fuzzy numbers can be converted into intervals. Therefore, representing uncertain parameters as intervals can simplify the modeling process for uncertainty. CMOPs with interval parameters are referred to as ICMOPs. Typical applications include portfolio decisions [1] and 5G base station placement [2]. This paper focuses on tackling ICMOPs.

For interval multiobjective optimization problems (IMOPs), Gan et al. proposed an IMOEA based on adaptive reference vectors within the MOEA/D framework [3]. Yu et al. proposed a non-probabilistic set selection strategy and applied dimensionality analysis to quantify the boundaries of interval objectives [4]. However, IMOEA does not consider constraints and cannot be directly applied to solving ICMOPs.

Research on interval-constrained multiobjective evolutionary algorithms (ICMOEAs) for ICMOPs remains limited. Zeng et al. designed an individual selection strategy based on interval constraint violation degree to optimize the configuration and scheduling of renewable energy systems in 5G base stations [2]. Yu et al. proposed a penalty-based ICMOEA with a two-stage strategy [5]. Current constraint handling techniques (CHTs) in ICMOEA are relatively simple, making it difficult to effectively address feasible region contraction and search difficulties under multiple constraints. In contrast, constrained multiobjective evolutionary algorithm (CMOEA) have developed a mature set of CHTs, providing important reference value for further research on ICMOPs.

Based on the evaluation and selection of candidate solutions, the phase division, as well as the construction of evolutionary populations, CMOEAs can be categorized into penalty function methods [6], objective-constraint separation approaches [7], multi-stage techniques [8], and multi-population strategies [9]. Ma et al. combined shift operations with penalty mechanisms to guide infeasible solutions toward the feasible region [6]. Deb et al. proposed the constraint-dominance principle, which always prefers feasible solutions; if both candidates are infeasible, the one with a smaller constraint violation is selected [7]. Penalty function and objective-constraint separation methods emphasize feasibility. However, under multiple constraints the feasible region often becomes narrow and fragmented, and the search may get trapped in local optimum. To address this issue, Fan et al. introduced the “push-pull search” framework, where the push phase considers only the objectives, while the pull phase simultaneously considers both objectives and constraints [8]. Liang et al. exploited the relationship between the unconstrained Pareto front (UPF) and the constrained Pareto front (CPF), and used feasibility and dominance information to guide the search toward the CPF [9]. The aforementioned approaches focus on the relationship between UPF and CPF when solving problems. For complex problems with multiple constraints, analyzing inter-constraint relationships is crucial, as they shape the feasible region and affect search efficiency.

Several studies have explored inter-constraint relationships. Zou et al. constructed a separate population for each constraint, analyzing the relationship between the UPF/common front of constraints and the CPF to design population activation/hibernation and combination mechanisms [10]. Qiao et al. transformed CMOP into a multi-task optimization problem, using single-constraint auxiliary tasks to search for the CPF. They selected highly correlated tasks based on the relative positions and dominance relationships between auxiliary task populations and the main task population in the objective space [11]. Despite considering constraint relationships, these methods may be computationally expensive, especially when they maintain one population per constraint. Xu et al. categorized constraints into strong and weak types, and formulated corresponding subproblems, with knowledge transfer to assist solving the original problem [12]. Different constraint grouping strategies can have different impacts on problem solving.

For ICMOPs, constraint violations are interval-valued. When two interval constraints have little overlap, it becomes harder to find solutions that satisfy both constraints simultaneously. The interrelationships among interval constraints directly impact the efficiency and direction of the search process. Therefore, this paper proposes an ICMOEA that integrates interval constraint clustering with transfer learning (ICCTL-ICMOEA).

The remainder of this paper is organized as follows. Section II reviews problem formulation. Section III presents the proposed algorithm. Section IV reports experimental results on constructed benchmark problems. Section V concludes the paper and discusses future work.

II. Problem Formulation

The mathematical formulation of ICMOPs [13] is given as follows.

$$\begin{aligned}
&\min F(\boldsymbol{x},\boldsymbol{c}) = (f_1(\boldsymbol{x},\boldsymbol{c}_1), f_2(\boldsymbol{x},\boldsymbol{c}_2), \ldots, f_m(\boldsymbol{x},\boldsymbol{c}_m))\\
&s.t. \begin{cases} \text{constraint}_j : g_j(\boldsymbol{x},\boldsymbol{c}_{m+j}) \le 0, & j = 1,\ldots,q \\ \text{constraint}_j : h_j(\boldsymbol{x},\boldsymbol{c}_{m+j}) = 0, & j = q+1,\ldots,n \end{cases}\\
&\boldsymbol{c}_i = (c_{i1}, c_{i2}, \ldots, c_{il_i})^T,\ c_{io} = [\underline{c}_{io}, \overline{c}_{io}]\\
&o = 1,2,\ldots,l_i,\ i = 1,2,\ldots,m+n.
\end{aligned} \tag{1}$$

Here, $\boldsymbol{x} = (x_1, x_2, \ldots, x_D)$ is a D-dimensional decision vector; $f_i(\boldsymbol{x},\boldsymbol{c}_i)$ $(i = 1,2,\ldots,m)$ denotes the i-th interval objectives. $g_j(\boldsymbol{x},\boldsymbol{c}_{m+j})$ and $h_j(\boldsymbol{x},\boldsymbol{c}_{m+j})$ represent the interval inequality and equality constraints, respectively. $\boldsymbol{c}_i$ is an interval parameter vector with l_i components, where each component c_{io} is an interval number, i.e., $c_{io} = [\underline{c}_{io}, \overline{c}_{io}]$. In particular, $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_m$ are associated with the m objectives, while $\boldsymbol{c}_{m+1}, \ldots, \boldsymbol{c}_{m+n}$ are associated with the n constraints. Since the coefficients are interval-valued, for any $\boldsymbol{x}$, each objective is also interval-valued, denoted as $f_i(\boldsymbol{x},\boldsymbol{c}_i) \triangleq [\underline{f}_i(\boldsymbol{x},\boldsymbol{c}_i), \overline{f}_i(\boldsymbol{x},\boldsymbol{c}_i)]$.

III. Proposed Algorithm

This section proposes an ICCTL-ICMOEA. First, it defines similarity using interval constraint overlap degree to cluster constraints. Each constraint class is then combined with interval objectives to construct multiple subproblems. Next, a population state-aware search strategy (PSASS) independently evolves the main population and subpopulations. Finally, transfer learning strategies transfer some individuals from subproblems to the original problem, aiding main population evolution to accelerate the search process and enhance solution efficiency.

A. Algorithm Framework

Algorithm 1 outlines the overall framework of ICCTL-ICMOEA. First, the main population $mainPop$ is initialized using Latin hypercube sampling (LHS), boundary sampling, and random sampling according to a certain proportion (Line 1). Then, the interval constraint clustering (ICC) method proposed in Section III.B partitions the constraints into K subclasses (Line 2). Each subclass is combined with the objectives to construct K subproblems (Lines 3–5). In addition, the subpopulation $SubPop_k$ $(k = 1,2,\ldots,K)$ is initialized using the same method as $mainPop$ (Line 6); Next, the PSASS proposed in Section III.C updates $mainPop$ and $SubPop_k$ (Line 8) independently; finally, the main population $mainPop$ is updated using the knowledge-transfer-based strategy proposed in Section III.D (Line 9). Repeat the above steps until the termination condition is met, then output the optimal population P (Line 11). Fig. 1 illustrates the overall framework of the proposed ICCTL-ICMOEA.

B. *Interval Constraint Cluster Guided by Constraint Overlap Degree*

In ICMOPs, interval parameters make each constraint violation degree interval-valued rather than deterministic. If the violation degrees of two constraints exhibit a large overlap for most individuals, the two constraints are considered highly similar. Furthermore, if constraints with high similarity are grouped with the objectives to form subproblems, the feasible region of these subproblems will be larger than that of the original problem, making them easier to solve. Based on this, this section proposes an ICC guided by constraint overlap degree. Constraints within the same subclasses exhibit high similarity, while those between subclasses exhibit low similarity.

The main steps are as follows. First, for each individual $\boldsymbol{x}_p$ in the population, the interval constraint violation degrees of each pair of constraints, $G_j(\boldsymbol{x}_p)$ and $G_h(\boldsymbol{x}_p)$, are calculated using the method in [13]. Specifically, they are represented as intervals

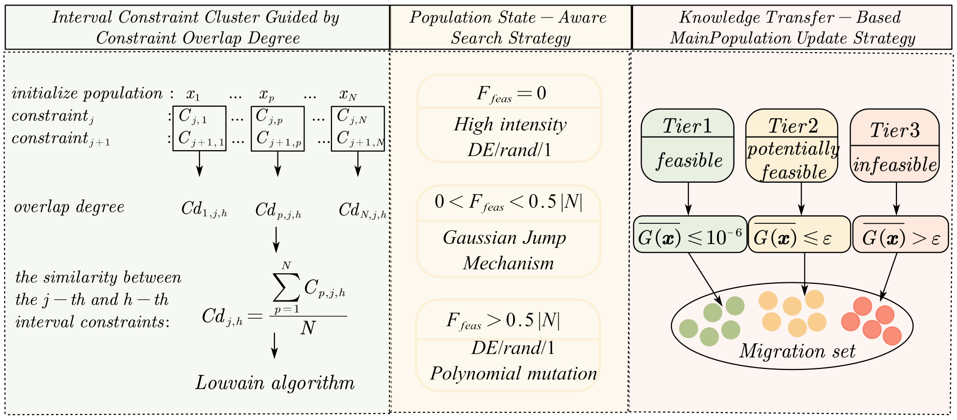


Fig. 1: Overall framework of the proposed ICCTL-ICMOEA.

Algorithm 1 Framework of ICCTL-ICMOEA

Input: Constraint set $constraint_j, (j = 1, \dots, n)$, uncertainty parameter ω, generations Gen, size N.
Output: Optimal population P.

Initialize $mainPop$ using LHS, Boundary Sampling, and Random Sampling;
$\{C_1, \dots, C_K\} = \text{ICC}(mainPop, c_j, N)$;
for $k = 1 : K$ **do**
    $subP_k = \{Obj; C_k\}$;
    % Obj is the interval objectives
end for
Initialize sub-population $SubPop_k$ corresponding to $subP_k$;
for $t = 1 : Gen$ **do**
    $[mainPop, SubPop_k] =$ PSASS$(mainPop, SubPop_k, \omega, N)$;
    $mainPop =$ MergeAndSelect $(mainPop, SubPop_k, t, Gen, c_j, N)$;
end for
$P \leftarrow mainPop$.

$[\underline{G_j}(\boldsymbol{x}_p), \overline{G_j}(\boldsymbol{x}_p)]$ and $[\underline{G_h}(\boldsymbol{x}_p), \overline{G_h}(\boldsymbol{x}_p)]$. Then, the interval overlap degree is defined by the intersection-over-union ratio of the corresponding violation intervals. The formula is given as follows.

$$C_{pjh} = \frac{W(G_j(\boldsymbol{x}_p) \cap G_h(\boldsymbol{x}_p))}{W(G_j(\boldsymbol{x}_p) \cup G_h(\boldsymbol{x}_p))} \tag{2}$$

where $W(\cdot)$ denotes the width of an interval. Subsequently, the similarity between the j-th and h-th interval constraints is quantified by the average overlap degree across the population $Cd_{jh} = \frac{\sum_{p=1}^{N} C_{pjh}}{N}$.

By computing Cd_{jh} for all constraint pairs, an $n \times n$ pairwise similarity matrix Cd is obtained, as shown in (6).

$$Cd = \begin{bmatrix} 1 & Cd_{12} & Cd_{13} & \dots & Cd_{1n} \\ Cd_{21} & 1 & Cd_{23} & \dots & Cd_{2n} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ Cd_{n1} & Cd_{n2} & \dots & \dots & 1 \end{bmatrix} \tag{3}$$

Since $Cd_{jh} = Cd_{hj}$, Cd is symmetric. Each element $Cd_{jh} \in [0, 1]$ indicates the similarity between two interval constraints; $Cd_{jh} = 0$ means that their constraint-violation intervals have no overlap, whereas $Cd_{jh} = 1$ means that the two intervals coincide. The diagonal elements satisfy $Cd_{jj} = 1$.

The similarity matrix Cd is directly interpreted as a weighted adjacency matrix to construct an undirected graph. Each node represents a constraint, and each edge weight corresponds to the coupling degree between a pair of constraints. Specifically, the weight between the j-th and h-th interval constraints is defined by the average overlap ratio of their interval constraint violation across the population (i.e., the ratio of intersection to union). Thus, no additional transformation or arbitrary mapping is required to convert the similarity values into graph edge weights. Finally, the constraints are clustered using the Louvain algorithm [14] based on Cd. The clustering procedure consists of two stages. In Stage 1, constraints connected by edges with similarity 1 are merged to form strongly connected groups. In Stage 2, the Louvain algorithm is applied to cluster the remaining constraints, yielding K subclasses $\{C_1, C_2, \dots, C_K\}$. By leveraging the intrinsic modularity-driven mechanism of the Louvain algorithm, our method adaptively identifies constraint communities based on the relative contrast of coupling strengths, thereby eliminating the need for an arbitrary manual threshold while maintaining hyperparameter parsimony.

C. *Population State-Aware Search Strategy*

Due to differences in the feasible domain shapes of various subproblems, the feasible domains of the original problem and subproblems also exhibit significant variations. As a result, the main population and subpopulations often experience distinct search phases characterized by varying numbers of feasible individuals. A fixed search strategy cannot adapt to such state changes: in early stages it may fail to guide the population into the feasible region efficiently, while in later stages it may waste excessive computational effort on feasibility maintenance, thereby reducing overall search efficiency.

During the evolutionary process, CMOP typically undergoes three distinct phases: exploring feasible solutions, developing the feasible region, and optimizing the objectives. Since the search focus differs across processes, the evolutionary operators should also be adjusted accordingly. Therefore, we propose a PSASS that uses the feasible-solution ratio F_{fea} as a state indicator to adaptively control the search operators.

Phase I: Exploring Feasible Solutions. When $F_{fea} = 0$, no feasible solutions exist. The search primarily aims to enter the feasible region. To this end, a high-intensity differential mutation is employed, temporarily emphasizing minimization of interval constraint violation degree rather than objective values. Specifically, a larger scaling factor $Factor = 0.6$ and a high crossover rate $CR = 0.9$ are used to enhance global exploration and facilitate early discovery of feasible solutions.

Phase II: Developing the Feasible Region. When $0 < F_{fea} < 0.5$, only a few feasible solutions exist and they are sparsely distributed, making the search prone to local optimum. To quickly locate the boundary of the feasible region, new solutions are generated via the following Gaussian leap mechanism:

$$\boldsymbol{x}'_p = \boldsymbol{x}_p + \frac{1}{2}(\boldsymbol{x}_p - \boldsymbol{x}_p^{fea}) + \mathcal{N}(0, \sigma_p^2) \tag{7}$$

where $\boldsymbol{x}_p^{fea}$ is a randomly selected feasible solution. The coefficient $\frac{1}{2}$ controls the step length along this direction to avoid generating solutions too far from the feasible region. $\mathcal{N}(0, \sigma_p^2)$ denotes a Gaussian perturbation with mean 0 and variance σ_p^2. The standard deviation is adaptively set as:

$$\sigma_p = \beta(u_p - l_p) \tag{8}$$

where u_p and l_p are the upper and lower bounds of the p-th decision variable, respectively, and β is randomly sampled from $[0.02, 0.12]$ to ensure an appropriate perturbation scale while promoting diversity near the boundary of the feasible region. If an offspring becomes infeasible, a binary boundary repair strategy [15] is applied to project it back toward the nearest feasible point along the parent offspring line.

Phase III: Optimizing the objectives. When $F_{fea} > 0.5$, a sufficient number of feasible solutions are available and the search focus shifts from feasibility to objective optimization. At this stage, standard DE/rand/1 operators combined with polynomial mutation are adopted to perform global optimization within the feasible region.

This strategy guides the population into the feasible region when feasible solutions are absent, explores feasible boundaries when feasible solutions are scarce, and switches to objective optimization once feasibility becomes sufficient, thereby maintaining appropriate exploration intensity and convergence speed across different phases.

D. Knowledge Transfer-Based Main Population Update Strategy

For ICMOPs, the constraint violation degree is represented as an interval $\left[\underline{G}(\boldsymbol{x}), \overline{G}(\boldsymbol{x})\right]$. When $\overline{G}(\boldsymbol{x}) \leqslant 10^{-6}$, the interval constraint violation degree is 0, indicating the individual is a feasible solution. For ICMOPs with multiple constraints, the feasible region is small. If only strictly feasible solutions are selected during environment selection, it may easily lead to restricted search or even stagnation.

When the lower bound $\underline{G}(\boldsymbol{x})$ of the interval constraint violation is small, the individual can be considered close to the feasible region and possesses the potential to transform into a feasible solution. Therefore, this section introduces an interval constraint relaxation strategy, which treats infeasible solutions with small interval constraint violations degree as promising candidates, thereby enhancing the exploration during the evolutionary process. Specifically, individuals are classified into three tiers according to the relaxation factor ε:

1) $\overline{G}(\boldsymbol{x}) \leqslant 10^{-6}$: strictly feasible solutions, *Tier 1*;
2) $\underline{G}(\boldsymbol{x}) \leqslant \varepsilon$:potentially feasible solutions, *Tier 2*;
3) $\underline{G}(\boldsymbol{x}) > \varepsilon$: infeasible solutions, *Tier 3*.

For each subproblem, the tolerance threshold $\varepsilon_{k,t}$ is set as the α-quantile of $\underline{G}(\boldsymbol{x})$ within the corresponding subpopulation.

$$\varepsilon_{k,t} = \text{Quantile}_\alpha\left(\{\underline{G}(\boldsymbol{x}) \mid \boldsymbol{x} \in N_k\}\right), \quad \alpha = 0.4 \tag{9}$$

where N_k denotes the solution set of the k-th subproblem, and $\text{Quantile}_\alpha(\cdot)$ denotes the α-quantile operator. Sort the infeasible solutions in ascending order by $\underline{G}(\boldsymbol{x})$. The top 40% of solutions are treated as potential solutions, with the largest $\underline{G}(\boldsymbol{x})$ serving as $\varepsilon_{k,t}$. The number of individuals migrated from each subpopulation is controlled by parameter ρ, set to $\lceil |N_k| \cdot \rho \rceil$, where $\rho = 0.5$.

The size of the elite archive is set to N. The migrated individuals, the main population, and the elite archive are merged into a set U. Under the global tolerance ϵ_t, environmental selection is then performed on U to update the main population $mainPop$. Specifically, Tier 1 and Tier 2 individuals are treated with equal priority and selected by non-dominated sorting. If fewer than N individuals are obtained, the remaining slots are filled with Tier 3 individuals in ascending order of constraint violation $G(x)$. The same selection criterion is also adopted in migration solution selection. The global tolerance ε_t is dynamically updated using a cosine annealing strategy.

$$\varepsilon_t = \begin{cases} 0, & t > 0.9Gen \\ \text{Sorted}_{\underline{G}(\boldsymbol{x})}(\lfloor |\text{Infeasible}| \cdot r(t) \rfloor), & \text{otherwise} \end{cases} \tag{10}$$

$$r(t) = r_0 \cdot \frac{1}{2}\left(\cos\left(\frac{t}{Gen}\pi\right) + 1\right), \quad r_0 = 0.5 \tag{11}$$

where $|Infeasible|$ denotes the number of infeasible solutions, and $\text{Sorted}_{G(x)}(q)$ denotes the q-th smallest

$G(x)$ after sorting them in ascending order, t denotes the current evolutionary generation and Gen is the maximum number of generations. Only within the top 90% evolutionary generations are *Tier 2* potential solutions permitted to compete, with $r(t)$ adaptively controlling their quantity. In the early stage, $r(t)$ decreases slowly to retain more potential solutions for exploration, whereas in the later stage, it decreases faster to reduce potential solutions and accelerate convergence toward the feasible region and the Pareto front.

E. Time Complexity

Suppose that the problem contains n constraints, the population size is N, and the number of objectives is m. In the interval constraint clustering stage, the similarity matrix is constructed by evaluating all constraint pairs over the whole population, resulting in a time complexity of $O(n^2N)$. In the multi-population evolutionary stage, the main population and subpopulations are evolved under the NSGA-II framework. Since the number of subpopulations is at most n, the worst-case time complexity is $O((n+1)mN^2)$. In addition, boundary repair by bisection requires $O(NB)$ time in the worst case, where B denotes the number of bisection iterations. In the main-population update stage, the main population, migrated solutions, and elite solutions are merged for environmental selection, with a time complexity of $O(mN_c^2)$, where N_c is the number of individuals in the combined set. In the worst case, its complexity is $O(mn^2N^2)$. Therefore, the overall time complexity of ICCTL-ICMOEA is $O(mn^2N^2)$.

For the compared algorithms, the computational complexity of CG-MOEA is $O(l(N + @))$. For example, the total computational complexity of CG-NSGA-II is $O(l(N+mN^2))$, where the computational complexity of NSGA-II is $O(mN^2)$. For PF-ICMOA, the overall time complexity is $O(N_f N_{pop}^2)$, where N_f denotes the number of objective functions and N_{pop} denotes the population size.

IV. Experimental Result

This section compares the proposed ICCTL-ICMOEA with an ICMOEA, namely, PF-ICMOA [5] and a CMOEA, namely, CG-based CMOEA [12]. The experimental environment is as follows: Windows 10, Intel i9-14900 @ 2.00 GHz, and MATLAB R2020b with the INTLAB 6.0 toolbox [3].

A. Benchmark Problems

Based on the existing constrained multiobjective benchmark functions, we introduce a perturbation factor [5] to construct a suite of benchmark problems with interval objectives and constraints, aiming to comprehensively evaluate algorithm performance under uncertain parameters. These problems are designed to comprehensively evaluate the optimization performance of algorithms under uncertain environments. The perturbation factors are set as follows.

$$\alpha = \begin{bmatrix} \sin\left(10\pi \sum\{f_1, f_2, \ldots, f_m\}\right)/4 \\ \sin\left(20\pi \sum\{f_1, f_2, \ldots, f_m\}\right)/4 \end{bmatrix} \cdot \omega \quad (12)$$

where ω controls the numerical width of the interval, we set $\omega = 0.5$. DASCMOP1-DASCMOP6 [16] were selected to construct the benchmark functions IDASCMOP1 to IDASCMOP6. All test problems have two objectives and 30 decision variables.

B. Experimental Parameter Settings

Both algorithms employ the following parameter settings: population size $N = 100$, maximum number of generations $Gen = 600$, with sampling ratios of LHS, boundary sampling, and random sampling set at 0.6, 0.2, and 0.2 respectively.

C. Performance Metrics

This study employs the following metrics to evaluate algorithm performance: 1) Hypervolume value (HV): The midpoint of the interval hypervolume [17] serves as the hypervolume value, with the reference interval for IDASCMOP1–IDASCMOP6 set at [2.5, 2.5]. 2) Imprecision (IM) [17]: Evaluates population uncertainty by calculating the mean imprecision across all individuals in the population. 3) Inverse Generation Distance (IGD) [17], reflecting the algorithm's convergence and distribution.

D. Experimental Results and Analysis

To assess performance, ICCTL-ICMOEA is compared with a state-of-the-art ICMOEA and a CMOEA on the constructed IDASCMOP test suite. Each algorithm was run independently 30 times. Table I and Table II report the mean and standard deviation of HV, IGD, IM, and Time. The best results are highlighted in bold. The Wilcoxon rank sum test was employed to examine statistical significance; symbols "+", "−", and "=" indicate that the compared algorithm performs significantly better than, significantly worse than, or not significantly different from ICCTL-ICMOEA, respectively.

From Tables I and II, several observations can be drawn. First, in terms of HV and IGD, the proposed ICCTL-ICMOEA achieves superior performance on most test problems, demonstrating its strong capability in balancing convergence and diversity. However, relatively weaker performance is observed on IDASCMOP5 and IDASCMOP6. The deterministic CG-based CMOEA attains competitive results on these problems by ignoring interval uncertainty, which simplifies the search space and facilitates the identification of narrow feasible regions. In contrast, when interval uncertainty is explicitly considered, the feasible regions tend to shrink and deform, thereby increasing the search difficulty and leading to relatively lower HV and IGD

TABLE I: Mean and standard deviation of HV and IGD obtained by the three algorithms on IDASCMOP1–IDASCMOP6

Function	HV			IGD		
	CG-based CMOEA	PF-ICMOA	CCTL-ICMOEA	CG-based CMOEA	PF-ICMOA	ICCTL-ICMOEA
IDASCMOP1	3.0749(0.2939)−	2.4360 (0.2357)−	**3.1630 (0.2596)**	0.1745(0.2594)−	0.6806 (0.2453)−	**0.0860 (0.0752)**
IDASCMOP2	3.5120(0.1411)−	3.2609 (0.0508)−	**3.6244 (0.2040)**	0.1382(0.0548)−	0.2244 (0.0161)−	**0.0700 (0.0867)**
IDASCMOP3	3.1595(0.1637)−	2.8751 (0.0394)−	**3.5368 (0.2092)**	0.2444(0.0510)−	0.2705 (0.0084)−	**0.0761 (0.0982)**
IDASCMOP4	2.2613(0.4344)+	1.9615 (0.1072)−	**2.4296 (0.7147)**	0.4699(0.1760)−	0.6717 (0.0798)−	**0.4445 (0.4240)**
IDASCMOP5	**2.4280(0.6614)**+	1.9271 (0.1742)−	2.2276 (1.0067)	**0.2233(0.2726)**+	0.9104 (0.0413)−	0.7224(0.4872)
IDASCMOP6	**2.5085(0.6016)**+	1.7445 (0.0819)+	1.5855(0.1395)	**0.5369(0.2421)**+	0.8856 (0.0232)+	0.9797(0.4902)

TABLE II: Mean and standard deviation of IM and Time obtained by the three algorithms on IDASCMOP1–IDASCMOP6

Function	IM			Time		
	CG-based CMOEA	PF-ICMOA	CCTL-ICMOEA	CG-based CMOEA	PF-ICMOA	ICCTL-ICMOEA
IDASCMOP1	0.1556(0.0075)−	**0.1431 (0.0007)**+	0.1806 (0.0115)	**13.3960(3.6926)**+	173.4012(8.5821)+	566.8008((2.1407)
IDASCMOP2	0.1606(0.0047)−	0.1072 (0.0016)=	**0.1024 (0.0394)**	**14.2374(3.268)**+	143.0097(2.1361)+	539.8706(7.3523)
IDASCMOP3	0.1641(0.0328)−	0.1509 (0.0008)−	**0.1234 (0.0258)**	**16.7498(2.5186)**+	141.4064(0.7292)+	545.9830(9.5481)
IDASCMOP4	0.1567(0.0099)−	**0.0923 (0.0006)**+	0.1434 (0.0397)	**13.5860(2.2483)**+	142.1792(1.1684)+	698.8134(5.6309)
IDASCMOP5	0.1741(0.0044)−	0.1101 (0.0016)−	**0.1067 (0.1921)**	**13.9968(4.6371)**+	152.7206(16.7155)+	717.7006(9.7992)
IDASCMOP6	0.1496(0.0111)−	0.1131(0.0017)−	**0.1065 (0.3173)**	**13.6116(6.2841)**+	181.5923(29.4932)+	663.3297(5.1553)

values. Nevertheless, since the deterministic approach cannot adequately characterize the interval properties of the original problem, its performance on the IM metric is significantly inferior.

Moreover, PF-ICMOEA performs better on IDASCMOP6. The inferior performance of ICCTL-ICMOEA on this instance is mainly attributed to the reduced effectiveness of the transfer mechanism in highly discontinuous landscapes. In IDASCMOP6, the feasible regions are extremely fragmented and disjoint, making it highly likely that solutions feasible in subproblems become infeasible in the main problem after transfer. Consequently, the transferred knowledge may fail to provide effective guidance and may even induce negative transfer, misleading the search toward deceptive infeasible regions rather than narrow feasible regions.

Second, regarding the IM metric, PF-ICMOEA achieves relatively better results on IDASCMOP1 and IDASCMOP4, indicating its effectiveness in capturing uncertainty characteristics. However, the corresponding HV values are significantly worse, implying poor convergence or diversity. Therefore, an improvement in IM alone is of limited practical significance when the overall solution quality is unsatisfactory.

Finally, in terms of computational time, the deterministic CG-based CMOEA is the most efficient, as it avoids interval-based non-dominated sorting and uncertainty-related computations. PF-ICMOEA also incurs relatively low computational cost due to its simple constraint-handling strategy. In contrast, the proposed ICCTL-ICMOEA introduces multi-population co-evolution and solution migration, and processes more candidate solutions per generation, resulting in higher computational time. However, this additional cost leads to a significant improvement in solution quality and overall performance.

Fig. 2 displays the interval front of the benchmark function obtained by PF-ICMOA and ICCTL-ICMOEA algorithms. As shown in Fig. 2, the proposed algorithm, consistently achieving solutions closer to the true Pareto front while distributing solutions more uniformly within the feasible region. Compared to PF-ICMOA, the proposed algorithm demonstrates superior diversity while ensuring solution set convergence.

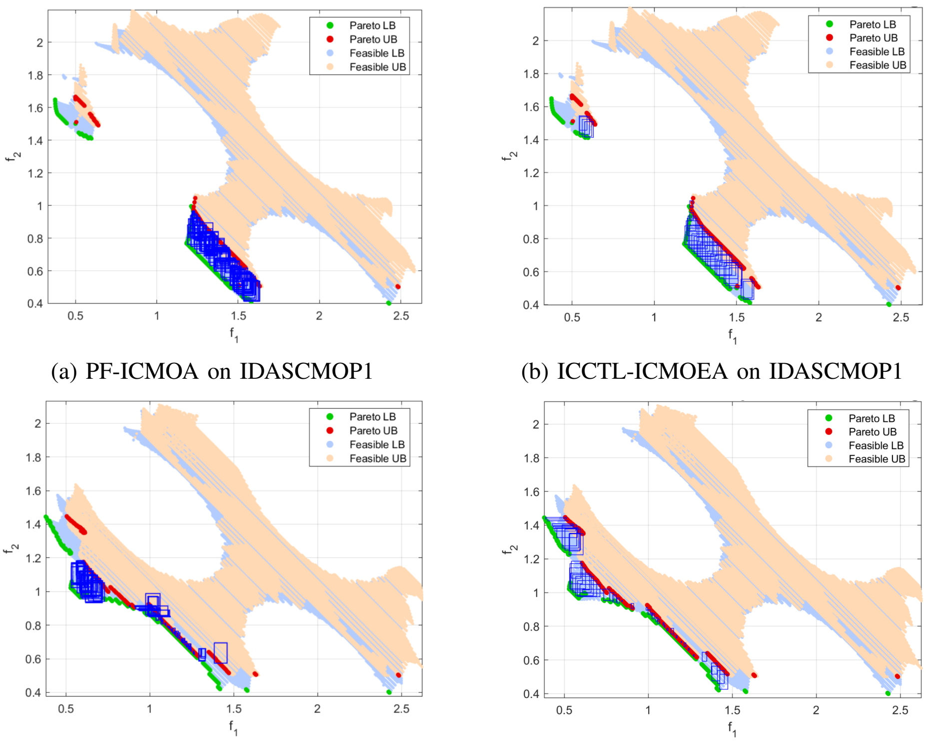
V. Conclusion

In ICMOPs, uncertainty and multiple constraints often lead to a small feasible region and a difficult search process. To tackle this issue, this paper proposes an ICCTL-ICMOEA. Constraint similarity is quantified via an interval-overlap degree, and the original problem is decomposed into multiple simplified subproblems through ICC. A knowledge-transfer strategy is then designed to share search information from subproblems to the main problem, thereby effectively coping with uncertainty and facilitating the discovery of feasible solutions.

Experimental results on the constructed interval-constrained benchmark problems demonstrate that the proposed algorithm outperforms competing algorithms in both feasible-solution search capability and convergence performance. In this paper, most parameters and the selection of migrated individuals are determined based on experience. Future work will focuses on designing adaptive migration strategies.

References

[1] J. Sun, Y. Xiong, H. Zhang and Z. Liu, "Interval multiobjective programming methods for solving multi-period portfolio



(a) PF-ICMOA on IDASCMOP1 (b) ICCTL-ICMOEA on IDASCMOP1

(c) PF-ICMOA on IDASCMOP2 (d) ICCTL-ICMOEA on IDASCMOP2

Fig. 2: Comparison of interval fronts obtained by different algorithms.

selection problems”, *Control and Decision*, vol. 35, no. 3, pp. 645-650, 2020.

[2] B. Zeng, W. Zhang, P. Hu, J. Sun and D. Gong, “Synergetic renewable generation allocation and 5G base station placement for decarbonizing development of power distribution system: A multiobjective interval evolutionary optimization approach”, *Applied Energy*, vol. 351, pp. 121831, 2023.

[3] X. Gan, J. Sun, D. Gong, D. Jia, H. Dai and Z. Zhong, “An adaptive reference vector-based interval multiobjective evolutionary algorithm”, *IEEE Transactions on Evolutionary Computation*, vol. 27, no. 5, pp. 1235-1249, 2023.

[4] Q. Yu, C. Yang, G. Dai and L. Peng, “Non-probabilistic set-based selection strategy for multiobjective optimization with interval uncertainties”, *Swarm and Evolutionary Computation*, vol. 98, pp. 102153, 2025.

[5] Q. Yu, C. Yang, G. Dai, L. Peng and J. Li, “A novel penalty function-based interval constrained multiobjective optimization algorithm for uncertain problems”, *Swarm and Evolutionary Computation*, vol. 88, pp. 101584, 2024.

[6] 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.

[7] 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.

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

[9] 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.

[10] J. Zou, R. Sun, Y. Liu, Y. Hu, S. Yang, J. Zheng and K. Li, “A multipopulation evolutionary algorithm using new cooperative mechanism for solving multiobjective problems with multiconstraint”, *IEEE Transactions on Evolutionary Computation*, vol. 28, no. 1, pp. 267-280, 2024.

[11] K. Qiao, J. Liang, K. Yu, X. Ban, C. Yue, B. Qu and P. N. Suganthan, “Constraints separation based evolutionary multitasking for constrained multiobjective optimization problems”, *IEEE/CAA Journal of Automatica Sinica*, vol. 11, no. 8, pp. 1819-1835, 2024.

[12] B. Xu, Y. Zheng, W. Li, X. Gao, D. Gong, J. He and Z. Fan, “Handling multiobjective optimization problems with complex constraints: A constraints grouping-based approach”, *IEEE Transactions on Evolutionary Computation*, vol. 55, no. 6, pp. 3866-3880, 2025.

[13] F. Wang, J. Sun and H. Dai, “An interval constraint violation degree guided interval constrained multiobjective evolutionary algorithm”, *Control and Decision*, vol. 39, no. 12, pp. 4083-4092, 2024.

[14] V. D. Blondel, J. Guillaume, R. Lambiotte and E. Lefebvre, “Fast unfolding of communities in large networks”, *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, pp. P10008, 2008.

[15] 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.

[16] 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.

[17] D. Gong, J. Sun and Z. Miao, “A set-based genetic algorithm for interval many-objective optimization problems”, *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 47-60, 2018.