

Feasible Regions Identification based on Historical Solutions for Constrained Optimization Problems

Mengli Shan^{1,4,5}, Changhe Li^{2,3}, Mai Peng^{1,4,5}, Xiaobo Liu^{1,4,5}, Michalis Mavrovouniotis^{6,7}, Shengxiang Yang⁸

¹School of Automation, China University of Geosciences, Wuhan, 430074, China

²School of Artificial Intelligence, Anhui University of Science & Technology, Hefei, 231131, China

³State Key Laboratory of Digital Intelligent Technology for Unmanned Coal Mining, Anhui University of Science & Technology, Huainan, 232001, China

⁴Hubei Key Laboratory of Advanced Control and Intelligent Automation for Complex Systems, Wuhan, 430074, China

⁵Engineering Research Center of Intelligent Technology for Geo-Exploration, Ministry of Education, Wuhan, 430074, China

⁶ERATOSTHENES Centre of Excellence, Limassol, Cyprus

⁷Department of Civil Engineering and Geomatics, Cyprus University of Technology, Limassol, Cyprus

⁸School of Computer Science and Informatics, De Montfort University, Leicester, LE1 9BH, United Kingdom

Email: shanmengli@cug.edu.cn, changhe.li@gmail.com (Corresponding author: Changhe Li)

Abstract—The presence of constraints often leads to the formation of narrow and fragmented feasible regions within the search region, presenting significant challenges for optimization problem-solving. This paper introduces a novel approach, Feasible Regions Identification based on Historical Solutions (FRIHS), designed to address these challenges. FRIHS leverages previously evaluated solutions to partition the search region into ε -feasible and ε -infeasible regions. Additionally, by analyzing the correlations among constraints, they are reformulated as auxiliary objectives, effectively transforming the constrained optimization problem into a constrained multi-objective optimization problem. The method employs the classical evolutionary algorithm Differential Evolution and the multi-objective method NSGA-III to search the most promising feasible regions. The effectiveness of FRIHS is evaluated through a comparative analysis with five advanced constraint-handling algorithms across a benchmark test suite. Experimental results indicate that the proposed approach demonstrates competitive performance on the test problems.

Index Terms—constrained optimization problem, multiobjective optimization, region partitioning.

I. INTRODUCTION

Constrained optimization problems (COPs) are widely present in social production and daily life. A constrained optimization problem involves finding the optimal solution to an objective function within the bounds of constraints. The constraints usually include upper and lower bounds on decision variables, as well as equality and inequality constraints. Generally, a COP in the continuous space is mathematically described as

$$\begin{aligned} \min \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{g}(\mathbf{x}) = (g_1(\mathbf{x}), g_2(\mathbf{x}), \dots, g_p(\mathbf{x})) \leq \mathbf{0} \\ & \mathbf{h}(\mathbf{x}) = (h_{p+1}(\mathbf{x}), \dots, h_m(\mathbf{x})) = \mathbf{0} \\ & \mathbf{x} = (x_1, \dots, x_D) \in \mathbb{X} \end{aligned} \quad (1)$$

where $f(\mathbf{x})$ denotes the objective function; $\mathbf{x}$ represents the decision vector; $\mathbf{g}(\mathbf{x})$ is the vector of inequality constraints, and $\mathbf{h}(\mathbf{x})$ is the vector of equality constraints; p and $m - p$ denote the number of inequality and equality constraints. $\mathbb{X} = \{\mathbf{x} \mid l_d \leq x_d \leq u_d, d = 1, \dots, D\}$ describes the continuous search region, in which u_d and l_d is the upper and lower bounds of x_d , respectively. The violation of a constraint is usually calculated by

$$G_i(\mathbf{x}) = \begin{cases} \max\{g_i(\mathbf{x}), 0\} & , 1 \leq i \leq p \\ \max\{|h_i(\mathbf{x})| - \delta, 0\} & , p + 1 \leq i \leq m \end{cases} \quad (2)$$

where δ is the tolerate factor, and its value is always set to 0.0001.

The existence of constraints partitions the search region into feasible and infeasible regions. When constraints involve characteristics such as equalities, non-convexity, or multimodality, several challenges emerge, including the presence of extremely narrow feasible regions, multiple discrete feasible regions, and optimal solutions located on the boundary of the feasible region. In order to solve constrained continuous optimization problems, many researchers have developed a variety of constraint handling techniques (CHTs). These include penalty functions [1], [2], feasibility rules [3], stochastic ranking [4], ε -constrained methods [5], multi-objective methods [6], [7], and ensemble methods. These techniques are primarily applied during the individual selection phase. It is widely recognized that distinct CHTs are better suited for specific types of optimization problems, and even varying stages of the optimization process may necessitate different constraint handling strategies [8].

In response to these challenges, a novel framework is proposed for constrained continuous optimization problems characterized by complex feasible regions, such as those that are narrow or discrete. This framework identifies feasible regions by partitioning and aggregating sub-regions using

historical search information. By leveraging historical search information, the framework dynamically adapts to the structure of the feasible region, guiding the search towards promising areas. Additionally, to enhance search capability and increase population diversity, a method is introduced that groups constraints based on their correlations and transforms them into auxiliary objectives. This transformation effectively converts the COP into a constrained multi-objective optimization problem (CMOP).

The rest of the paper is organized as follows. Sec. II introduces six types of CHTs and their respective strengths and weaknesses are evaluated. Then, Sec. III describes the proposed method Feasible Regions Identification based on Historical Solutions (FRIHS). Sec. IV presents the experimental studies. Finally, Sec. V summarizes this study and provides future works.

II. RELATED WORKS

In constrained continuous optimization algorithms, Evolutionary algorithms (EAs) are tasked with generating new solutions, while CHTs are responsible for comparing and selecting individuals. The earliest and most widely adopted approach, the penalty function method, transforms a COP into an unconstrained optimization problem by introducing a penalty factor [9], [10]. The efficacy of this method in solving various problems is highly sensitive to the appropriate selection of the penalty factor. An excessively large or overly small penalty factor can impose unduly stringent penalties on infeasible solutions or impede the population's convergence toward feasible regions, thereby affecting the algorithm's overall performance. Consequently, numerous adaptive penalty factor strategies have been proposed in recent years to address these challenges [11], [12].

The feasibility rules, introduced by Deb [13], asserts that feasible solutions are always prioritized over infeasible ones. Due to its inherent tendency to prioritize constraint satisfaction, the population is highly prone to becoming confined within a local feasible region. Despite this limitation, the method's simplicity and ease of implementation have made it a popular choice among researchers. Many have successfully combined it with various diversity maintenance techniques to mitigate its shortcomings and achieve improved optimization performance [14], [15]. On the other hand, the stochastic ranking method, proposed by Runarsson and Yao [16], employs a probabilistic mechanism to determine whether to compare individuals based on their constraint violations or their objective function values. However, this probabilistic design introduces its own challenge: the need to carefully tune the probability, which can significantly impact the algorithm's performance [17].

The ε -constrained method [18] relaxes the constraint boundaries by adjusting the parameter ε , which plays a pivotal role in balancing the trade-off between the constraints and the objective function. Consequently, numerous strategies for adjusting ε have been proposed. The most widely adopted approach, introduced by Takahama and Sakai [19], involves

gradually decreasing the value of ε as the evolutionary generations progress. Furthermore, adaptive strategies have been developed that adjust ε based on factors such as the proportion of feasible solutions within the population [20].

Multi-objective methods aim to transform the constraints of a COP into objectives to be optimized. Generally, two main approaches are used: one aggregates all constraints into a single objective [21], [22], while the other treats each constraint as an individual objective [23]. The latter approach can result in a more complex many-objective optimization problem, which is why it is infrequently employed. In contrast, the former approach typically converts COPs into a multi-objective optimization problem with two objectives, the original objective and constraint violation, which can subsequently be addressed using multi-objective optimization techniques [24], [25].

Given the inherent limitations of each CHT in solving COPs, hybrid approaches that integrate multiple CHTs have been developed. Wen et al. [26] introduced a voting mechanism, where fixed weight values are assigned to each CHT, with the selection process being probabilistic. Building on this framework, Wu et al. [8] proposed an adaptive strategy that dynamically adjusts the weights based on the problem context. However, effectively combining various CHTs to leverage their strengths while mitigating their weaknesses remains a significant challenge [27].

Although these CHTs can achieve satisfactory results for certain problems through parameter tuning or strategy design, different CHTs are typically only suitable for specific types of problems. Even within different stages of solving the same problem, varying CHTs may be required. The root cause of this issue lies in the complex feasible region structures formed by the constraints. Faced with different feasible region distribution landscapes, distinct selection strategies are necessary. For instance, when a problem has only a very narrow feasible region, priority should be given to satisfying the constraints; whereas when a problem contains multiple discrete feasible regions, maintaining population diversity is essential to prevent the population from becoming trapped in local feasible regions.

Therefore, if the distribution information of feasible and infeasible regions formed by the constraints can be obtained to provide a clear direction for the search process, it will help reduce reliance on specific CHTs and enhance the generality and robustness of the algorithm. To this end, this paper proposes constructing a distribution model of feasible and infeasible regions in the decision space using the constraint violation information of historical solutions.

III. FEASIBLE REGIONS IDENTIFICATION BASED ON HISTORICAL SOLUTIONS

A. Overall Framework

The proposed method aims to iteratively identify ε -feasible and ε -infeasible regions within the search region and evolve within promising regions to find the optimal solution. The framework of the proposed FRIHS is shown in Alg. 1. The algorithm primarily comprises three components: transforming

the original COP into two CMOPs, identifying feasible and infeasible regions, and searching promising regions. It is important to clarify that if the individual with the worst violation in a sub-region is ε -feasible, the entire sub-region is considered ε -feasible. Moreover, multiple adjacent ε -feasible sub-regions can collectively form an ε -feasible region.

Algorithm 1: FRIHS($\mathbb{X}$, N_1 , N_2)

Input : search region $\mathbb{X}$, size of exploit solutions N_1 , size of explore solutions N_2
Output : solution x_n
Initialize: sub-regions $\mathcal{O} \leftarrow \{\mathbb{X}\}$, ε -feasible regions $\mathbb{A} \leftarrow \{\mathbb{X}\}$, ε -infeasible regions $\mathbb{B} \leftarrow \emptyset$, evaluated solutions $\mathcal{S} \leftarrow \emptyset$, subpopulations $\mathcal{P} \leftarrow \emptyset$;

```

1 initializePopulation();
2 classifyConstraints();
3 while termination conditions not reached do
4   if  $k \leq K$  then
5     reduce  $\varepsilon$ ;
6      $\mathcal{O} \leftarrow \text{divideSubregions}(\mathcal{O})$ ;
7     updateSubregionTag( $\varepsilon$ );
8      $\mathbb{A}, \mathbb{B} \leftarrow \text{groupSubregions}(\mathcal{O})$ ;
9      $\mathcal{S}, \mathcal{P} \leftarrow \text{classifyPopulations}(\mathcal{S}, \mathcal{P}, \mathbb{A}, \mathbb{B})$ ;
10     $pt_1, pt_2, pt_3 \leftarrow \text{updatePotential}(\mathbb{A}, \mathbb{B})$ ;
11     $a_1, a_2, b \leftarrow \text{rouletteWheelSelect}(\mathbb{A}, \mathbb{B}, pt_1, pt_2, pt_3)$ ;
12     $P_1, P_2, P_3 \leftarrow \text{subpopulation located in } a_1, a_2, b \text{ among } \mathcal{P}$ ;
13     $\mathcal{S} \leftarrow \text{evolveWithinRegions}(P_1, P_2, P_3)$ ;
14     $k++$ ;
15  end
16   $a_3 \leftarrow \text{minObjective}(\mathbb{A})$ ;
17   $P_3 \leftarrow \text{subpopulation located in } a_3 \text{ among } \mathcal{P}$ ;
18   $\mathcal{S} \leftarrow \text{evolve}(P_3)$ ;
19 end

```

In the initialization phase, the framework first sets the ε -feasible regions $\mathbb{A}$ and sub-regions $\mathcal{O}$ to be the entire search region $\mathbb{X}$, and the ε -infeasible region $\mathbb{B}$ is initialized as an empty set. At the same time, an exploitation population and an exploration population are generated in the search region, with sizes of N_1 and N_2 , respectively. As is well known, multi-objective methods are increasingly being applied to COPs. To enhance the diversity of the population and improve its search ability, the Spearman's rank correlation coefficient is used to classify the constraints. Constraints of the same type are combined by addition or subtraction to form an additional optimization objective, thus transforming the original COP into a CMOP.

In the main loop, to address scenarios where the feasible region is extremely narrow, the ε -constrained method is employed to relax the constraint boundaries. The boundaries are initially scaled using the parameter ε , ensuring that a majority of solutions are ε -feasible at the beginning of the evolution. As

the search progresses, the value of ε is gradually reduced. This approach allows the violation information of the solutions to progressively guide the population toward convergence to the feasible regions, while simultaneously refining the description of the boundary of the feasible regions. Considering that if ε decays too quickly, the problem will quickly revert to the original one, failing to achieve the effect of scaling the constraint boundaries; if it decays too slowly, the convergence efficiency of the population will be affected. To ensure a steady identification of the feasible regions during the search process, a linear decay approach is adopted for ε , which is calculated by

$$\varepsilon_{i,k} = \varepsilon_{i,0} - \frac{k}{K} \cdot \varepsilon_{i,0} \quad (3)$$

$$1 \leq i \leq m, 0 \leq k \leq K$$

where $\varepsilon_{i,k}$ denotes the ε boundary of the i th constraint at the k th iteration, K is the maximum number of iterations allowed for identifying feasible regions. In order to make the initial population ε feasible, the maximum violation of the i th constraint in initial exploit population is selected as the initial constraint boundary $\varepsilon_{i,0}$.

At each iteration, the decision whether to subdivide a sub-region is based on the historical solutions it holds. If a sub-region contains both feasible and infeasible solutions, it will be subdivided. After subdividing the sub-regions, the ε feasible status of each sub-region is updated according to the value of ε_k . Adjacent ε feasible sub-regions are grouped into an ε -feasible region, while adjacent ε infeasible sub-regions are grouped into an ε -infeasible region. Based on the relative positions of all individuals in the subpopulations with respect to the different ε -feasible and ε -infeasible regions in the decision space, the subpopulations are re-divided. Individuals within the same region are grouped into the same subpopulation.

During this process, each ε -feasible region is ensured to comprise one exploitation subpopulation and one exploration subpopulation, while each ε -infeasible region contains a single exploration subpopulation. Subsequently, based on the feasibility status of each domain, the three most promising populations are identified and allocated distinct search tasks, thereby generating new search data.

B. Conversion of COP to CMOP

The effectiveness of multi-objective optimization methods in solving constrained optimization problems is well-established. To enhance the exploration capability of the population while minimizing the loss of information embedded in each constraint, this paper analyzes the relationships between the constraints using the Spearman rank correlation coefficient. The correlation coefficient between two constraints, G_a and G_b ($1 \leq a, b \leq m$), is denoted as $\rho_{a,b}$, and its calculation is given by Eq. (4).

$$\rho_{a,b} = 1 - \frac{6 \sum_{j=1}^N (\text{rank}(G_a(x_j)) - \text{rank}(G_b(x_j)))^2}{N(N^2) - 1} \quad (4)$$

If the correlation coefficient $\rho_{a,b}$ between any two constraints G_a and G_b satisfies $|\rho_{a,b}| \geq 0.7$, then G_a and G_b are merged in this manner

$$\begin{cases} G_a(\mathbf{x}) + G_b(\mathbf{x}), & \text{if } \rho_{a,b} \geq 0.7 \\ G_a(\mathbf{x}) - G_b(\mathbf{x}), & \text{if } \rho_{a,b} \leq -0.7 \end{cases} \quad (5)$$

After grouping and merging all the constraints, the m constraints are transformed into q additional optimization objectives, resulting in a CMOP, as shown in Eq. (6). This is the problem that P_1 is tasked with solving, which aims to optimize the original objective function while satisfying the constraints. The tasks for P_2 and P_3 are only related to constraints satisfaction, so the problems they need to solve do not include the original optimization objectives, as shown in Eq. (7).

$$\begin{aligned} \min \quad & (f(\mathbf{x}), v_1(\mathbf{x}), \dots, v_q(\mathbf{x})) \\ \text{s.t.} \quad & G_i(\mathbf{x}) \leq \varepsilon_{i,k}, 1 \leq i \leq m \end{aligned} \quad (6)$$

$$\begin{aligned} \min \quad & (v_1(\mathbf{x}), \dots, v_q(\mathbf{x})) \\ \text{s.t.} \quad & G_i(\mathbf{x}) \leq \varepsilon_{i,k}, 1 \leq i \leq m \end{aligned} \quad (7)$$

In this context, $v_l(\mathbf{x})$ represents the l -th violation objective obtained by transforming the constraint violation of the l -th group.

C. Update Feasible Regions by Historical Solutions

The process of identifying feasible regions involves two primary steps. First, the search region is partitioned into sub-regions. Second, these sub-regions are aggregated to delineate distinct ε -feasible and ε -infeasible regions. To address how to partition the sub-regions, the first consensus to be reached is that there must be a boundary between feasible and infeasible solutions. Based on this, the partitioning process is demonstrated in Alg. 2. When the worst violation objective individual $\mathbf{x}_\Omega^\dagger$ in the ε -feasible sub-region Ω is infeasible, it indicates the presence of infeasible segments within this sub-region and thus needs to be partitioned. Similarly, when the best violation objective individual $\mathbf{x}_\Omega^*$ in the ε -infeasible sub-region Ω is feasible, it suggests that there are feasible parts within this sub-region, which also requires further partitioning. To improve the efficiency of partitioning, this paper adopts a k -d space partitioning tree to organize the sub-regions and accelerate the indexing process.

Once the sub-regions are identified, they are aggregated into ε -feasible and ε -infeasible regions based on their relative positions and ε -feasibility, as shown in Alg. 3. Specifically, the ε -feasibility of each sub-region is determined by its historical solutions and the value of ε_k . For an ε -feasible sub-region, it is initially considered as a center and then expanded outward to identify all adjacent ε -feasible sub-regions, which are aggregated into a single ε -feasible region. Similarly, for an ε -infeasible sub-region, it serves as a center of an ε -infeasible region, with the outward expansion identifying all neighboring ε -infeasible sub-regions, which are then combined into a distinct ε -infeasible region.

Algorithm 2: divideSubregions($\mathcal{O}$)

Input : sub-regions $\mathcal{O}$
Output: sub-regions $\mathcal{O}$

```

1 for  $\Omega \in \mathcal{O}$  do
2   if  $\Omega \in \mathbb{A}$  then
3     if  $\mathbf{x}_\Omega^\dagger$  is infeasible solution then
4       halve  $\Omega$ ;
5     end
6   end
7   if  $\Omega \in \mathbb{B}$  then
8     if  $\mathbf{x}_\Omega^*$  is feasible solution then
9       halve  $\Omega$ ;
10    end
11  end
12 end
```

Algorithm 3: groupSubregions($\mathcal{O}$)

Input : sub-regions $\mathcal{O}$
Output: ε -feasible regions $\mathbb{A}$, ε -infeasible regions $\mathbb{B}$

```

1  $\mathbb{A} \leftarrow \emptyset, \mathbb{B} \leftarrow \emptyset$ ;
2 Mark the  $\varepsilon$ -feasibility status of  $\mathcal{O}$  according to  $S$  and  $\varepsilon_k$ ;
3 for  $\Omega \in \mathcal{O}$  do
4   if  $\Omega$  is  $\varepsilon$ -feasible then
5      $\alpha \leftarrow \{\Omega\}$ ;
6      $\alpha' \leftarrow \text{findFeasibleNeighbors}(\Omega)$ ;
7      $\alpha \leftarrow \alpha \cup \alpha', \mathbb{A} \leftarrow \mathbb{A} \cup \{\alpha\}$ ;
8   end
9   if  $\Omega$  is  $\varepsilon$ -infeasible then
10     $\beta \leftarrow \{\Omega\}$ ;
11     $\beta' \leftarrow \text{findInfeasibleNeighbors}(\Omega)$ ;
12     $\beta \leftarrow \beta \cup \beta', \mathbb{B} \leftarrow \mathbb{B} \cup \{\beta\}$ ;
13  end
14 end
```

Fig. 1 clearly illustrates the process of identifying feasible regions. Each sub-region is marked by two historical individuals, one being the best and the other the worst in terms of violation objective. Red points represent infeasible individuals, and green points represent feasible individuals. The red rectangles denote ε -infeasible sub-regions, while the green rectangles denote ε -feasible sub-regions. All adjacent green rectangles form an ε -feasible region, while all adjacent red rectangles form an ε -infeasible region.

It is important to note that if a sub-region contains no historical individuals, it is marked as a ε -feasible sub-region. The exploration and exploitation mechanisms of the ε -feasible regions account for sampling frequency and population coverage, thereby reducing the likelihood of sub-regions without historical individuals appearing.

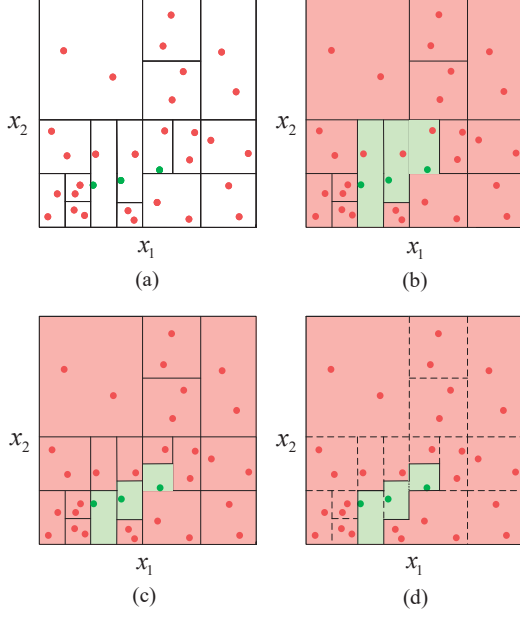


Fig. 1. Process of updating feasible regions by historical solutions. (a) sub-regions. (b) ε feasibility of sub-regions. (c) Halve sub-regions. (d) Group sub-regions.

D. Search within Regions

After dividing the search region into ε -feasible and ε -infeasible regions, the next step is to define the search tasks. Based on this division, three tasks are identified: (1) exploiting the ε -feasible region to further improve the objective values, (2) exploring the ε -infeasible region to identify potential ε -feasible regions, and (3) accurately characterizing the boundaries between the ε -feasible and ε -infeasible regions. Once the tasks are determined, the corresponding attribute metrics for each region are considered, as shown in Table I.

TABLE I
ATTRIBUTES OF THE REGIONS CORRESPONDING TO EACH SEARCH TASK.

Tasks	Attributes of $\mathbb{A}/\mathbb{B}$			Subpopulations
task1	f_a	$\frac{S_a}{C_a}$	$\frac{L_a^D}{C_a}$	P_1
task2	$cv_a^\dagger$	$\frac{S_a}{C_a}$	$\frac{L_a^D}{C_a}$	P_2
task3	cv_b^*	$\frac{S_b}{C_b}$	$\frac{L_b^D}{C_b}$	P_3

For task 1, the metrics include the objective value (f_a) of the ε -feasible region, the ratio of the number of historical solutions (S_a) and the maximum solution distance within the region (L_a^D) to region volume (C_a), which represent the sampling degree and population coverage, respectively. Task 2 focuses on refining the boundary, using the worst constraint violation ($cv_a^\dagger$), the ratio of region volume to the number of historical solutions, and the ratio of volume to maximum solution distance. Task 3 aims to explore ε -infeasible regions, considering the best constraint violation (cv_b^*), the ratio of

the number of historical solutions to volume, and the ratio of the maximum solution distance to volume. Here, the best and worst constraint violation values are determined as the minimum and maximum constraint violation values, respectively, among all solutions within the current region. The constraint violation of a solution is calculated as the average of the normalized violations across all constraints for that solution:

$$cv = \frac{1}{m} \sum_{i=1}^m \frac{G_i(\mathbf{x})}{\max_{\mathbf{x} \in P(0)} \{G_i(\mathbf{x})\}} \quad (8)$$

where $P(0)$ refers to the initial population. The regions are ranked based on these metrics, and a roulette wheel selection method is used to choose the regions for each task.

Algorithm 4: evolveWithinRegions

Input : evaluated solutions S , subpopulation P_1, P_2, P_3 , region a_1, a_2, b
Output : evaluated solutions S , subpopulation P_1, P_2, P_3

```

1  $Q \leftarrow P_1/P_2/P_3$ ;
2 for  $x \in P_1/P_2/P_3$  do
3   generateReferencePoints();
4   while  $x'' \notin a_1/\mathbb{X}/b$  do
5      $x' \leftarrow \text{mutate}(x, P_1/P_2/P_3)$ ;
6      $x'' \leftarrow \text{crossover}(x, x')$ ;
7   end
8    $S \leftarrow S \cup \{\text{evaluated}(x'')\}$ ;
9    $Q \leftarrow Q \cup \{\text{evaluated}(x'')\}$ ;
10 end
11 nonDominatedSorting( $Q$ );
12 calculateCrowdingDistance( $Q$ );
13 sort( $Q$ );
14  $P_1/P_2/P_3 \leftarrow \text{selectNextPopulation}(Q)$ ;

```

Once the regions to search are determined, subpopulations within the regions employ a combination of the classical DE/best/1 [28] and NSGA-III [29], as shown in the Alg. 4. It is important to emphasize that during the mutation step with the DE/best/1 operator, both P_1 and P_3 select the best individuals from their own populations, while P_2 selects the best individuals from all ε -infeasible sub-regions adjacent to the current ε -feasible region. Furthermore, the trial vector x'' generated by P_1 and P_3 are required to lie within the corresponding ε -feasible region a_1 and ε -infeasible region b , while the trial vector x'' generated by P_2 are not subject to this constraint.

At the end of each iteration, the framework selects the ε -feasible region with the minimum objective value, denoted as a_3 , for further evolution. Once the termination conditions are satisfied, the framework outputs the optimal solution x_n . Overall, FRIHS achieves a notable balance between exploration and exploitation while making an initial effort to delineate feasible and infeasible regions.

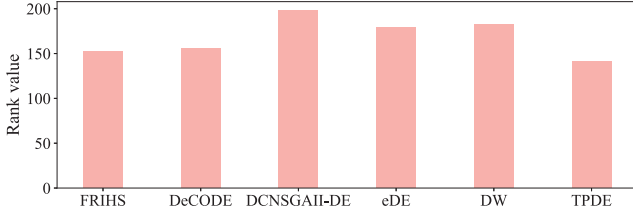


Fig. 2. Ranks of the methods on problems of 10 dimensions based on mean value and median solution.

IV. EXPERIMENTAL STUDIES

A. Parameter Settings of FRIHS

Parameter settings of FRIHS are as follows.

- (1) The number of runs: 25.
- (2) Max FEs = 200,000.
- (3) Max iteration for identifying feasible regions: $K = 1,000$.
- (4) The size of exploit solutions $N_1 = 20$ and the size of explore solutions $N_2 = 5$.
- (5) The settings of the DE operator in Alg. 4: scaling factor $F = 0.7$, crossover probability $CR = 0.9$.

B. Experiments on CEC2017 Benchmark Test Functions

To evaluate the performance of the FRIHS in solving COPs, the proposed algorithm was compared with five state-of-the-art algorithms on the IEEE CEC 2017 (D=10) [30]. The comparison involved DeCODE [31], DCNSGAI-DE [21], adaptive ϵ DE [32], DW [33], and TPDE [1]. The algorithms were ranked based on the evaluation criteria provided in the test set. The ranking results based on mean values are presented in Table II, while the rankings derived from median solutions are shown in Table III. The algorithm achieving the top rank for solving each problem is highlighted in bold. The total rank values combining mean values and median solutions are illustrated in Fig. 2.

As observed in Table II, FRIHS demonstrates strong performance by achieving top rankings (rank < 3) in 17 problems, indicating its ability to find better solutions for the majority of the tested cases. However, the results in Table III reveal that FRIHS achieves similar high rankings in only 9 problems, suggesting that the stability of the proposed method requires further improvement. From the total ranking illustrated in Fig. 2, it can be seen that the comprehensive performance of FRIHS is inferior to TPDE, slightly better than DeCODE, and significantly superior to DCNSGAI-DE, adaptive ϵ DE, and DW.

C. Investigation of the Regions Identification

This chapter conducts more comprehensive experiments to further validate the effectiveness of FRIHS in feasible regions identification. To visually demonstrate the performance of FRIHS, the process of feasible regions identification is illustrated using three test problems, ATF1, ATF2, and ATF3, proposed in [34]. Specifically, ATF1 contains an extremely narrow linear feasible region, as shown in Fig. 3 (a); ATF2

includes two discrete feasible regions, as shown in Fig. 3 (b); and ATF3 features a single feasible region, where the optimal solution lies on the boundary of the feasible region, as shown in Fig. 3 (c).

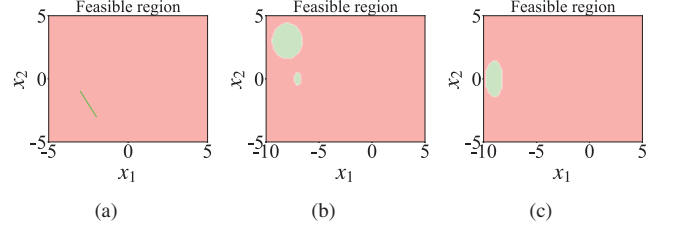


Fig. 3. Visualization of the decision space, the feasible region of tree constrained optimization problems.

1) *For problems with narrow feasible region:* Fig. 4 illustrates three generations of feasible regions identification by FRIHS during a single run on ATF1. In Fig. 4 (a), most of the sub-regions are identified as ϵ -feasible. As the search progresses, the ϵ -feasible region gradually shrinks. By the point when FEs = 14129, the feasible region defined by the original constraints has been roughly identified. Although this process provides a coarse depiction of the extremely narrow, line-segment-shaped feasible region, the ideal outcome would be to restore the shape of the line segment as accurately as possible.

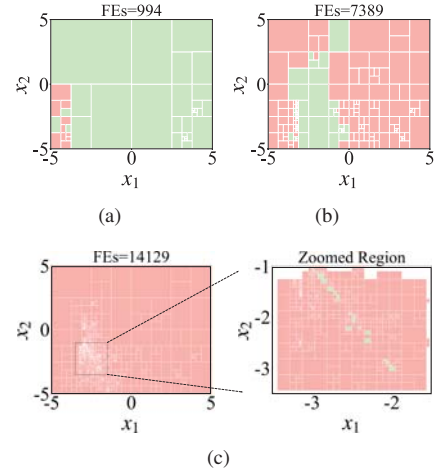


Fig. 4. A single-run visualization of the FRIHS on instance ATF1 instance.

2) *For problems with disjoint feasible regions:* Fig. 5 presents three generations of feasible regions identification by FRIHS during a single run on ATF2. In Fig. 5 (a), the boundary of the ϵ -feasible region begins to contract toward the original constraint boundary. Fig. 5 (b) shows further contraction, while Fig. 5 (c) demonstrates that FRIHS achieves effective identification even for problems with discrete feasible regions.

3) *For problems with Boundary Optimal Solutions:* For problems where the optimal solution lies on the boundary of

TABLE II
RANKS BASED ON MEAN VALUES ON THE 28 FUNCTIONS OF 10 DIMENSIONS.

Algorithm	C01	C02	C03	C04	C05	C06	C07	C08	C09	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	C20	C21	C22	C23	C24	C25	C26	C27	C28
FRIHS	1	1	1	1	2	1	2	4	4	2	4	2	2	2	2	1	5	4	4	6	2	2	6	2	6	5	4	1
DeCODE	2	2	2	6	4	2	6	2	2	4	2	6	4	6	5	6	3	3	6	2	4	5	4	5	2	3	3	6
DCNSGAI-DE	4	4	6	2	5	6	3	1	3	3	3	4	6	3	3	2	6	6	3	1	6	4	2	4	5	6	6	5
eDE	3	5	5	3	1	3	5	6	1	6	6	5	5	4	6	5	4	5	5	3	1	3	5	6	1	4	1	3
DW	5	6	3	5	6	5	1	3	5	1	5	3	3	5	4	3	2	1	1	5	5	6	3	1	4	1	5	4
TPDE	6	3	4	4	3	4	4	5	6	5	1	1	1	1	1	4	1	2	2	4	3	1	1	3	3	2	2	2

TABLE III
RANKS BASED ON MEDIAN SOLUTION ON THE 28 FUNCTIONS OF 10 DIMENSIONS.

Algorithm	C01	C02	C03	C04	C05	C06	C07	C08	C09	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	C20	C21	C22	C23	C24	C25	C26	C27	C28
FRIHS	1	1	1	1	4	1	5	3	4	4	2	2	6	6	6	1	6	5	5	3	4	6	6	5	4	4	4	1
DeCODE	2	2	2	3	1	2	1	2	2	1	6	1	2	1	1	3	5	6	6	2	1	1	3	1	2	6	6	6
DCNSGAI-DE	4	6	5	4	6	4	3	5	3	3	3	5	5	3	3	5	2	2	3	4	6	5	5	6	6	2	2	4
eDE	3	3	6	6	3	6	6	1	1	2	1	3	3	4	5	6	4	1	1	6	2	3	2	3	5	5	1	5
DW	5	4	4	5	2	5	4	6	5	6	5	4	4	5	2	4	1	4	4	5	5	2	4	2	3	1	5	3
TPDE	6	5	3	2	5	3	2	4	6	5	4	6	1	2	4	2	3	3	2	1	3	4	1	4	1	3	3	2

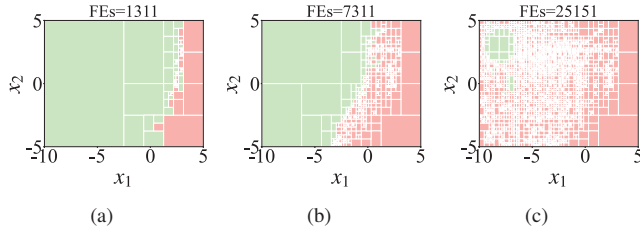


Fig. 5. A single-run visualization of the FRIHS on instance ATF2 instance.

the feasible region, precise characterization of the boundary is critical. Fig. 6 depicts three generations of feasible region identification by FRIHS during a single run on ATF3. As the search progresses, it can be observed that FRIHS successfully characterizes the feasible region and achieves a satisfactory identification result.

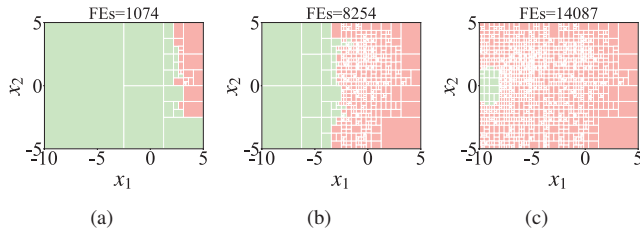


Fig. 6. A single-run visualization of the FRIHS on instance ATF3 instance.

V. CONCLUSIONS

This paper addresses the challenges of constrained optimization problems, particularly the narrow and discrete feasible regions caused by constraints, by proposing a feasible regions identification method based on historical solutions. This method progressively subdivides the search region using historical solution information and clusters sub-regions into ε -feasible or ε -infeasible regions based on their states. Furthermore, to maintain population diversity and enhance

search capability, a problem transformation mechanism based on Spearman rank correlation analysis is introduced. Experimental results demonstrate that the proposed method is competitive in solving COPs and has shown the ability to effectively identify narrow and discrete feasible regions in two-dimensional problems.

However, it must be acknowledged that the method faces substantial computational challenges when dealing with high-dimensional problems, rendering its performance significantly limited in such scenarios. Additionally, the method still requires considerable improvement in handling extremely narrow feasible regions shaped by complex constraints, such as equality constraints. Addressing these limitations will be the primary focus of our future research endeavors.

ACKNOWLEDGMENT

This work was supported in part by the National Natural Science Foundation of China under Grant 62476006, in part by the Hubei Provincial Natural Science Foundation of China under Grant 2023AFA049, in part by the Fundamental Research Funds of the AUST under Grant 2024JBZD0007, and in part by the SmartPLAIGO: Advancing Public Health services through AI driven GIS optimization, funded by the European Union Recovery and Resilience Facility of the NextGenerationEU instrument through the Research and Innovation Foundation under the Grant Agreement No CODEVELOP-AG-SH-HE/0823/0112.

REFERENCES

- [1] 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*, vol. 53, no. 10, pp. 6194–6203, 2023.
- [2] B. Wang, H. Li, and Y. Feng, "An improved teaching-learning-based optimization for constrained evolutionary optimization," *Information Sciences*, vol. 456, pp. 131–144, 2018.
- [3] B. Xu, X. Chen, and L. Tao, "Differential evolution with adaptive trial vector generation strategy and cluster-replacement-based feasibility rule for constrained optimization," *Information Sciences*, vol. 435, pp. 240–262, 2018.

- [4] U. Balade and D. Shrimankar, "Srifa: Stochastic ranking with improved-firefly-algorithm for constrained optimization engineering design problems," *Mathematics*, vol. 7, no. 3, p. 250, 2019.
- [5] A. Kumar, S. Das, and V. Snášel, "Improved spherical search with local distribution induced self-adaptation for hard non-convex optimization with and without constraints," *Information Sciences*, vol. 615, pp. 604–637, 2022.
- [6] 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, 2020.
- [7] Y. Zhou, M. Zhu, J. Wang, Z. Zhang, Y. Xiang, and J. Zhang, "Tri-goal evolution framework for constrained many-objective optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 8, pp. 3086–3099, 2018.
- [8] G. Wu, X. Wen, L. Wang, W. Pedrycz, and P. N. Suganthan, "A voting-mechanism-based ensemble framework for constraint handling techniques," *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 4, pp. 646–660, 2021.
- [9] A. Kumar, S. Das, A. K. Misra, and D. Singh, "A v -constrained matrix adaptation evolution strategy with broyden-based mutation for constrained optimization," *IEEE Transactions on Cybernetics*, vol. 52, no. 6, pp. 4784–4796, 2021.
- [10] 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, 2021.
- [11] C. Saha, S. Das, K. Pal, and S. Mukherjee, "A fuzzy rule-based penalty function approach for constrained evolutionary optimization," *IEEE Transactions on Cybernetics*, vol. 46, no. 12, pp. 2953–2965, 2016.
- [12] G. Li and Q. Zhang, "Multiple penalties and multiple local surrogates for expensive constrained optimization," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 4, pp. 769–778, 2021.
- [13] K. Deb, "An efficient constraint handling method for genetic algorithms," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, no. 2–4, pp. 311–338, 2000.
- [14] R. A. Sarker, S. M. Elsayed, and T. Ray, "Differential evolution with dynamic parameters selection for optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 5, pp. 689–707, 2013.
- [15] Z. Fan, W. Li, X. Cai, K. Hu, H. Lin, and H. Li, "Angle-based constrained dominance principle in moea/d for constrained multi-objective optimization problems," in *2016 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2016, pp. 460–467.
- [16] P. T. Runarsson and X. Yao, "Stochastic ranking for constrained evolutionary optimization," *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 3, pp. 284–294, 2000.
- [17] T. P. Runarsson and X. Yao, "Search biases in constrained evolutionary optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 35, no. 2, pp. 233–243, 2005.
- [18] T. Takahama, S. Sakai, and N. Iwane, "Constrained optimization by the ε constrained hybrid algorithm of particle swarm optimization and genetic algorithm," in *Proceedings of the 18th Australian Joint Conference on Advances in Artificial Intelligence*. Berlin, Heidelberg: Springer-Verlag, 2005, p. 389–400.
- [19] T. Takahama and S. Sakai, "Constrained optimization by the constrained differential evolution with gradient-based mutation and feasible elites," in *2006 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2006, pp. 1–8.
- [20] W. Yi, Z. Lin, Y. Chen, Z. Pei, and J. Lu, "An enhanced adaptive differential evolution approach for constrained optimization problems," *CMES-Computer Modeling in Engineering & Sciences*, vol. 136, no. 3, 2023.
- [21] S. Zeng, R. Jiao, C. Li, X. Li, and J. S. Alkasasbeh, "A general framework of dynamic constrained multiobjective evolutionary algorithms for constrained optimization," *IEEE Transactions on Cybernetics*, vol. 47, no. 9, pp. 2678–2688, 2017.
- [22] W. Gao, G. G. Yen, and S. Liu, "A dual-population differential evolution with coevolution for constrained optimization," *IEEE Transactions on Cybernetics*, vol. 45, no. 5, pp. 1108–1121, 2014.
- [23] W. Gong and Z. Cai, "A multiobjective differential evolution algorithm for constrained optimization," in *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*. IEEE, 2008, pp. 181–188.
- [24] Z. Hu and W. Gong, "Constrained evolutionary optimization based on reinforcement learning using the objective function and constraints," *Knowledge-Based Systems*, vol. 237, p. 107731, 2022.
- [25] J. Liu, Y. Wang, B. Xin, and L. Wang, "A biobjective perspective for mixed-integer programming," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 4, pp. 2374–2385, 2021.
- [26] X. Wen, G. Wu, M. Fan, R. Wang, and P. N. Suganthan, "Voting-mechanism based ensemble constraint handling technique for real-world single-objective constrained optimization," in *2020 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2020, pp. 1–8.
- [27] H. Lu, H. Tseng, and S. Lin, "Double-track particle swarm optimizer for nonlinear constrained optimization problems," *Information Sciences*, vol. 622, pp. 587–628, 2023.
- [28] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, pp. 341–359, 1997.
- [29] 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, 2013.
- [30] G. Wu, R. Mallipeddi, and P. N. Suganthan, "Problem definitions and evaluation criteria for the cec 2017 competition on constrained real-parameter optimization," *National University of Defense Technology, Changsha, Hunan, PR China and Kyungpook National University, Daegu, South Korea and Nanyang Technological University, Singapore, Technical Report*, 2017.
- [31] B. Wang, H. Li, Q. Zhang, and Y. Wang, "Decomposition-based multi-objective optimization for constrained evolutionary optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 1, pp. 574–587, 2018.
- [32] 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, 2020.
- [33] C. Peng, H. Liu, and F. Gu, "A novel constraint-handling technique based on dynamic weights for constrained optimization problems," *Soft Computing*, vol. 22, pp. 3919–3935, 2018.
- [34] Y. Wang, B. Wang, H. Li, and G. G. Yen, "Incorporating objective function information into the feasibility rule for constrained evolutionary optimization," *IEEE Transactions on Cybernetics*, vol. 46, no. 12, pp. 2938–2952, 2015.