CaR: A Cutting and Repulsion-based Evolutionary Framework for Mixed-Integer Programming Problems

Jiao Liu, Yong Wang, *Senior Member, IEEE*, Pei-Qiu Huang, and Shouyong Jiang

***Abstract*—A mixed-integer programming (MIP) problem contains both constraints and integer restrictions. Integer restrictions divide the feasible region defined by constraints into multiple discontinuous feasible parts. In particular, the number of discontinuous feasible parts will drastically increase with the increase of the number of integer decision variables and/or the size of candidate set of each integer decision variable. Due to the fact that the optimal solution is located in one of discontinuous feasible parts, it is a challenging task to solve a MIP problem. This paper presents a cutting and repulsion-based evolutionary framework (called CaR) to solve MIP problems. CaR includes two main strategies: the cutting strategy and the repulsion strategy. In the cutting strategy, an additional constraint is constructed based on the objective function value of the best individual found so far, the aim of which is to continuously cut unpromising discontinuous feasible parts. As a result, the probability of the population entering a wrong discontinuous feasible part can be decreased. In addition, in the repulsion strategy, once it has been detected that the population has converged to a discontinuous feasible part, the population will be reinitialized. Moreover, a repulsion function is designed to repulse the previous explored discontinuous feasible parts. Overall, the cutting strategy can significantly reduce the number of discontinuous feasible parts and the repulsion strategy can probe the remaining discontinuous feasible parts. Sixteen test problems developed in this paper and two real-world cases are used to verify the effectiveness of CaR. The results demonstrate that CaR performs well on solving MIP problems.**

***Index Terms*—Evolutionary algorithms, mixed-integer programming problems, cutting, repulsion, differential evolution**

I. Introduction

In many science and engineering disciplines, it is common to encounter mixed-integer programming (MIP) problems [1]-[6]. In general, a MIP problem can be formulated as:

$$\begin{aligned} & min \;\; f(\mathbf{x},\mathbf{y}) \\ & s.t. \; g_k(\mathbf{x},\mathbf{y}) \le 0, \; k=1,\ldots,l \\ & \quad\; h_k(\mathbf{x},\mathbf{y}) = 0, \; k=l{+}1,\ldots,p \\ & \quad\; x_i^L \le x_i \le x_i^U, \; i=1,\ldots,n_1 \\ & \quad\; y_i^L \le y_i \le y_i^U, \; i=1,\ldots,n_2 \\ & \quad\; [\mathbf{x},\mathbf{y}] \; \in \; S \\ & \quad\; y_i \; is \; an \; integer \end{aligned} \tag{1}$$

This work was supported by the National Natural Science Foundation of China under Grant 61976225. (*Corresponding authors: Yong Wang and Pei-Qiu Huang.*)

Jiao Liu, Yong Wang, and Pei-Qiu Huang are with the School of Automation, Central South University, Changsha 410083, China. (Email: liu_jiao@csu.edu.cn; ywang@csu.edu.cn; pqhuang@csu.edu.cn).

Shouyong Jiang is with the School of Computer Science, University of Aberdeen, Aberdeen AB24 3EE, U.K. (Email:math4neu@gmail.com).

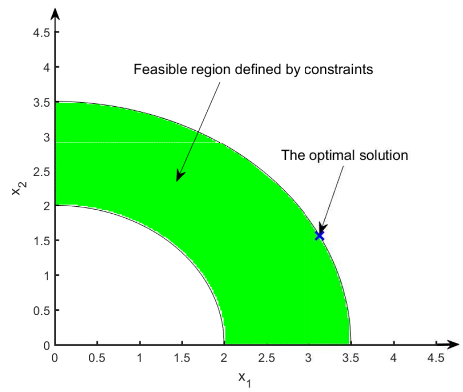


Fig. 1. A common constrained optimization problem in (5). The green area denotes the feasible region defined by constraints, and the blue cross denotes the optimal solution.

where $\mathbf{x} = (x_1,\ldots,x_{n_1})$ and $\mathbf{y} = (y_1,\ldots,y_{n_2})$ are the continuous and integer decision vectors, respectively; S is the decision space; $f(\mathbf{x},\mathbf{y})$ is the objective function, $g_k(\mathbf{x},\mathbf{y})$ is the kth inequality constraint, and $h_k(\mathbf{x},\mathbf{y})$ is the $(k-l)$th equality constraint; x_i^L and x_i^U are the lower and upper bounds of x_i, respectively; y_i^L and y_i^U are the lower and upper bounds of y_i, respectively; n_1 is the number of continuous decision variables, n_2 is the number of integer decision variables, l is the number of inequality constraints, and $(p-l)$ is the number of equality constraints. The degree of constraint violation of $[\mathbf{x},\mathbf{y}]$ is calculated as follows:

$$G_k(\mathbf{x},\mathbf{y}) = \begin{cases} max\{0, g_k(\mathbf{x},\mathbf{y})\}, & 1 \le k \le l \\ max\{0, |h_k(\mathbf{x},\mathbf{y})| - \delta\}, & l{+}1 \le k \le p \end{cases} \tag{2}$$

$$G(\mathbf{x},\mathbf{y}) = \sum_{k=1}^{p} G_k(\mathbf{x},\mathbf{y}) \tag{3}$$

where δ is a positive tolerance value to relax equality constraints to a certain extent. The feasible region defined by all the constraints is denoted as:

$$\Omega = \{[\mathbf{x},\mathbf{y}] | G(\mathbf{x},\mathbf{y}) = 0, [\mathbf{x},\mathbf{y}] \in S\} \tag{4}$$

It is noteworthy that MIP problems can be regarded as a special class of constrained optimization problems. For a common constrained optimization problem, its feasible region is defined by constraints. However, apart from constraints, a MIP problem also contains integer restrictions. These integer

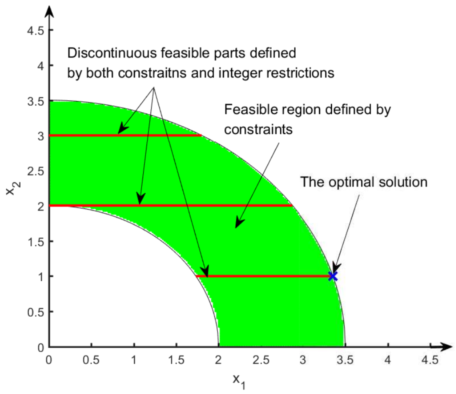


Fig. 2. A MIP problem in (6). The green area denotes the feasible region defined by constraints, the red lines denote the discontinuous feasible parts defined by both constraints and integer restrictions, and the blue cross denotes the optimal solution.

restrictions divide the feasible region into several discontinuous feasible parts. We construct two examples to illustrate the difference between a common constrained optimization problem and a MIP problem. The first example is a common constrained optimization problem:

$$\begin{aligned} &min \ -20x_1 - 10x_2 \\ &s.t. \ x_1^2 + x_2^2 \leq 12.25 \\ &\quad -x_1^2 - x_2^2 \leq -4 \\ &\quad x_1, x_2 > 0 \end{aligned} \tag{5}$$

Fig. 1 shows its feasible region (the green area). The second example is a MIP problem:

$$\begin{aligned} &min \ -20x_1 - 10x_2 \\ &s.t. \ x_1^2 + x_2^2 \leq 12.25 \\ &\quad -x_1^2 - x_2^2 \leq -4 \\ &\quad x_1, x_2 > 0 \\ &\quad x_2 \ is \ an \ integer \end{aligned} \tag{6}$$

Fig. 2 shows its feasible region (the green area) and three discontinuous feasible parts (the red lines). The optimal solution of the first example is located in its feasible region. In contrast, the optimal solution of the second example is located in one of the discontinuous feasible parts. Compared with the first example, solving the second example is more challenging since we need to find the optimal solution from multiple discontinuous feasible parts. Moreover, if a MIP problem has many integer decision variables and/or the value of each integer decision variable can be selected from many integers, it will have a large number of discontinuous feasible parts, which makes it more difficult to be solved. For example, if a MIP problem has two integer decision variables, and the value of each integer decision variable can be selected from 100 integers, it briefly has 10000 discontinuous feasible parts. In this situation, we need to find the optimal solution from these 10000 discontinuous feasible parts, which is obviously a challenging task.

During the past decades, there has been a growing interest in applying evolutionary algorithms (EAs) to solve MIP problems [7]-[11]. When solving MIP problems, EAs should be integrated with constraint-handling techniques and integer-restriction-handling techniques due to the presence of both constraints and integer restrictions. Current constraint-handling techniques can mainly be classified into three classes: methods based on penalty functions [12]-[15], methods based on the preference of feasible solutions over infeasible solutions [16]-[21], and methods based on multiobjective optimization [22]-[25]. Meanwhile, many indirect integer-restriction-handling techniques (e.g., rounding and truncation) [26]-[33] and direct integer-restriction-handling techniques (e.g., discrete code) [34]-[38] have been widely used to handle integer restrictions in MIP problems. However, current methods generally combine constraint-handling techniques with integer-restriction-handling techniques in a straightforward way, and ignore the effect caused by discontinuous feasible parts. Thus, they are easy to converge to a local optimal solution.

In order to find the optimal solution of a MIP problem, we consider the following two issues in this paper:

- Since there are a considerable number of discontinuous feasible parts and the optimal solution is located in one of them, can we gradually remove unpromising discontinuous feasible parts during the evolution?
- If the population has converged to a local optimal solution in one of the remaining discontinuous feasible parts, how can we guide the population to jump out of this local optimal solution and explore other discontinuous feasible parts?

To address these two issues, a novel *c*utting *a*nd *r*epulsion-based evolutionary framework, called CaR, is proposed in this paper to solve MIP problems. CaR contains two main strategies: the cutting strategy and the repulsion strategy.

The main contributions of this paper can be summarized as follows:

- In the cutting strategy, an additional constraint is constructed based on the objective function value of the best individual found so far. By adding this constraint to the original MIP problem, a transformed MIP problem is obtained. This constraint has the capability to continuously cut unpromising discontinuous feasible parts along with the evolution. Therefore, the transformed MIP problem has fewer discontinuous feasible parts than the original one. As a result, the possibility that the population converges to a wrong discontinuous feasible part can be reduced. Note that the feasibility rule [16] and rounding are combined with differential evolution (DE) [39] to solve the transformed MIP problem.
- Even though the cutting strategy can continuously remove some unpromising discontinuous feasible parts, the algorithm may still converge to a local optimal solution in one of the remaining discontinuous feasible parts. In the repulsion strategy, once it has been detected that the algorithm cannot obtain a better solution during a fixed number of generations, we consider that the population has converged to one of the discontinuous feasible parts. Under this condition, the integer decision vector corresponding to this discontinuous feasible part is

recorded and the population is reinitialized. Subsequently, we design a repulsion function to avoid to revisit this discontinuous feasible part. Thus, the population is capable of exploring other promising discontinuous feasible parts.
- Systematic experiments have been conducted to study the performance of CaR by 16 test problems devised in this paper. The results suggest that CaR performs better than three other state-of-the-art EAs on these 16 test problems. Moreover, CaR has been successfully applied to two real-world cases: the deployment optimization problem in the multiunmanned aerial vehicle (multi-UAV)-assisted Internet of things (IoT) data collection system and the path planing problem of the curvature-constrained UAV.

The rest of this paper is organized as follows. Section II introduces DE. The proposed framework, CaR, is elaborated in Section III. The experimental studies and case studies are executed in Section IV and Section V, respectively. Section Finally, Section VI concludes this paper.

II. Differential Evolution (DE)

DE is a very popular population-based optimizer proposed by Storn and Price, which have been applied to solve many real-world optimization problems [39], [40], [41], [42]. It contains four processes: initialization, mutation, crossover, and selection.

In initialization, NP individuals are randomly generated from the decision space:

$$\mathbf{x}_i = (x_{i,1}, \dots, x_{i,n}),\ i = 1, \dots, NP \tag{7}$$

where $\mathbf{x}_i$ is the ith individual and n is the number of the decision variables.

In mutation, a mutant vector $\mathbf{v}_i = (v_{i,1}, \dots, v_{i,n})$ is created for each individual $\mathbf{x}_i$ by the mutation operator. The commonly used mutation operators include:

- DE/rand/1:

$$\mathbf{v}_i = \mathbf{x}_{r_1} + F \times (\mathbf{x}_{r_2} - \mathbf{x}_{r_3}) \tag{8}$$

- DE/rand/2:

$$\mathbf{v}_i = \mathbf{x}_{r_1} + F \times (\mathbf{x}_{r_2} - \mathbf{x}_{r_3}) + F \times (\mathbf{x}_{r_4} - \mathbf{x}_{r_5}) \tag{9}$$

- DE/current-to-rand/1:

$$\mathbf{v}_i = \mathbf{x}_i + rand \times (\mathbf{x}_{r_1} - \mathbf{x}_i) + F \times (\mathbf{x}_{r_2} - \mathbf{x}_{r_3}) \tag{10}$$

- DE/rand-to-best/1:

$$\mathbf{v}_i = \mathbf{x}_{r_1} + rand \times (\mathbf{x}_{best} - \mathbf{x}_{r_1}) + F \times (\mathbf{x}_{r_2} - \mathbf{x}_{r_3}) \tag{11}$$

where r_1, r_2, r_3, r_4, and r_5 are five mutually different integers randomly selected from $\{1, \dots, NP\}$, $\mathbf{x}_{best}$ is the best individual in the population, $rand$ is a uniformly distributed random number from [0, 1], and F is the scaling factor.

By implementing crossover, a trial vector $\mathbf{u}_i = (u_{i,1}, \dots, u_{i,D})$ is generated based on $\mathbf{x}_i$ and $\mathbf{v}_i$. The commonly used crossover operators include:

- Binomial crossover [43]:

$$u_{i,j} = \begin{cases} v_{i,j}, & \text{if } rand_j < CR \text{ or } j = j_{rand} \\ x_{i,j}, & \text{otherwise} \end{cases} \tag{12}$$

where $i = 1, \dots, NP$, $j = 1, \dots, n$, $CR \in [0, 1]$ is the crossover control parameter, $rand_j$ is a uniformly distributed random number between 0 and 1, and j_{rand} is an integer randomly selected from [1, n].

- Exponential crossover [44]:

$$u_{i,j} = \begin{cases} v_{i,j}, & \text{if } j = <l>_n, \dots, <l+L-1>_n \\ x_{i,j}, & \text{otherwise} \end{cases} \tag{13}$$

where $i = 1, \dots, NP$, $j = 1, \dots, n$, $<\cdot>_n$ denotes the modulo function with modulus n, l is a randomly chosen integer from the interval $[1, n]$, and L is an integer drawn from the interval $[1, n]$ with the probability $Pr(L \geq v) = CR^{v-1} (v > 0)$.

Das *et al.* [45] pointed out the behavior of these two crossover operators: the behavior of the exponential crossover is more sensitive to the problem size than the behavior of the binomial crossover. The difference between these two crossover operators lies in different distributions of the number of mutated components.

In selection, the better one between $\mathbf{x}_i$ and $\mathbf{u}_i$ is selected into the next generation:

$$\mathbf{x}_i = \begin{cases} \mathbf{u}_i, & \text{if } f(\mathbf{u}_i) \leq f(\mathbf{x}_i) \\ \mathbf{x}_i, & \text{otherwise} \end{cases} \tag{14}$$

III. Proposed Method

A. Cutting

At each generation, the best individual found so far is denoted as $[\mathbf{x}', \mathbf{y}']$, and its objective function value and degree of constraint violation are denoted as f'_{best} and G'_{best}, respectively. Based on f'_{best} and G'_{best}, the following constraint is constructed:

$$g_{p+1}(\mathbf{x}, \mathbf{y}) = f(\mathbf{x}, \mathbf{y}) - f_{cons} \leq 0 \tag{15}$$

where

$$f_{cons} = \begin{cases} f'_{best}, & \text{if } G'_{best} \leq 0 \\ +\infty, & \text{otherwise} \end{cases} \tag{16}$$

According to (16), if $[\mathbf{x}', \mathbf{y}']$ is an infeasible individual, $f_{cons} = +\infty$. In this case, (15) will not take effect. On the other hand, if $[\mathbf{x}', \mathbf{y}']$ is a feasible individual, $f_{cons} = f'_{best}$. Then, to satisfy (15), an individual must be better than $[\mathbf{x}', \mathbf{y}']$ in terms of the objective function value. Under this condition, the discontinuous feasible parts in which the objective function values of all solutions are worse than the objective function value of $[\mathbf{x}', \mathbf{y}']$ cannot satisfy (15). As a result, they are cut by (15). In this paper, such discontinuous feasible parts are considered as unpromising discontinuous feasible parts.

By adding (15) to (1), a transformed MIP problem can be

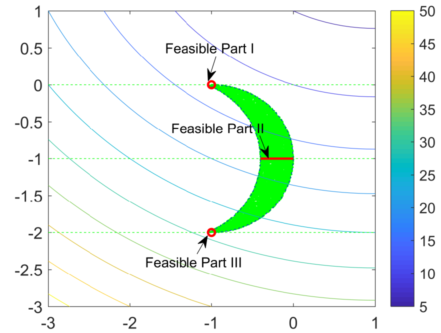


Fig. 3. Contours of the objective function and the feasible region of $P1$. The green area is the feasible region defined by constraints, the red line and red points are the discontinuous feasible parts defined by both constraints and integer restrictions, and the optimal solution is located in feasible part I.

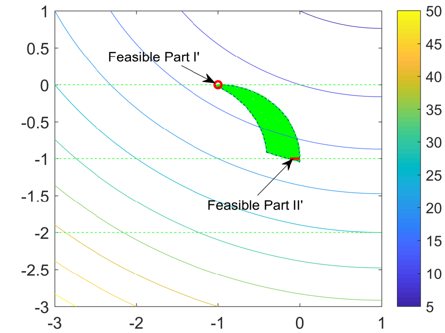


Fig. 4. Contours of the objective function and the feasible region of $P2$. The green area is the feasible region defined by constraints, the red line and red points are the discontinuous feasible parts defined by both constraints and integer restrictions, and the optimal solution is located in feasible part I′.

obtained:

$$\begin{aligned} & min \quad f(\mathbf{x},\mathbf{y}) \\ & s.t. \; g_k(\mathbf{x},\mathbf{y}) \leq 0, \;\; k=1,\ldots,l \\ & \quad\;\; h_k(\mathbf{x},\mathbf{y}) = 0, \;\; k=l{+}1,\ldots,p \\ & \quad\;\; g_{p+1}(\mathbf{x},\mathbf{y}) = f(\mathbf{x},\mathbf{y}) - f_{cons} \leq 0 \\ & \quad\;\; x_i^L \leq x_i \leq x_i^U, \;\; i=1,\ldots,n_1 \\ & \quad\;\; y_i^L \leq y_i \leq y_i^U, \;\; i=1,\ldots,n_2 \\ & \quad\;\; [\mathbf{x},\mathbf{y}] \;\in\; S \\ & \quad\;\; y_i \; is \; an \; integer \end{aligned} \tag{17}$$

This transformed MIP problem has the following three properties:

- **Property 1**: Due to the fact that the optimal solution of (1) can satisfy (15), (1) and (17) have the same optimal solution.
- **Property 2**: Compared with (1), (17) has fewer discontinuous feasible parts. It is because unpromising discontinuous feasible parts have been removed.
- **Property 3**: When solving (17), all the individuals in the population, other than $[\mathbf{x}', \mathbf{y}']$, are infeasible solutions. These infeasible solutions will be motivated to enter the remaining promising discontinuous feasible parts during the evolution.

Next, we design two artificial test functions (i.e., $P1$ and $P2$) to illustrate the principle of the cutting strategy.

$$\begin{aligned} P1: \; & min \quad (x_1-1)^2+(x_2-3)^2 \\ & s.t. \; (x_1+1)^2+(x_2+1)^2 \leq 1 \\ & \quad\;\; (x_1+1.5)^2+(x_2+1)^2 \leq 1.2 \\ & \quad\;\; x_1 \in [-3,1] \\ & \quad\;\; x_2 \in \{-3,-2,-1,0,1\} \end{aligned} \tag{18}$$

$$\begin{aligned} P2: \; & min \quad (x_1-1)^2+(x_2-3)^2 \\ & s.t. \; (x_1+1)^2+(x_2+1)^2 \leq 1 \\ & \quad\;\; (x_1+1.5)^2+(x_2+1)^2 \leq 1.2 \\ & \quad\;\; (x_1-1)^2+(x_2-3)^2 \leq 17.3 \\ & \quad\;\; x_1 \in [-3,1] \\ & \quad\;\; x_2 \in \{-3,-2,-1,0,1\} \end{aligned} \tag{19}$$

It can be observed from (18) and (19) that the only difference between $P1$ and $P2$ is that $P2$ has an additional constraint. The left side of this constraint is the same with the objective function, and the right side is a constant 17.3. The optimal solutions of $P1$ and $P2$ are the same, which are $(-1,0)$, and the optimal objective function values of $P1$ and $P2$ are both 13. The contours of the objective function and the feasible region of $P1$ and $P2$ are shown in Fig. 3 and Fig. 4, respectively. The green area is the feasible region defined by constraints, and the red line and red points are the discontinuous feasible parts defined by both constraints and integer restrictions. As shown in Fig. 3 and Fig. 4, $P1$ contains three discontinuous feasible parts (denoted as feasible part I, feasible part II, and feasible part III) and $P2$ contains two discontinuous feasible parts (denoted as feasible part I′ and feasible part II′). Note that the optimal solutions of $P1$ and $P2$ are located in feasible part I and feasible part I′, respectively.

Assume that the best individual found so far is a feasible individual, and its objective function value is 17.3. Clearly, as shown in Fig. 4, the unpromising discontinuous feasible parts are cut by the additional constraint in $P2$. Meanwhile, since the optimal objective function value of $P1$ is 13, this additional constraint does not cut the optimal solution. As a result, compared with $P1$, $P2$ has only two discontinuous feasible parts, and the best individuals in these two discontinuous feasible parts have better objective function values than the best individual in feasible part III.

B. Repulsion

The repulsion strategy includes two steps. The aim of the first step are twofold: 1) judging whether the population has converged to a discontinuous feasible part, and 2) recording the explored discontinuous feasible parts. In addition, the second step is to repulse the explored discontinuous feasible parts.

The process of the first step is explained as follows. First of all, we define a counter denoted as ctr, the initial value of which is equal to zero. Suppose that $f_{best}^{''}$ and $G_{best}^{''}$ are the objective function value and the degree of constraint violation of the best individual in the current generation, respectively. If $(f_{best}' - f_{best}^{''}) \leq 0$ and $(G_{best}' - G_{best}^{''}) \leq 0$, which

Algorithm 1 CaR

1: Generate the initial population $\mathbf{X}_0 = \{\mathbf{x}_{0,1}, \ldots, \mathbf{x}_{0,NP}\}$;
2: Evaluate the f value and G value of each individual in $\mathbf{X}_0$;
3: $Arc = \emptyset$; // Arc denotes a predefined archive
4: t = 0; // t denotes the generation number
5: ctr = 0; // ctr denotes the counter
6: *FEs* = 0; // *FEs* denotes the number of fitness evaluations
7: **while** *FEs*<*MaxFEs* **do**
8: **for** each individual $\mathbf{x}_{t,i}$ in $\mathbf{X}_t$ **do**
9: Generate a trial vector $\mathbf{v}_{t,i}$ via DE's mutation and crossover operators;
10: Implement rounding on the integer decision variables of $\mathbf{v}_{t,j}$;
11: Calculate the objective function values of $\mathbf{v}_{t,i}$ and $\mathbf{x}_{t,i}$ based on $f(\mathbf{x}, \mathbf{y})$ in (17);
12: Calculate the degree of constraint violation of $\mathbf{v}_{t,i}$ and $\mathbf{x}_{t,i}$ based on (20);
13: Select the better one between $\mathbf{v}_{t,i}$ and $\mathbf{x}_{t,i}$ according to the feasibility rule;
14: *FEs* = *FEs* + 1;
15: **end for**
16: **if** $(f'_{best} - f''_{best}) \leq 0$ & $(G'_{best} - G''_{best}) \leq 0$ **then**
17: $ctr = ctr + 1$;
18: **else**
19: $ctr = 0$;
20: **end if**
21: **if** $ctr > T$ **then**
22: Add the integer decision vector of the current best individual into Arc;
23: Reinitialize the population;
24: $ctr = 0$;
25: **end if**
26: $t = t + 1$;
27: **end while**
28: Output the best individual

means that the algorithm fails to find a better individual, then $ctr = ctr + 1$; otherwise, $ctr = 0$. If ctr is bigger than a predefined threshold *T*, which means that the algorithm cannot find a better individual during consecutive T generations, then we consider that the population has converged to a discontinuous feasible part and this discontinuous feasible part has been explored. Under this condition, the integer decision vector of the current best individual, which corresponds to the explored discontinuous feasible part, is recorded into a predefined archive *Arc*. Meanwhile, to make the population jump out of this discontinuous feasible part, the population will be reinitialized, and ctr will be reset as 0. Overall, *Arc* records all the explored discontinuous feasible parts.

In the second step, the explored discontinuous feasible parts recorded in *Arc* are repulsed as follows. Based on the integer decision vectors in *Arc*, the actual degree of constraint violation of an individual $[\mathbf{x}, \mathbf{y}]$ is calculated as:

$$G_{rep}(\mathbf{x}, \mathbf{y}) = \begin{cases} G(\mathbf{x}, \mathbf{y}), & \text{if } \mathbf{y} \text{ is different from any} \\ & \text{integer decision vector in } Arc \\ G(\mathbf{x}, \mathbf{y}) + \eta, & \text{otherwise} \end{cases} \tag{20}$$

where $G(\mathbf{x}, \mathbf{y})$ is calculated according to (3) and η is a very big positive number. When calculating $G(\mathbf{x}, \mathbf{y})$, all the constraints in (17) are employed. According to (20), it can be concluded that an individual which has the same integer decision vector with one of the integer decision vectors in *Arc* will have a very big G_{rep} value. Since the individuals are selected based on their f values and G_{rep} values, the individuals with very big G_{rep} values are hard to survive into the next generation. As a result, all the explored discontinuous feasible parts can be repulsed.

C. CaR

The implementation of CaR is shown in **Algorithm 1**. First, the generation number $t = 0$, an initial population $\mathbf{X}_0 = \{\mathbf{x}_{0,1}, \ldots, \mathbf{x}_{0,NP}\}$ is randomly produced from the decision space, *Arc* is initialized as an empty set, and ctr is initialized as 0. Note that $\mathbf{x}_{0,j}$ $(j \in \{1, \ldots, NP\})$ is an individual containing both continuous decision variables and integer decision variables. At each generation, according to the cutting strategy, (17) is constructed. Next, for each individual $\mathbf{x}_{t,j}$, a trial vector $\mathbf{v}_{t,j}$ is generated via DE operators, and rounding is executed on the integer decision variables of $\mathbf{v}_{t,j}$. Afterward, $\mathbf{v}_{t,j}$ and $\mathbf{x}_{t,j}$ are evaluated based on (17) and (20), and their f values and G_{rep} values can be obtained as follows: Subsequently, the better one between $\mathbf{v}_{t,j}$ and $\mathbf{x}_{t,j}$ will be selected for the next generation according to the feasibility rule [16]. In **Algorithm 1**, Steps 15-19 are executed to update the status of *ctr*, and Steps 20-24 are executed to determine whether to update *Arc* and to reinitialize the population according to the status of *ctr*. The above procedure is repeated until the termination condition is satisfied.

Next, we would like to give the following comments on CaR:

- In fact, CaR is implemented based on the following two key components: 1) transforming (1) to (17) by adding a constructed constraint (15) (i.e., the cutting strategy), and 2) redefining the degree of constraint violation of an individual by (20) (i.e., the repulsion strategy).
- Many methods have been proposed to deal with infeasible solutions [46]. In this paper, the feasibility rule proposed by Deb *et al.* [16] is employed to handle infeasible solutions. We select this method because of the following two reasons: 1) the feasibility rule is simple and easy to operate, and 2) according to property 3, when solving (17), all the solutions in the population, other than the best solution in the population, are infeasible solutions, and it is desirable to motivate these infeasible solutions to enter the remaining discontinuous feasible parts. As the feasibility rule tends to push the population into the feasible region [21], it is very suitable for our method.

D. Discussions about CaR

1) Relationship Between the Branch-and-Bound Method and the Cutting Strategy: Both the branch-and-bound method and the cutting strategy aim to bound the search. However, they have the following two differences:

- In the branch-and-bound method, some of the branches that have no chance to find a better solution are eliminated directly. However, in the cutting strategy, the added constraint bounds the search by changing the unpromising discontinuous feasible parts into the infeasible regions, thus reducing the number of the discontinuous feasible parts.

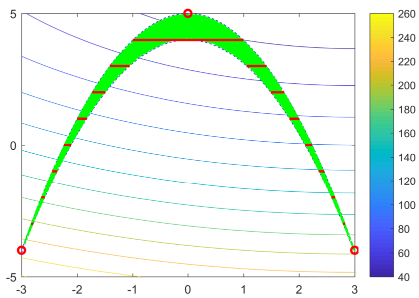


Fig. 5. Contours of the objective function and the feasible region of $P3$. The green area is the feasible region defined by constraints, and the red line and red points are the discontinuous feasible parts defined by both constraints and integer restrictions.

- Commonly, in the branch-and-bound method, an accurate algorithm is employed to solve the subproblem corresponding to each branch. However, in the cutting strategy, DE is used to solve the transformed problem.

2) Relationship Between the Tabu Search and the Repulsion Strategy: The repulsion strategy shares the similar idea with the tabu search. Both of them record a set of local optimal solutions into an archive, and taboo these recorded local optimal solutions. However, they have the following two differences:

- In the tabu search, some complete solutions are recoded and tabooed, the aim of which is to avoid revisiting previous areas. However, in the repulsion strategy, only the integer variables of a local optimal solution are recored and repulsed, the aim of which is to repulse an unpromising discontinuous part.
- In the tabu search, the archive is updated with the iteration of the algorithm, and some of the recorded solutions have the chance to be dropped by the archive. However, in the repulsion strategy, once the integer variables of a local optimal solution are recorded, they never be dropped by the archive.

E. Proof-of-Principle Results

An artificial test function $P3$ is constructed to explain the principle of CaR.

$$
\begin{aligned}
P3: \ min \ \ & (x_1-3)^2+(x_2-10)^2 \\
s.t. \ & -x_1^2-x_2 \leq -5 \\
& 0.9x_1^2+x_2 \leq 4 \\
& x_1 \in [-3,5] \\
& x_2 \in \{-5,-4,-3,-2,-1, \\
& \quad 0,1,2,3,4,5\}
\end{aligned} \tag{21}
$$

The optimal solution of $P3$ is (0, 5) and the optimal objective function value is 34. The contours of the objective function and the feasible region of $P3$ are shown in Fig. 5. The green area is the feasible region defined by constraints, and the red line and red points are the discontinuous feasible parts defined by both constraints and integer restrictions. In total, $P3$ contains 18 discontinuous feasible parts.

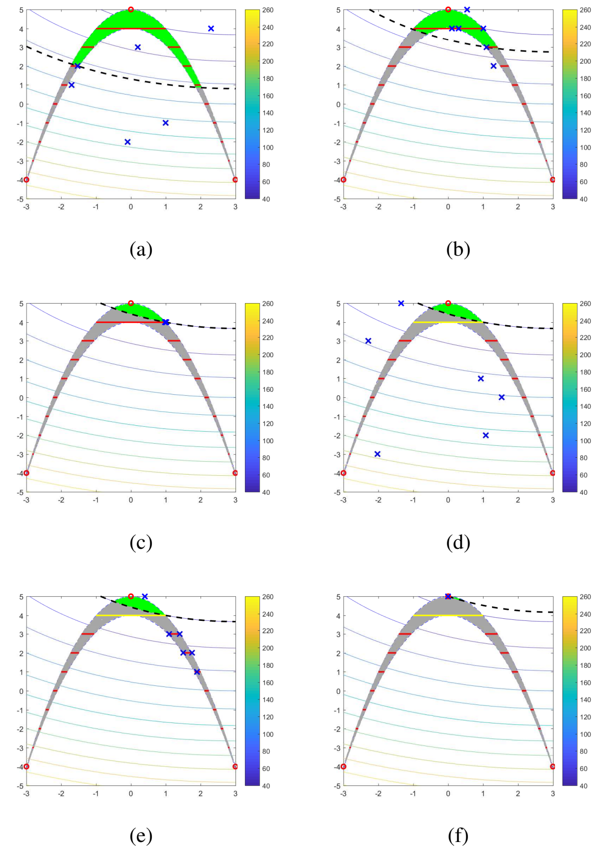


Fig. 6. Evolution of CaR in a typical run on $P3$. The black dashed line is the constraint constructed based on (15), the region cut by this constraint is shown in gray, and the yellow line segment is the repulsed discontinuous feasible part. (a) The 1th generation. (b) The 10th generation. (c) The 78th generation. (d) The 85th generation. (e) The 100th generation. (f) The 150th generation.

When solving $P3$ by CaR, NP was set to 6, T was set to 50, and the evolution operator was DE/rand/1/bin. Fig. 6 provides a typical run derived from CaR. In Fig. 6, the black dashed line is the constraint constructed based on (15), and the region cut by (15) is shown in gray. It can be observed from Fig. 6(a) that (15) can cut 11 discontinuous feasible parts at the beginning. Along with the evolution, more region can be cut. Specifically, 15 and 16 discontinuous feasible parts are cut in Fig. 6(b) and Fig. 6(c), respectively. In the 85th generation, the population cannot find a better solution during consecutive 50 generations. Therefore, we consider that the population has converged to a local optimum. Under this condition, the integer decision vector of the current best individual is stored into *Arc*. Then, as shown in Fig. 6(d), the yellow discontinuous feasible part, which corresponds to the recorded integer decision vector, is repulsed and the population is reinitialized. Afterward, the evolution proceeds. From Fig. 6(e), it can be seen that the population cannot enter the yellow discontinuous feasible part again, and will search for the optimal solution from other promising discontinuous feasible parts. Finally, the population attains the optimal solution as shown in Fig. 6(f).

TABLE I
CHARACTERISTICS OF THE 16 TEST PROBLEMS, WHERE *n* IS THE NUMBER OF DECISION VARIABLES, n_1 IS THE NUMBER OF CONTINUOUS DECISION VARIABLES, n_2 IS THE NUMBER OF INTEGER DECISION VARIABLES, *IC* IS THE NUMBER OF INEQUALITY CONSTRAINTS, AND *EC* IS THE NUMBER OF EQUALITY CONSTRAINTS.

Problem	*n*	n_1	n_2	*IC*	*EC*
F1	2	1	1	1	0
F2	3	1	2	1	0
F3	2	1	1	3	0
F4	2	1	1	2	0
F5	2	1	1	0	1
F6	2	1	1	2	0
F7	5	3	2	0	3
F8	8	5	3	6	0
F9	8	5	3	6	0
F10	8	5	3	6	0
F11	15	12	3	5	0
F12	15	10	5	5	0
F13	6	4	2	0	4
F14	6	4	2	0	4
F15	10	7	3	8	0
F16	10	5	5	8	0

TABLE II
PARAMETER SETTINGS OF CAR

Parameter	Value
Population size: NP	60
Predefined threshold: T	800
Maximum number of function evaluations: $MaxFEs$	2.0E+05
Tolerance value: δ	0.0001

IV. EXPERIMENTAL STUDY

A. Test Problems and Parameters Settings

Sixteen test problems (called F1-F16) were developed in this paper to investigate the performance of CaR, which are provided in the supplementary file. All of them are minimization MIP problems. Specifically, F1-F4 were designed by ourselves and F5-F16 were designed based on the test functions collected in IEEE CEC2006 [47]. Their characteristics are listed in Table I, where *n* is the number of decision variables, n_1 is the number of continuous decision variables, n_2 is the number of integer decision variables, *IC* is the number of inequality constraints, and *EC* is the number of equality constraints.

For each test problem, 25 independent runs were implemented. The parameter settings of CaR are listed in Table II. These parameters were set according to the following considerations:

- For NP, in most of current EAs, it was set between 20 and 100 [20], [48]. In terms of T, inspired by [49] and [50], we considered that if an algorithm cannot obtain a better solution after several hundreds of FEs, it would be difficult to continue to find a better solution. As a result, NP and T were set to 60 and 800, respectively. Furthermore, we investigated the influence of these two parameters in Section S-III of the supplementary file, and found that the performance of CaR is insensitive to NP and T.
- In terms of $MaxFEs$, we plotted the convergence curves to exhibit the convergence of CaR in Section S-III of the supplementary file, and found that, in most cases, CaR could converge within 5.0E+04 function evaluations (FEs). However, to ensure that CaR can converge completely when solving an unknown problem, $MaxFEs$ was set to a large value, i.e., 2.0E+05.
- In terms of δ, in existing studies of constrained EAs, δ was usually set to 0.0001 [11], [20], [21] since it was considered as an acceptable precision.

To evaluate the performance of different algorithms, the following five statistical values were calculated:

- Feasible Rate (*FR*): The percentage of runs where an algorithm can find at least one feasible solution in the end.
- Successful Rate (*SR*): The percentage of runs where an algorithm can successfully obtain the optimal solution in the end. Note that, a run is considered as successful if the following condition is satisfied: $|f(\mathbf{x}_{best}) - f(\mathbf{x}^*)| \leq 0.0001$, where $\mathbf{x}^*$ is the best known solution and $\mathbf{x}_{best}$ is the best feasible solution provided by an algorithm.
- Average (*Ave*): The average objective function value of the best feasible solutions provided by an algorithm over 25 independent runs. If an algorithm cannot achieve 100% *FR* over 25 independent runs, *Ave* is denoted as "*NA*".
- Standard Deviation (*Std Dev*): The standard deviation of the objective function values of the best feasible solutions provided by an algorithm over 25 independent runs. Similarly, if an algorithm cannot achieve 100% *FR* over 25 independent runs, *Std Dev* is denoted as "*NA*".
- Average CPU Time (*ACT*): The average CPU time over 25 independent runs on each case. In this paper, the CPU time is measured by second.

In the experimental study, the Wilcoxon's rank-sum test at a 0.05 significance level was implemented between CaR and its competitor to test the statistical significance. In Tables S-II–S-V of the supplementary file, "+", "−", and "≈" denote that CaR is better than, worse than, and similar to its competitor, respectively.

B. Effectiveness of CaR with Different DE Variants

In this subsection, we validated the effectiveness of CaR by employing four different DE variants, i.e., DE/rand/1/bin, DE/rand/2/bin, DE/current-to-rand/bin, and DE/rand-to-best/bin introduced in Section II. The resultant four algorithms are denoted as DE/rand/1/bin-CaR, DE/rand/2/bin-CaR, DE/current-to-rand/bin-CaR, and DE/rand-to-best/bin-CaR, respectively. These four algorithms were compared with their corresponding original versions (i.e., DE/rand/1/bin-MIP, DE/rand/2/bin-MIP, DE/current-to-rand/bin-MIP, and DE/rand-to-best/bin-MIP). In the four original versions, the cutting strategy and the repulsion strategy were not used. Note that the eight compared algorithms employed the feasibility rule to handle constraints and rounding to deal with integer restrictions. In DE, the values of F and CR have an important influence on the performance of the algorithm [45]. F controls the ranges of the generated mutation vectors and CR controls how many variables in expectation are changed in a population member. In this paper, we followed the suggestion of Store *et al*. [43], and set these two parameters to 0.5 and 0.9, respectively.

The results are summarized in Table S-II of the supplementary file. From Table S-II, it is clear that no matter which DE variant is employed, CaR can significantly improve the performance. The detailed discussions are given below:

- DE/rand/1/bin-CaR provides 100% *FR* on all the 16 test problems; however, DE/rand/1/bin-MIP fails to provide 100% *FR* on F10 and F14. In addition, DE/rand/1/bin-CaR provides better *SR* on 14 test problems (i.e., F1-F3, F5, F6, and F8-F16). For seven out of the 16 test problems, DE/rand/1/bin-CaR successfully solves them over 25 runs. In contrast, DE/rand/1/bin-MIP achieves 100% *SR* only on F4. Moreover, for all the 16 test problems except F4, DE/rand/1/bin-CaR performs better than DE/rand/1/bin-MIP in terms of *Ave*. From the Wilcoxon's rank-sum test, DE/rand/1/bin-CaR surpasses DE/rand/1/bin-MIP on 14 test problems. However, DE/rand/1/bin-MIP cannot beat DE/rand/1/bin-CaR on any test problem.
- For DE/rand/2/bin-MIP and DE/rand/2/bin-CaR, although both of them can provide 100% *FR* on all the 16 test problems except F7, DE/rand/2/bin-CaR produces better *SR* on 14 test problems (i.e., F1-F3, F5, F6, and F8-F16). In terms of *Ave*, DE/rand/2/bin-CaR exhibits better performance on all the 16 test problems except F4 and F7. According to the Wilcoxon's rank-sum test, DE/rand/2/bin-CaR beats DE/rand/2/bin-MIP on 13 test problems. However, DE/rand/2/bin-MIP cannot outperform DE/rand/2/bin-CaR on any test problem.
- DE/current-to-rand/bin-MIP provides 100% *FR* on 14 test problems (i.e., F1-F6, F8, and F10-F16). In contrast, DE/current-to-rand/bin-CaR achieves 100% *FR* on all the 16 test problems. In terms of *SR*, DE/current-to-rand/bin-CaR is better than DE/current-to-rand/bin-MIP on 13 test problems (i.e., F1-F3, F5, F6, F8-F10, and F12-F16). In addition, DE/current-to-rand/bin-CaR provides better *Ave* on all the 16 test problems except F4 and F15. From the Wilcoxon's rank-sum test, DE/current-to-rand/bin-CaR surpasses DE/current-to-rand/bin-MIP on 13 test problems; however, DE/current-to-rand/bin-MIP cannot beat DE/current-to-rand/bin-CaR on any test problem.
- DE/rand-to-best/bin-MIP fail to provide 100% *FR* on four test problems (F8-F10 and F14); however, DE/rand-to-best/bin-CaR can provide 100% *FR* on all the 16 test problems. Regarding *SR*, DE/rand-to-best/bin-CaR is better than DE/rand-to-best/bin-MIP on 11 test problems (i.e., F1-F3, F5-F7, F9, F10, F13, F14, and F16). In terms of *Ave*, DE/rand-to-best/bin-CaR shows better performance than DE/rand-to-best/bin-MIP on 15 test problems (F1-F3 and F5-F16). According to the Wilcoxon's rank-sum test, DE/rand-to-best/bin-CaR outperforms DE/rand-to-best/bin-MIP on 11 test problems and performs similarly on the remaining test problems.
- In terms of *ACT*, it can be observed that the four DE variants with CaR usually take about one second longer than the four DE variants without CaR on each case. Therefore, the use of CaR does not add any significant computational burden.

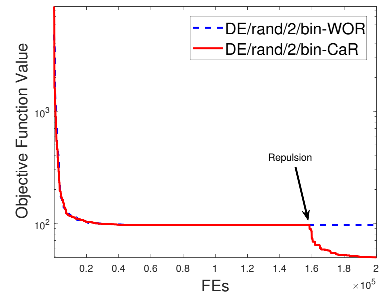


Fig. 7. Convergence curves provided by DE/rand/2/bin-WOR and DE/rand/2/bin-CaR on solving F12.

The superiority of CaR can be attributed to the fact that it is able to effectively reduce the possibility that the population converges to a wrong discontinuous feasible part.

C. Effectiveness of The Cutting Strategy and The Repulsion Strategy

As introduced in Section III, CaR includes two main strategies, i.e., the cutting strategy and the repulsion strategy. To investigate the effectiveness of them, three variants of DE/rand/2/bin-CaR were devised, which were named as DE/rand/2/bin-WOCR, DE/rand/2/bin-WOC, and DE/rand/2/bin-WOR, respectively. In DE/rand/2/bin-WOC, only the repulsion strategy was employed; in DE/rand/2/bin-WOR, only the cutting strategy was utilized; and in DE/rand/2/bin-WOCR, both the cutting strategy and the repulsion strategy were removed.

The results are given in Table S-III of the supplementary file. As shown in Table S-III, in terms of *SR* and *Ave*, DE/rand/2/bin-WOC and DE/rand/2/bin-WOR have an edge over DE/rand/2/bin-WOCR on 14 test problems (i.e., F1, F3, and F5-F16) and 14 test problems (i.e., F1-F3 and F6-F16), respectively. Thus, one can conclude that both of these two strategies have a positive influence on the performance of CaR. However, compared with DE/rand/2/bin-CaR, DE/rand/2/bin-WOC and DE/rand/2/bin-WOR perform worse on 11 (i.e., F2, F3, F6, and F8-F15) and nine (i.e., F5, F7, and F9-F15) test problems, respectively. From the Wilcoxon's rank-sum test, it can be seen that DE/rand/2/bin-CaR beats DE/rand/2/bin-WOCR, DE/rand/2/bin-WOC, and DE/rand/2/bin-WOR on 13, eight, and seven test problems, respectively. However, DE/rand/2/bin-WOCR, DE/rand/2/bin-WOC, and DE/rand/2/bin-WOR cannot surpass FROFI-CaR on any test problem. The above results can be explained as follows. Although the cutting strategy has the capability to remove unpromising discontinuous feasible parts, the population may be trapped into one of the remaining wrong discontinuous feasible parts if the repulsion strategy is not utilized. On the other hand, if only the repulsion strategy is adopted, even though the population can jump out of some wrong discontinuous feasible parts, it still runs a high risk to converge to a wrong discontinuous feasible part since a MIP problem may have a large number of discontinuous feasible parts.

We also use an example to further exhibit the effectiveness of the repulsion strategy. This example shows the convergence curves of DE/rand/2/bin-WOR and DE/rand/2/bin-CaR on solving F12 in Fig. 7. From Fig. 7, CaR-WOR stagnates from about 4.0E+04 FEs to 2.0E+05 FEs. Although CaR also stagnates from about 4.0E+04 FEs to 1.6E+05 FEs, it jumps out from the local optimum at about 1.6E+05 FEs. This is because the repulsion strategy reinitializes the population and makes the algorithm explore other promising feasible parts, thus finding a better solution.

D. Combining CaR with Other EAs

In principle, CaR is an open framework and can be flexibly combined with EAs to deal with MIP problems. To verify this, by combining CaR with JADE [48] and FROFI [21], two new algorithms, called JADE-CaR and FROFI-CaR, were designed. We also developed two compared methods called JADE-MIP and FROFI-MIP. In JADE-MIP and FROFI-MIP, rounding was used to deal with integer restrictions. For JADE-MIP and JADE-CaR, the feasibility rule was employed to handle constraints. In these four compared algorithms, all the parameter settings were the same with their original algorithms (i.e., JADE and FROFI).

The results are shown in Table S-IV of the supplementary file and the detailed discussions are given below:

- As far as *FR*, *SR*, and *Ave* are concerned, JADE-CaR achieves better performance than JADE-MIP on 14 test problems (i.e., F1-F3, F5, and F7-F16). According to the Wilcoxon's rank-sum test, JADE-CaR surpasses JADE-MIP on 10 test problems. However, JADE-MIP cannot beat JADE-CaR on any test problem.
- Regarding *SR* and *Ave*, FROFI-CaR is better than FROFI-MIP on 13 test problems (i.e., F1, F3, F5, and F7-F16). According to the Wilcoxon's rank-sum test, FROFI-CaR outperforms FROFI-MIP on 11 test problems and performs similarly on the remaining test problems.

The above comparison demonstrates the potential of CaR when combined with other EAs. In addition, from Table S-IV, it can also be observed that FROFI-CaR is significantly better than JADE-CaR on 10 test problems (i.e., F7-F16) in terms of *SR* and *Ave*. This is because FRORI includes a replacement mechanism which can alleviate the greediness of the feasibility rule by exploiting objective function information.

E. Comparison with Three State-of-the-Art Methods for Solving MIP Problems

Based on the results in Section IV-D, as an instance of our framework, FROFI-CaR shows good performance for solving MIP problems. Next, we selected three state-of-the-art EAs (i.e., MDE-LS [31], MDE-IHS [31], and EMDE [32]), and compared their performance with FROFI-CaR. MDE-LS incorporates a local search operator into DE, with the aim of improving the exploitation ability. In MDE-IHS, harmony search is integrated with DE to enhance the search ability. In EMDE, a novel triangular mutation operator is designed. In our experiment, the DE operators used in MDE-LS, MDE-IHS, and FROFI-CaR were the same with FROFI. Since one main contribution of EMDE is the triangular mutation operator, EMDE was implemented without any modification.

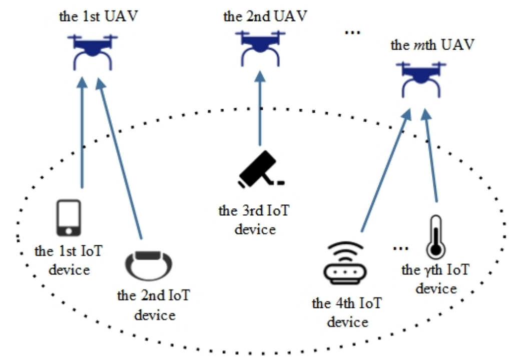


Fig. 8. Multi-UAV-assisted IoT data collection system considered in this paper

The results provided by FROFI-CaR, MDE-LS, MDE-IHS, and EMDE are recorded in Table S-V of the supplementary file. In terms of *SR* and *Ave*, FROFI-CaR is better than MDE-LS and MDE-IHS on all the 16 test problems except F2 and F4. For F2 and F4, FROFI-CaR, MDE-LS, and MDE-IHS have similar performance. In addition, FROFI-CaR provides similar results with EMDE on F4 and F6, and performs better than EMDE on the remaining 14 test problems. According to the Wilcoxon's rank-sum test, FROFI-CaR outperforms MDE-LS, MDE-IHS, and EMDE on 13, 14, and 14 test problems, respectively. However, MDE-LS, MDE-IHS, and EMDE cannot surpass FROFI-CaR on any test problem. The above comparison suggests that, overall, the performance of FROFI-CaR is better than that of the three competitors on solving the 16 test problems.

V. Case Studies

A. Deployment Optimization Problem in the Multi-UAV-Assisted IoT Data Collection System

Recently, UAVs have become emerging data collection tools in current data collection systems [51], [52]. However, to efficiently use UAVs, the deployment of them should be optimized. In this paper, we use up to m rotary-wing UAVs to collect data from a set of γ ground IoT devices as shown in Fig. 8. Note that, considering the cost of the system management, not all of the m UAVs are employed to collect data; therefore, we use $u_j \in \{0,1\}$ $(j = 1,\ldots,m)$ to represent whether the jth UAV is employed or not. Specifically, if the jth UAV is employed, then $u_j = 1$; otherwise, $u_j = 0$. For each IoT device, its coordinate is known and fixed at $(x_i, y_i, 0)$ $(i = 1,\ldots,\gamma)$. Each UAV is flying horizontally at a constant altitude H, and the location of the jth UAV is denoted as (X_j, Y_j, H). Thus, the distance between the ith IoT device and the jth UAV is expressed as:

$$d_{ij} = \sqrt{(X_j - x_i)^2 + (Y_j - y_i)^2 + H^2} \tag{22}$$

We apply a_{ij} to represent the association between the ith IoT device and the jth UAV. If $a_{ij} = 1$, then the ith IoT device sends data to the jth UAV; otherwise, $a_{ij} = 0$. In the studied system, each IoT always chooses the nearest UAV to send data, thus a_{ij} is calculated as:

$$a_{ij} = \begin{cases} 1, & \text{if } j = \arg\min_{j} d_{ij} \\ 0, & \text{otherwise} \end{cases} \tag{23}$$

Also, each IoT device chooses only one UAV to send its data, thus

$$\sum_{j=1}^{m} u_j a_{ij} = 1 \tag{24}$$

Each UAV can accept at most M IoT devices to send data simultaneously due to the system bandwidth limitation. Thus, we have:

$$\sum_{i=1}^{\gamma} u_j a_{ij} \leq M \tag{25}$$

Moreover, to ensure that all the IoT devices can be serviced, the following condition should be satisfied:

$$\sum_{i=1}^{\gamma}\sum_{j=1}^{m} u_j a_{ij} = \gamma \tag{26}$$

Our aim is to minimize the energy consumption of the whole system, which is composed of the energy consumption of all the IoT devices and UAVs. The energy consumption of all the IoT devices is calculated as:

$$E_{iot} = \sum_{i=1}^{\gamma}\sum_{j=1}^{m} u_j a_{ij} E_{ij} \tag{27}$$

where E_{ij} is the energy consumption of the ith IoT device if it sends data to the jth UAV. E_{ij} is calculated as follows:

$$E_{ij} = \frac{p_i D_i}{r_{ij}} \tag{28}$$

$$r_{ij} = B \log(1 + \frac{p_i h_0 d_{ij}^{-2}}{\sigma^2}) \tag{29}$$

where p_i and D_i are the transmitting power and datasize of the ith IoT device, respectively, r_{ij} is the transmission rate if the ith IoT device sends data to the jth UAV, h_0 is the channel power gain at the reference distance $d_0 = 1m$, σ^2 is the white Gaussian noise power, and B is the system bandwidth.

The energy consumption of all the UAVs can be calculated as:

$$E_{uav} = \sum_{j=1}^{m} u_j (E_j^n + E_j^h) \tag{30}$$

where E_j^n and E_j^h are the no-load energy consumption and hover energy consumption of the jth UAV, respectively. For each UAV, the no-load energy consumption is a fixed value. In addition, the hover energy consumption of the jth UAV is calculated as:

$$E_j^h = p^h T_j^h \tag{31}$$

where T_j^h is the hover time of the jth UAV. As each UAV will not be landed until all the data sent from the corresponding IoT devices have been collected, the hover time of the jth UAV is given by:

$$T_j^h = \max_i \{\frac{u_j a_{ij} D_i}{r_{ij}}\} \tag{32}$$

In summary, the considered deployment optimization problem can be formulated as:

$$\begin{aligned} & min\ E(\mathbb{X}, \mathbb{Y}, \mathbb{U}) = E_{uav} + \phi E_{iot} \\ & s.t.\ a_{ij} \in \{0,1\}, \forall i \in \{1, \ldots, \gamma\}, j \in \{1, \ldots, m\} \\ & \quad u_j \in \{0,1\}, \forall j \in \{1, \ldots, m\} \\ & \quad \sum_{j=1}^{m} u_j a_{ij} = 1, \forall i \in \{1, \ldots, \gamma\} \\ & \quad \sum_{i=1}^{\gamma} u_j a_{ij} \leq M, \forall j \in \{1, \ldots, m\} \\ & \quad \sum_{i=1}^{\gamma}\sum_{j=1}^{m} u_j a_{ij} = \gamma \\ & \quad X_{min} \leq X_j \leq X_{max}, \forall j \in \{1, \ldots, m\} \\ & \quad Y_{min} \leq Y_j \leq Y_{max}, \forall j \in \{1, \ldots, m\} \end{aligned} \tag{33}$$

where $\mathbb{X} = \{X_j | j = 1, \ldots, m\}$, $\mathbb{Y} = \{Y_j | j = 1, \ldots, m\}$, $\mathbb{U} = \{u_j | j = 1, \ldots, m\}$, ϕ is a weight, X_{min} and X_{max} are the lower and upper bounds of X_j, respectively, and Y_{min} and Y_{max} are the lower and upper bounds of Y_j, respectively. In this optimization problem, $\mathbb{X}$ and $\mathbb{Y}$ are two sets of continuous variables, and $\mathbb{U}$ is a set of integer variables; therefore, it is a typical MIP problem.

The parameters of (33) were set as follows:

- All the IoT devices were randomly distributed in a 2000 m $\times$ 2000 m square area, i.e., $X_{min} = Y_{min} = -1000$ and $X_{max} = Y_{max} = 1000$.
- The flight altitude of each UAV was 200 m, i.e., $H = 200$.
- D_i was randomly distributed within [1,1000] MB.
- The no-load energy consumption E_j^n was set to 100 kJ.
- $M = 10$, $p_i = 0.1$ W, $h_0 = -30$ dB, $\sigma = -250$ dBm, $B = 1$ MHz, $p^h = 1000$ W, $\phi = 10000$, $\gamma = 50$, and $m = 10$.

We employed FROFI-CaR to solve (33). For comparison, we also used FROFI-MIP to solve (33). Specifically, each algorithm was independently run 25 times. According to the results, the average E values produced by FROFI-CaR and FROFI-MIP are 1.5735E+06 J and 1.6065E+06 J, respectively. Clearly, FROFI-CaR can provide smaller energy consumption than FROFI-MIP. The above results verify the effectiveness of CaR on solving the deployment optimization problem in the multi-UAV-assisted IoT data collection system.

B. Path Planning Problem of the Curvature-Constrained UAV

Due to the advantages of low cost, high maneuverability, and good survivability, fixed-wing UAVs have shown their great potential in performing surveillance of multiple ground targets [53]. To better complete the task of surveillance, it is necessary to optimize the path of a UAV, and make it fly through all the targets by using the shortest distance traveled,

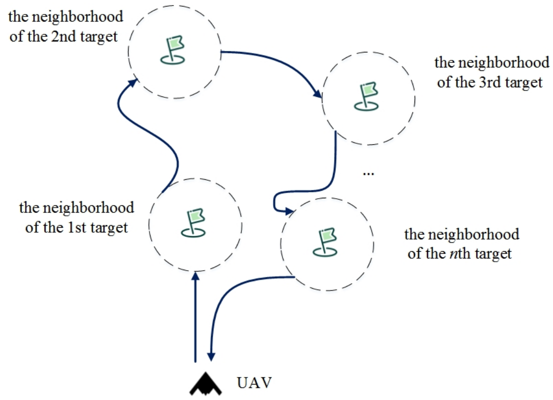
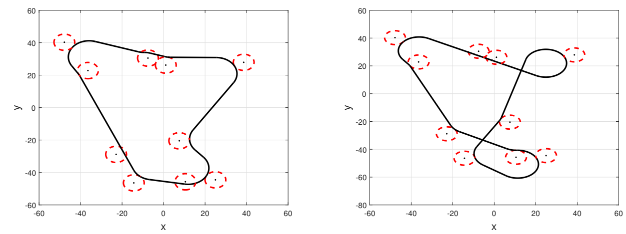


Fig. 9. Curvature-constrained UAV performs surveillant of multiple ground targets



(a) (b)

Fig. 10. Paths provided by DE-CE-CaR and DE-CE-MIP in a typical run. (a) the path provided by DE-CE-CaR (b) the path provided by DE-CE-MIP.

as shown in Fig 9. Note that, the path of the UAV should satisfy the following two conditions:

- To make the planned path flyable, the curvature constraint should be satisfied, i.e., the minimum curvature radius of the path should not be less than the minimal turning radius of the UAV [54]. For simplicity, in this paper, we consider the UAV flies in the Dubins trajectory, which can be described as:

$$\begin{cases} \dot{x} = v\cos(\theta) \\ \dot{y} = v\sin(\theta) \\ \dot{\theta} = \frac{v}{r}u, u \in [-1,1] \\ \dot{v} = 0 \end{cases} \tag{34}$$

  where (x, y) and θ are the planar coordinate and heading of the UAV, respectively, v is the speed of the UAV, r is the minimal turning radius, and u is the control input. Commonly, a triplet (x, y, θ) is called as a configuration.
- As the equipments on the UAV usually have large surveillance scopes, the UAV only needs to pass through a certain neighborhood of each target.

Overall, the considered path planning problem can be described as a Dubins traveling salesman problem with neighborhood [55], which is formulated as:

$$\begin{aligned} &min\ D(\mathbb{R}, \mathbb{X}) = \sum_{i=1}^{n-1} d(\mathbf{X}_{R_i}, \mathbf{X}_{R_{i+1}}) + d(\mathbf{X}_{R_n}, \mathbf{X}_{R_1}) \\ &\ s.t.\ R_i \neq R_j, \text{if } i \neq j, \forall i \in \{1, \ldots, n\}, \forall j \in \{1, \ldots, n\} \\ &\qquad \sqrt{(x_i - ox_i)^2 + (y_i - oy_i)^2} \leq S_i, \forall i \in \{1, \ldots, n\} \\ &\qquad X_{min} \leq x_i \leq X_{max}, \forall i \in \{1, \ldots, n\} \\ &\qquad Y_{min} \leq y_i \leq Y_{max}, \forall i \in \{1, \ldots, n\} \\ &\qquad 0 \leq \theta_i \leq 2\pi, \forall i \in \{1, \ldots, n\} \end{aligned} \tag{35}$$

where $\mathbb{R} = \{R_i | i = 1, \ldots, n\}$ represents the sequence of targets that the UAV needs to surveil, $R_i \in \{1, \ldots, n\}$ is the ith target, n is the number of targets, $\mathbb{X} = \{\mathbf{X}_i | i = 1, \ldots, n\}$ is the set containing all the waypoints that the UAV passes through, $\mathbf{X}_i = (x_i, y_i, \theta_i)$ is the configuration of the ith waypoint, $(x_i,\ y_i)$ is the coordinate of the ith waypoint, θ_i is the heading of the UAV at the ith waypoint, $d(\cdot, \cdot)$ represents the shortest Dubins path between two configurations, $(ox_i,\ oy_i)$ is the coordinate of the ith target, S_i is the radius of the neighborhood of the ith target, X_{min} and X_{max} are the lower and upper bounds of x_i, respectively, and Y_{min} and Y_{max} are the lower and upper bounds of y_i, respectively. For $d(\cdot, \cdot)$, the shortest path from one configuration to another must be one of the following six Dubins path patterns {RSL, LSR, RSR, LSL, RLR, LRL} [56], in which L, R, and S represent turning left with the minimal turning radius, turning right with the minimal turning radius, and moving along a straight line, respectively. Since $\mathbb{X}$ contains several continuous variables, and the elements in $\mathbb{R}$ can be treated as integer variables, (35) is a MIP problem. Moreover, r, S_i, and n were set to 5, 5, and 10, respectively.

We combined DE with CaR (denoted as DE-CE-CaR) to solve (35). To deal with the sequence variables, the encoding method in [55] was employed. For comparison, we also combined DE, the encoding method in [55], and the feasibility rule to solve (35). The resultant algorithm is called DE-CE-MIP. After 20 independently runs, the average D values of DE-CE-CaR and DE-CE-MIP are 299.09 and 329.12, respectively. Hence, the average D value provided by DE-CE-CaR is 9.12% better than that of DE-CE-MIP. Fig. 10(a) and 10(b) depict the paths provided by DE-CE-CaR and DE-CE-MIP in a typical run, respectively. Obviously, DE-CE-CaR can provide a shorter path. The above experiments demonstrate the effectiveness of CaR on solving the path planning problem of the curvature-constrained UAV.

VI. Conclusion

A MIP problem may contain a large number of discontinuous feasible parts if it has many integer decision variables and/or the value of each integer variable can be selected from many integers. Therefore, when solving MIP problems, an algorithm is very likely to converge to a wrong discontinuous feasible part. To overcome the issue, a cutting and repulsion-based evolutionary framework (called CaR) was proposed in this paper. CaR contained two main strategies: the cutting strategy and the repulsion strategy. The cutting strategy aimed to remove unpromising discontinuous feasible parts during the evolution. This strategy was implemented by adding a constraint constructed based on the objective function value of the best individual found so far. The repulsion strategy aimed at exploring the remaining discontinuous feasible parts.

Moreover, when the population had converged to a discontinuous feasible part, it was reinitialized. Afterward, a repulsion function was designed to make the population repulse the explored discontinuous feasible parts, thus motivating the population to search for the optimal solution from other promising discontinuous feasible parts. From the comparative studies on 16 test problems, the effectiveness of CaR was verified. CaR could also be flexibly combined with other EAs to solve MIP problems. In addition, the results showed that the performance of CaR is better than that of three other state-of-the-art EAs. Moreover, we also applied CaR to solve two practical cases: the deployment optimization problem in the multi-UAV-assisted IoT data collection system and the path planing problem of the curvature-constrained UAV. The results showed that CaR has the capability to solve MIP problems in the real world.

The source code of this paper can be downloaded from: https://intleo.csu.edu.cn/publication.html.

REFERENCES

[1] R. Angira and B. V. Babu, "Optimization of process synthesis and design problems: A modified differential evolution approach," *Chemical Engineering Science*, vol. 61, no. 14, pp. 4707–4721, 2006.

[2] R. Balamurugan and S. Subramanian, "Hybrid integer coded differential evolution dynamic programming approach for economic load dispatch with multiple fuel options," *Energy Conversion & Management*, vol. 49, no. 4, pp. 608–614, 2008.

[3] X. F. Liu, Z. H. Zhan, J. D. Deng, Y. Li, and J. Zhang, "An energy efficient ant colony system for virtual machine placement in cloud computing," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 113–128, 2018.

[4] X. Y. Zhang, Y. J. Gong, Z. Zhan, W. N. Chen, Y. Li, and J. Zhang, "Kuhn-munkres parallel genetic algorithm for the set cover problem and its application to large-scale wireless sensor networks," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 5, pp. 695–710, 2016.

[5] C. Zhan, Y. Zeng, and R. Zhang, "Energy-efficient data collection in uav enabled wireless sensor network," *IEEE Wireless Communications Letters*, vol. 7, no. 3, pp. 328–331, 2018.

[6] H. Wu, C. Nie, F. C. Kuo, H. Leung, and C. J. Colbourn, "A discrete particle swarm optimization for covering array generation," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 4, pp. 575–591, 2015.

[7] V. Hinojosa and R. Araya, "Modeling a mixed-integer-binary small-population evolutionary particle swarm algorithm for solving the optimal power flow problem in electric power systems," *Applied Soft Computing*, vol. 13, no. 9, pp. 3839–3852, 2013.

[8] Y. Lin, Y. Liu, W.-N. Chen, and J. Zhang, "A hybrid differential evolution algorithm for mixed-variable optimization problems," *Information Sciences*, vol. 466, pp. 170–188, 2018.

[9] H.-Y. Zheng and L. Wang, "Reduction of carbon emissions and project makespan by a pareto-based estimation of distribution algorithm," *International Journal of Production Economics*, vol. 164, pp. 421–432, 2015.

[10] Q. K. Pan, L. Wang, K. Mao, J. H. Zhao, and M. Zhang, "An effective artificial bee colony algorithm for a real-world hybrid flowshop problem in steelmaking process," *IEEE Transactions on Automation Science & Engineering*, vol. 10, no. 2, pp. 307–322, 2013.

[11] G. Jia, Y. Wang, Z. Cai, and Y. Jin, "An improved (μ+ λ)-constrained differential evolution for constrained optimization," *Information Sciences*, vol. 222, pp. 302–322, 2013.

[12] D. W. Coit, A. E. Smith, and D. M. Tate, "Adaptive penalty methods for genetic optimization of constrained combinatorial problems," *INFORMS Journal on Computing*, vol. 8, no. 2, pp. 173–182, 1996.

[13] J. Liu, K. L. Teo, X. Wang, and C. Wu, "An exact penalty function-based differential search algorithm for constrained global optimization," *Soft Computing*, vol. 20, no. 4, pp. 1305–1313, 2016.

[14] V. V. D. Melo and G. Iacca, "A modified covariance matrix adaptation evolution strategy with adaptive penalty function and restart for constrained optimization," *Expert Systems with Applications*, vol. 41, no. 16, pp. 7077–7094, 2014.

[15] R. Farmani and J. A. Wright, "Self-adaptive fitness formulation for constrained optimization," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 5, pp. 445–455, 2003.

[16] 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.

[17] T. Takahama and S. Sakai, "Constrained optimization by applying the α constrained method to the nonlinear simplex method with mutations," *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 5, pp. 437–451, 2005.

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

[19] E. Mezura-Montes and C. A. C. Coello, "A simple multimembered evolution strategy to solve constrained optimization problems," *IEEE Transactions on Evolutionary computation*, vol. 9, no. 1, pp. 1–17, 2005.

[20] B.-C. Wang, H.-X. Li, J.-P. Li, and Y. Wang, "Composite differential evolution for constrained evolutionary optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, no. 99, pp. 1–14, 2018.

[21] Y. Wang, B.-C. Wang, H.-X. 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, 2016.

[22] C. A. Coello Coello, "Constraint-handling using an evolutionary multiobjective optimization technique," *Civil Engineering Systems*, vol. 17, no. 4, pp. 319–346, 2000.

[23] C. Fonseca and P. Fleming, "Multiobjective optimization and multiple constraint handling with evolutionary algorithms. I. A unified formulation," *IEEE Transactions on Systems Man & Cybernetics Part A: Systems and Humans*, vol. 28, no. 1, pp. 26–37, 2002.

[24] L. Jiao, L. Li, R. Shang, F. Liu, and R. Stolkin, "A novel selection evolutionary strategy for constrained optimization," *Information Sciences*, vol. 239, pp. 122–141, 2013.

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

[26] Y.-C. Lin, K.-S. Hwang, and F.-S. Wang, "Co-evolutionary hybrid differential evolution for mixed-integer optimization problems," *Engineering Optimization*, vol. 33, no. 6, pp. 663–682, 2001.

[27] J.-P. Chiou and F.-S. Wang, "A hybrid method of differential evolution with application to optimal control problems of a bioprocess system," in *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*. IEEE, 1998, pp. 627–632.

[28] K. Deep, K. P. Singh, M. Kansal, and C. Mohan, "A real coded genetic algorithm for solving integer and mixed integer optimization problems," *Applied Mathematics & Computation*, vol. 212, no. 2, pp. 505–518, 2009.

[29] R. Angira and B. Babu, "Optimization of process synthesis and design problems: A modified differential evolution approach," *Chemical Engineering Science*, vol. 61, no. 14, pp. 4707–4721, 2006.

[30] J. Wu, Y. Gao, and L. Yan, "An improved differential evolution algorithm for mixed integer programming problems," in *Computational Intelligence and Security (CIS), 2013 9th International Conference on*. IEEE, 2013, pp. 31–35.

[31] T. W. Liao, "Two hybrid differential evolution algorithms for engineering design optimization," *Applied Soft Computing*, vol. 10, no. 4, pp. 1188–1199, 2010.

[32] A. W. Mohamed, "An efficient modified differential evolution algorithm for solving constrained non-linear integer and mixed-integer global optimization problems," *International Journal of Machine Learning and Cybernetics*, vol. 8, no. 3, pp. 989–1007, 2017.

[33] Y. Luo, X. Yuan, and Y. Liu, "An improved PSO algorithm for solving non-convex NLP/MINLP problems with equality constraints," *Computers & Chemical Engineering*, vol. 31, no. 3, pp. 153–162, 2007.

[34] R. Li, M. T. Emmerich, J. Eggermont, T. Bäck, M. Schütz, J. Dijkstra, and J. H. Reiber, "Mixed integer evolution strategies for parameter optimization," *Evolutionary Computation*, vol. 21, no. 1, pp. 29–64, 2013.

[35] W.-C. Wu and M.-S. Tsai, "Application of enhanced integer coded particle swarm optimization for distribution system feeder reconfiguration," *IEEE Transactions on Power Systems*, vol. 26, no. 3, pp. 1591–1599, 2011.

[36] S. Y. Wang and L. Wang, "An estimation of distribution algorithm-based memetic algorithm for the distributed assembly permutation flow-shop scheduling problem," *IEEE Transactions on Systems Man & Cybernetics: Systems*, vol. 46, no. 1, pp. 139–149, 2015.

[37] J. Kennedy and R. C. Eberhart, “A discrete binary version of the particle swarm algorithm,” in *1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*, vol. 5. IEEE, 1997, pp. 4104–4108.
[38] D. Datta and J. R. Figueira, “A real-integer-discrete-coded particle swarm optimization for design problems,” *Applied Soft Computing*, vol. 11, no. 4, pp. 3625–3633, 2011.
[39] R. Storn and K. Price, “Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces,” *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.
[40] P.-Q. Huang, Y. Wang, K. Wang, and K. Yang, “Differential evolution with a variable population size for deployment optimization in a uav-assisted iot data collection system,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 4, no. 3, pp. 324–335, 2020.
[41] Y. Li, Z. Zhan, Y. Gong, W. Chen, J. Zhang, and Y. Li, “Differential evolution with an evolution path: A deep evolutionary algorithm,” *IEEE Transactions on Cybernetics*, vol. 45, no. 9, pp. 1798–1810, 2015.
[42] Q. Fan and X. Yan, “Self-adaptive differential evolution algorithm with zoning evolution of control parameters and adaptive mutation strategies,” *IEEE Transactions on Cybernetics*, vol. 46, no. 1, pp. 219–232, 2016.
[43] R. Storn and K. Price, “Differential evolution ? a simple and efficient heuristic for global optimization over continuous spaces,” *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.
[44] R. Storn, “System design by constraint adaptation and differential evolution,” *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 1, pp. 22–34, 1999.
[45] S. Das and P. N. Suganthan, “Differential evolution: A survey of the state-of-the-art,” *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 4–31, 2011.
[46] F. Caraffini, A. V. Kononova, and D. Corne, “Infeasibility and structural bias in differential evolution,” *Information Sciences*, vol. 496, pp. 161–179, 2019.
[47] J. Liang, T. P. Runarsson, E. Mezura-Montes, M. Clerc, P. N. Suganthan, C. C. Coello, and K. Deb, “Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization,” *Journal of Applied Mechanics*, vol. 41, no. 8, pp. 8–31, 2006.
[48] J. Zhang and A. C. Sanderson, “JADE: Adaptive differential evolution with optional external archive,” *IEEE Transaction on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958, 2009.
[49] P. Huang and Y. Wang, “A framework for scalable bilevel optimization: Identifying and utilizing the interactions between upper-level and lower-level variables,” *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2020.
[50] X. He, Y. Zhou, and Z. Chen, “Evolutionary bilevel optimization based on covariance matrix adaptation,” *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 2, pp. 258–272, 2019.
[51] X. Liu, “A deployment strategy for multiple types of requirements in wireless sensor networks,” *IEEE Transactions on Cybernetics*, vol. 45, no. 10, pp. 2364–2376, 2015.
[52] Y. Wang, Z. Ru, K. Wang, and P. Huang, “Joint deployment and task scheduling optimization for large-scale mobile users in multi-uav-enabled mobile edge computing,” *IEEE Transactions on Cybernetics*, vol. 50, no. 9, pp. 3984–3997, 2020.
[53] X. Zhang, J. Chen, B. Xin, and Z. Peng, “A memetic algorithm for path planning of curvature-constrained uavs performing surveillance of multiple ground targets,” *Chinese Journal of Aeronautics*, vol. 27, no. 3, pp. 622 – 633, 2014.
[54] J. Drchal, J. Faigl, and P. V?a, “Wism: Windowing surrogate model for evaluation of curvature-constrained tours with dubins vehicle,” *IEEE Transactions on Cybernetics*, pp. 1–10, 2020.
[55] D. X. B. Xin, J. Chen and Y. Chen, “Hybrid encoding based differential evolution algorithms for dubins traveling salesman problem with neighborhood,” *American Journal of Mathematics*, vol. 79, no. 3, pp. 497–516, 1957.
[56] L. E. Dubins, “On curves of minimal length with a constraint on average curvature and with prescribed initial and terminal positions and tangents,” *American Journal of Mathematics*, vol. 79, no. 3, pp. 497–516, 1957.

Jiao Liu received the B.S. degree in process equipment and control engineering and the M.S. degree in power engineering and engineering thermophysics both from the Taiyuan University of Technology, Taiyuan, China, in 2013 and 2016, respectively. He is currently pursuing the Ph.D. degree in control science and engineering, Central South University, Changsha, China.

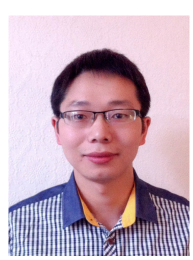
His current research interests include evolutionary computation, mixed-integer programming, and automotive lightweight design.

Yong Wang (M’08–SM’17) received the Ph.D. degree in control science and engineering from the Central South University, Changsha, China, in 2011.

He is a Professor with the School of Automation, Central South University, Changsha, China. His current research interests include intelligent learning and optimization and their interdisciplinary applications.

Dr. Wang is an Associate Editor for the *IEEE Transactions on Evolutionary Computation* and the *Swarm and Evolutionary Computation*. He was a recipient of Cheung Kong Young Scholar in 2018 and a Web of Science highly cited researcher in Computer Science in 2017 and 2018.

Pei-Qiu Huang received the B.S. degree in automation and the M.S. degree in control theory and control engineering both from the Northeastern University, Shenyang, China, in 2014 and 2017, respectively, and the Ph.D. degree in control science and engineering, Central South University, Changsha, China, in 2021. His current research interests include evolutionary computation, bilevel optimization, and mobile edge computing.



Shouyong Jiang received the B.Sc. degree in information and computation science and the M.Sc. degree in control theory and control engineering from in Northeastern University, China in 2011 and 2013, respectively, and the Ph.D. degree in computer science from De Montfort University, UK in 2017. He is now a Lecturer in Computer Science at the University of Aberdeen, UK. His current research interests include AI optimisation, evolutionary computation, machine learning, and data science.

Supplementary File for “CaR: A Cutting and Repulsion-based Evolutionary Framework for Mixed-Integer Programming Problems”

S-I. Test Problems F1-F16

$$
\begin{aligned}
\textbf{F1}:\ & \mathit{minimize}:\ f(x,y)=(x-1)^2+(y-3)^2\\
& \mathit{subject\ to}:\\
& \qquad g(x,y)=(x+1)^2+(y+1)^2-1\le 0\\
& \qquad x\in[-3,1]\\
& \qquad y\in\{-3,-2,-1,0,1\}
\end{aligned}
$$

The optimal solution is $x^*=-1$ and $y^*=0$, and $f(x^*,y^*)=13.0000$.

$$
\begin{aligned}
\textbf{F2}:\ & \mathit{minimize}:\ f(x,\mathbf{y})=x^2+(y_1-1)^2+(y_2-2)^2\\
& \mathit{subject\ to}:\\
& \qquad g(x,\mathbf{y})=x^2+y_1^2+0.5y_2^2-1.5\le 0\\
& \qquad x\in[-1,100]\\
& \qquad y_1,y_2\in\{-1,0,\ldots,100\}
\end{aligned}
$$

The optimal solution is $x^*=0$ and $\mathbf{y}^*=(1,1)$, and $f(x^*,\mathbf{y}^*)=1.0000$.

$$
\begin{aligned}
\textbf{F3}:\ & \mathit{minimize}:\ f(x,y)=-x-y\\
& \mathit{subject\ to}:\\
& \qquad g_1(x,y)=-x+y-2.005\le 0\\
& \qquad g_2(x,y)=x-y+0.5\le 0\\
& \qquad g_3(x,y)=0.505x+y-3.505\le 0\\
& \qquad x\in[-1,100]\\
& \qquad y\in\{-1,0,\ldots,100\}
\end{aligned}
$$

The optimal solution is $x^*=1$ and $y^*=3$, and $f(x^*,y^*)=-4.0000$.

$$
\begin{aligned}
\textbf{F4}:\ & \mathit{minimize}:\ f(x,y)=-x-y\\
& \mathit{subject\ to}:\\
& \qquad g_1(x,y)=y-3.4\le 0\\
& \qquad g_2(x,y)=x-y\le 0\\
& \qquad x\in[-1,100]\\
& \qquad y\in\{-1,0,\ldots,100\}
\end{aligned}
$$

The optimal solution is $x^*=3$ and $y^*=3$, and $f(x^*,y^*)=-6.0000$.

$$
\begin{aligned}
\textbf{F5}:\ & \mathit{minimize}:\ f(x,y)=(x-0.5)^2+(y-1)^2\\
& \mathit{subject\ to}:\\
& \qquad h(x,y)=-x^2+y=0\\
& \qquad x\in[-1,3.1]\\
& \qquad y\in\{-1,0,\ldots,4\}
\end{aligned}
$$

The best known optimal solution is $x^* = 1$ and $y^* = 1$, and $f(x^*, y^*) = 0.2500$.

$$
\begin{aligned}
\textbf{F6}:\ & minimize:\ \ f(x,y) = (x-10)^3 + (y-20)^3 \\
& subject\ to: \\
& \qquad\qquad g_1(x,y) = -(x-5)^2 - (y-4.86)^2 + 100 \le 0 \\
& \qquad\qquad g_2(x,y) = (x-8)^2 + (y-5.48)^2 - 60 \le 0 \\
& \qquad\qquad x \in [-1, 100] \\
& \qquad\qquad y \in \{-1, 0, \ldots, 100\}
\end{aligned}
$$

The best known optimal solution is $x^* = 14.22498780$ and $y^* = 1$, and $f(x^*, y^*) = -6783.5818$.

$$
\begin{aligned}
\textbf{F7}:\ & minimize:\ \ f(\mathbf{x},\mathbf{y}) = exp(x_1x_2x_3y_1y_2) \\
& subject\ to: \\
& \qquad\qquad h_1(\mathbf{x},\mathbf{y}) = x_1^2 + x_2^2 + x_3^2 + y_1^2 + y_2^2 - 10 = 0 \\
& \qquad\qquad h_2(\mathbf{x},\mathbf{y}) = x_2y_1 - 5x_3y_2 = 0 \\
& \qquad\qquad h_3(\mathbf{x},\mathbf{y}) = x_1^3 + y_1^3 + 1 = 0 \\
& \qquad\qquad x_1 \in [-2.3, 2.3] \\
& \qquad\qquad x_2, x_3 \in [-3.2, 3.2] \\
& \qquad\qquad y_1 \in \{-2, -1, 0, 1, 2\} \\
& \qquad\qquad y_2 \in \{-3, -2, -1, 0, 1, 2, 3\}
\end{aligned}
$$

The best known optimal solution is $\mathbf{x}^* = (-1.25994205, -2.48314049, 0.496648098)$ and $\mathbf{y}^* = (1, -1)$, and $f(\mathbf{x}^*, \mathbf{y}^*) = 0.2114$.

$$
\begin{aligned}
\textbf{F8}:\ & minimize:\ f(\mathbf{x},\mathbf{y}) = x_1 + x_2 + y_1 \\
& subject\ to: \\
& \qquad\qquad g_1(\mathbf{x},\mathbf{y}) = -1 + 0.0025(x_3 + x_4) \le 0 \\
& \qquad\qquad g_2(\mathbf{x},\mathbf{y}) = -1 + 0.0025(-x_3 + y_2 + y_3) \le 0 \\
& \qquad\qquad g_3(\mathbf{x},\mathbf{y}) = -1 + 0.01(x_5 - y_2) \le 0 \\
& \qquad\qquad g_4(\mathbf{x},\mathbf{y}) = -x_1x_4 + 833.33252x_3 + 100x_1 - 83333.333 \le 0 \\
& \qquad\qquad g_5(\mathbf{x},\mathbf{y}) = -y_1y_3 + 1250y_2 + x_3y_1 - 1250x_3 \le 0 \\
& \qquad\qquad g_6(\mathbf{x},\mathbf{y}) = -x_2x_5 + 1250000 + x_2y_2 - 2500y_2 \le 0 \\
& \qquad\qquad x_1 \in [100, 10000] \\
& \qquad\qquad x_2 \in [1000, 10000] \\
& \qquad\qquad x_3, x_4, x_5 \in [10, 1000] \\
& \qquad\qquad y_1 \in \{1000, 1020, \ldots, 10000\} \\
& \qquad\qquad y_2, y_3 \in \{20, 40, \ldots, 1000\}
\end{aligned}
$$

The best known optimal solution is $\mathbf{x}^* = (555.55433833, 5000, 180, 220, 400)$ and $\mathbf{y}^* = (1500, 300, 280)$, and $f(\mathbf{x}^*, \mathbf{y}^*) = 7055.5544$.

$$
\begin{aligned}
\textbf{F9}:\ & minimize:\ f(\mathbf{x},\mathbf{y}) = x_1 + x_2 + y_1 \\
& subject\ to: \\
& \qquad\qquad g_1(\mathbf{x},\mathbf{y}) = -1 + 0.0025(x_3 + x_4) \le 0 \\
& \qquad\qquad g_2(\mathbf{x},\mathbf{y}) = -1 + 0.0025(-x_3 + y_2 + y_3) \le 0 \\
& \qquad\qquad g_3(\mathbf{x},\mathbf{y}) = -1 + 0.01(x_5 - y_2) \le 0 \\
& \qquad\qquad g_4(\mathbf{x},\mathbf{y}) = -x_1x_4 + 833.33252x_3 + 100x_1 - 83333.333 \le 0 \\
& \qquad\qquad g_5(\mathbf{x},\mathbf{y}) = -y_1y_3 + 1250y_2 + x_3y_1 - 1250x_3 \le 0 \\
& \qquad\qquad g_6(\mathbf{x},\mathbf{y}) = -x_2x_5 + 1250000 + x_2y_2 - 2500y_2 \le 0 \\
& \qquad\qquad x_1 \in [100, 10000]
\end{aligned}
$$

$$
\begin{aligned}
& x_2 \in [1000, 10000] \\
& x_3, x_4, x_5 \in [10, 1000] \\
& y_1 \in \{1000, 1050, \ldots, 10000\} \\
& y_2, y_3 \in \{50, 100, \ldots, 1000\}
\end{aligned}
$$

The best known optimal solution is $\mathbf{x}^* = (833.33171, 5000, 200, 200, 400)$ and $\mathbf{y}^* = (1250, 300, 300)$, and $f(\mathbf{x}^*, \mathbf{y}^*) = 7083.3317$.

$$
\begin{aligned}
\textbf{F10}:\ & \textit{minimize}: f(\mathbf{x}, \mathbf{y}) = x_1 + x_2 + y_1 \\
& \textit{subject to}: \\
& \qquad g_1(\mathbf{x}, \mathbf{y}) = -1 + 0.0025(x_3 + x_4) \leq 0 \\
& \qquad g_2(\mathbf{x}, \mathbf{y}) = -1 + 0.0025(-x_3 + y_2 + y_3) \leq 0 \\
& \qquad g_3(\mathbf{x}, \mathbf{y}) = -1 + 0.01(x_5 - y_2) \leq 0 \\
& \qquad g_4(\mathbf{x}, \mathbf{y}) = -x_1 x_4 + 833.33252 x_3 + 100 x_1 - 83333.333 \leq 0 \\
& \qquad g_5(\mathbf{x}, \mathbf{y}) = -y_1 y_3 + 1250 y_2 + x_3 y_1 - 1250 x_3 \leq 0 \\
& \qquad g_6(\mathbf{x}, \mathbf{y}) = -x_2 x_5 + 1250000 + x_2 y_2 - 2500 y_2 \leq 0 \\
& \qquad x_1 \in [100, 10000] \\
& \qquad x_2 \in [1000, 10000] \\
& \qquad x_3, x_4, x_5 \in [10, 1000] \\
& \qquad y_1 \in \{1000, 1100, \ldots, 10000\} \\
& \qquad y_2, y_3 \in \{100, 200, \ldots, 1000\}
\end{aligned}
$$

The best known optimal solution is $\mathbf{x}^* = (833.33171, 5000, 200, 200, 400)$ and $\mathbf{y}^* = (1300, 300, 300)$, and $f(\mathbf{x}^*, \mathbf{y}^*) = 7133.3317$.

$$
\begin{aligned}
\textbf{F11}:\ & \textit{minimize}: \ f(\mathbf{x}) = \sum_{j=1}^{5} \sum_{i=1}^{5} c_{ij} x_{(10+i)} x_{(10+j)} + 2 \sum_{j=1}^{5} d_j x_{(10+j)}^3 - \sum_{i=1}^{10} b_i x_i \\
& \textit{subject to}: \\
& \qquad g_j(\mathbf{x}) = -2 \sum_{i=1}^{5} c_{ij} x_{(10+i)} - 3 d_j x_{(10+j)}^2 - e_j + \sum_{i=1}^{10} a_{ij} x_i \leq 0, j = 1, \ldots, 5 \\
& \qquad x_1, x_2, x_4, x_6, \ldots, x_{11}, x_{13}, x_{14}, x_{15} \in [0, 10] \\
& \qquad x_3, x_5, x_{12} \in \{0, 1, \ldots, 10\}
\end{aligned}
$$

where $\mathbf{b} = [-40, -2, -0.25, -4, -4, -1, -40, -60, 5, 1]$ and the remaining data is in Table S-I. The best known optimal solution is $\mathbf{x}^* = (x_1, x_2, x_4, x_6, \ldots, x_{11}, x_{13}, x_{14}, x_{15}) = (0, 0, 0, 9.99999985, 0, 0, 0, 0, 0.28879805, 0.43951302, 0.31935496, 0.44885950)$ and $\mathbf{y}^* = (x_3, x_5, x_{12}) = (4, 4, 0)$, and $f(\mathbf{x}^*, \mathbf{y}^*) = 33.5066$.

$$
\begin{aligned}
\textbf{F12}:\ & \textit{minimize}: \ f(\mathbf{x}) = \sum_{j=1}^{5} \sum_{i=1}^{5} c_{ij} x_{(10+i)} x_{(10+j)} + 2 \sum_{j=1}^{5} d_j x_{(10+j)}^3 - \sum_{i=1}^{10} b_i x_i \\
& \textit{subject to}: \\
& \qquad g_j(\mathbf{x}) = -2 \sum_{i=1}^{5} c_{ij} x_{(10+i)} - 3 d_j x_{(10+j)}^2 - e_j + \sum_{i=1}^{10} a_{ij} x_i \leq 0, j = 1, \ldots, 5 \\
& \qquad x_1, x_2, x_4, x_6, \ldots, x_9, x_{11}, x_{13}, x_{14} \in [0, 10] \\
& \qquad x_3, x_5, x_{10}, x_{12}, x_{15} \in \{0, 1, \ldots, 10\}
\end{aligned}
$$

where $\mathbf{b} = [-40, -2, -0.25, -4, -4, -1, -40, -60, 5, 1]$ and the remaining data is in Table S-I. The best known optimal solution is $\mathbf{x}^* = (x_1, x_2, x_4, x_6, \ldots, x_9, x_{11}, x_{13}, x_{14}) = (0, 0, 0, 9.99999999, 0, 0, 2.96750117, 0.39963905, 0.82151768, 0.64848398)$

TABLE S-I
DATASET FOR TEST PROBLEMS F11 AND F12

j	1	2	3	4	5
e_j	-15	-27	-36	-18	-12
c_{1j}	30	-20	-10	32	-10
c_{2j}	-20	39	-6	-31	32
c_{3j}	-10	-6	10	-6	-10
c_{4j}	32	-31	-6	39	-20
c_{5j}	-10	32	-10	-20	30
d_j	4	8	10	6	2
a_{1j}	-16	2	0	1	0
a_{2j}	0	-2	0	0.4	2
a_{3j}	-3.5	0	2	0	0
a_{4j}	0	-2	0	-4	-1
a_{5j}	0	-9	-2	1	-2.8
a_{6j}	2	0	-4	0	0
a_{7j}	-1	-1	-1	-1	-1
a_{8j}	-1	-2	-3	-2	-1
a_{9j}	1	2	3	4	5
a_{10j}	1	1	1	1	1

and $\mathbf{y}^* = (x_3, x_5, x_{10}, x_{12}, x_{15}) = (2, 4, 0, 0, 1)$, and $f(\mathbf{x}^*, \mathbf{y}^*) = 41.7399$.

$$
\begin{aligned}
\textbf{F13}: \ & minimize: \ f(\mathbf{x},\mathbf{y}) = f_1(y_1) + f_2(x_1) \\
& \quad where: \\
& \qquad f_1(y_1) = \begin{cases} 30y_1, & 0 \le y_1 \le 300 \\ 31y_1, & 300 \le y_1 \le 400 \end{cases} \\
& \qquad f_2(x_1) = \begin{cases} 28x_1, & 0 \le x_1 \le 100 \\ 29x_1, & 100 \le x_1 \le 200 \\ 30x_1, & 200 \le x_1 \le 1000 \end{cases} \\
& subject\ to: \\
& \qquad h_1(\mathbf{x},\mathbf{y}) = -y_1 + 300 - \frac{y_2 x_2}{131.078}\cos(1.48477 - x_4) + \frac{0.90798 y_2^2}{131.078}\cos(1.47588) = 0 \\
& \qquad h_2(\mathbf{x},\mathbf{y}) = -x_1 - \frac{y_2 x_2}{131.078}\cos(1.48477 + x_4) + \frac{0.90798 x_2^2}{131.078}\cos(1.47588) = 0 \\
& \qquad h_3(\mathbf{x},\mathbf{y}) = -x_3 - \frac{y_2 x_2}{131.078}\sin(1.48477 + x_4) + \frac{0.90798 x_2^2}{131.078}\sin(1.47588) = 0 \\
& \qquad h_4(\mathbf{x},\mathbf{y}) = 200 - \frac{y_2 x_2}{131.078}\sin(1.48477 - x_4) + \frac{0.90798 y_2^2}{131.078}\sin(1.47588) = 0 \\
& \qquad x_1 \in [0, 1000] \\
& \qquad x_2 \in [340, 420] \\
& \qquad x_3 \in [-1000, 1000] \\
& \qquad x_4 \in [0, 0.5236] \\
& \qquad y_1 \in \{0, 20, \ldots, 400\} \\
& \qquad y_2 \in \{340, 360, \ldots, 420\}
\end{aligned}
$$

The best known optimal solution is $\mathbf{x}^* = (81.57454322, 416.85149297, -9.77394390, 0.05912763)$ and $\mathbf{y}^* = (220, 380)$, and $f(\mathbf{x}^*, \mathbf{y}^*) = 8884.0872$.

$$
\begin{aligned}
\textbf{F14}: \ & minimize: \ f(\mathbf{x},\mathbf{y}) = f_1(y_1) + f_2(x_1) \\
& \quad where: \\
& \qquad f_1(y_1) = \begin{cases} 30y_1, & 0 \le y_1 \le 300 \\ 31y_1, & 300 \le y_1 \le 400 \end{cases} \\
& \qquad f_2(x_1) = \begin{cases} 28x_1, & 0 \le x_1 \le 100 \\ 29x_1, & 100 \le x_1 \le 200 \\ 30x_1, & 200 \le x_1 \le 1000 \end{cases}
\end{aligned}
$$

$$
\begin{aligned}
&\textit{subject to}:\\
&\quad h_1(\mathbf{x},\mathbf{y}) = -y_1 + 300 - \frac{y_2 x_2}{131.078}\cos(1.48477 - x_4) + \frac{0.90798 y_2^2}{131.078}\cos(1.47588) = 0\\
&\quad h_2(\mathbf{x},\mathbf{y}) = -x_1 - \frac{y_2 x_2}{131.078}\cos(1.48477 + x_4) + \frac{0.90798 x_2^2}{131.078}\cos(1.47588) = 0\\
&\quad h_3(\mathbf{x},\mathbf{y}) = -x_3 - \frac{y_2 x_2}{131.078}\sin(1.48477 + x_4) + \frac{0.90798 x_2^2}{131.078}\sin(1.47588) = 0\\
&\quad h_4(\mathbf{x},\mathbf{y}) = 200 - \frac{y_2 x_2}{131.078}\sin(1.48477 - x_4) + \frac{0.90798 y_2^2}{131.078}\sin(1.47588) = 0\\
&\quad x_1 \in [0, 1000]\\
&\quad x_2 \in [340, 420]\\
&\quad x_3 \in [-1000, 1000]\\
&\quad x_4 \in [0, 0.5236]\\
&\quad y_1 \in \{0, 50, \ldots, 400\}\\
&\quad y_2 \in \{350, 400\}
\end{aligned}
$$

The best known optimal solution is $\mathbf{x}^* = (51.69905661, 394.30118556, 20.47601024, 0.03816719)$ and $\mathbf{y}^* = (250, 350)$, and $f(\mathbf{x}^*, \mathbf{y}^*) = 8947.5736$.

$$
\begin{aligned}
\textbf{F15}: \ \textit{minimize}: \ f(\mathbf{x},\mathbf{y}) &= x_1^2 + y_1^2 + x_1 y_1 - 14x_1 - 16y_1 + (y_2 - 10)^2 + 4(x_2 - 5)^2 + (x_3 - 3)^2\\
&\quad + 2(x_4 - 1)^2 + 5x_5^2 + 7(x_6 - 11)^2 + 2(y_3 - 10)^2 + (x_7 - 7)^2 + 45\\
\textit{subject to}:&\\
&g_1(\mathbf{x},\mathbf{y}) = -105 + 4x_1 + 5y_1 - 3x_5 + 9x_6 \le 0\\
&g_2(\mathbf{x},\mathbf{y}) = 10x_1 - 8y_1 - 17x_5 + 2x_6 \le 0\\
&g_3(\mathbf{x},\mathbf{y}) = -8x_1 + 2y_1 + 5y_3 - 2x_7 - 12 \le 0\\
&g_4(\mathbf{x},\mathbf{y}) = 3(x_1 - 2)^2 + 4(y_1 - 3)^2 + 2y_2^2 - 7x_2 - 120 \le 0\\
&g_5(\mathbf{x},\mathbf{y}) = 5x_1^2 + 8y_1 + (y_2 - 6)^2 - 2x_2 - 40 \le 0\\
&g_6(\mathbf{x},\mathbf{y}) = x_1^2 + 2(y_1 - 2)^2 - 2x_1 y_1 + 14x_3 - 6x_4 \le 0\\
&g_7(\mathbf{x},\mathbf{y}) = 0.5(x_1 - 8)^2 + 2(y_1 - 4)^2 + 3x_3^2 - x_4^2 - 30 \le 0\\
&g_8(\mathbf{x},\mathbf{y}) = -3x_1 + 6y_1 + 12(y_3 - 8)^2 - 7x_7 \le 0\\
&x_1, x_2, \ldots, x_7 \in [-10, 10]\\
&y_1, y_2, y_3 \in \{-10, -9, \ldots, 10\}
\end{aligned}
$$

The best known optimal solution is $\mathbf{x}^* = (2.45799944, 5.10440319, 0.89287364, 1.45166575, 1.68117614, 9.99999999, 8.66800226)$ and $\mathbf{y}^* = (2, 8, 9)$, and $f(\mathbf{x}^*, \mathbf{y}^*) = 28.3514$.

$$
\begin{aligned}
\textbf{F16}: \ \textit{minimize}: \ f(\mathbf{x},\mathbf{y}) &= x_1^2 + y_1^2 + x_1 y_1 - 14x_1 - 16y_1 + (y_2 - 10)^2 + 4(x_2 - 5)^2 + (y_3 - 3)^2\\
&\quad + 2(x_3 - 1)^2 + 5x_4^2 + 7(y_4 - 11)^2 + 2(y_5 - 10)^2 + (x_5 - 7)^2 + 45\\
\textit{subject to}:&\\
&g_1(\mathbf{x},\mathbf{y}) = -105 + 4x_1 + 5y_1 - 3x_4 + 9y_4 \le 0\\
&g_2(\mathbf{x},\mathbf{y}) = 10x_1 - 8y_1 - 17x_4 + 2y_4 \le 0\\
&g_3(\mathbf{x},\mathbf{y}) = -8x_1 + 2y_1 + 5y_5 - 2x_5 - 12 \le 0\\
&g_4(\mathbf{x},\mathbf{y}) = 3(x_1 - 2)^2 + 4(y_1 - 3)^2 + 2y_2^2 - 7x_2 - 120 \le 0\\
&g_5(\mathbf{x},\mathbf{y}) = 5x_1^2 + 8y_1 + (y_2 - 6)^2 - 2x_2 - 40 \le 0\\
&g_6(\mathbf{x},\mathbf{y}) = x_1^2 + 2(y_1 - 2)^2 - 2x_1 y_1 + 14y_3 - 6x_3 \le 0\\
&g_7(\mathbf{x},\mathbf{y}) = 0.5(x_1 - 8)^2 + 2(y_1 - 4)^2 + 3y_3^2 - x_3^2 - 30 \le 0\\
&g_8(\mathbf{x},\mathbf{y}) = -3x_1 + 6y_1 + 12(y_5 - 8)^2 - 7x_5 \le 0\\
&x_1, x_2, \ldots, x_5 \in [-10, 10]\\
&y_1, y_2, \ldots, y_5 \in \{-10, -9, \ldots, 10\}
\end{aligned}
$$

The best known optimal solution is $\mathbf{x}^* = (2.45787583, 5.10288399, 1.70160838, 1.68110343, 8.66849668)$ and $\mathbf{y}^* = (2, 8, 1, 10, 9)$, and $f(\mathbf{x}^*, \mathbf{y}^*) = 28.4879$.

S-II. Results

TABLE S-II

Results of DE/rand/1/bin-MIP, DE/rand/2/bin-MIP, DE/current-to-rand/bin-MIP, DE/rand-to-best/bin-MIP, DE/rand/1/bin-CaR, DE/rand/2/bin-CaR, DE/current-to-rand/bin-CaR, and DE/rand-to-best/bin-CaR over 25 Independent Runs. *FR* and *SR* Indicate the Feasible Rate and Successful Rate, Respectively. *Ave* and *Std Dev* Indicate the Average and Standard Deviation of the Best Feasible Objective Function Values over 25 Independent Runs, Respectively. *ACT* Indicates the Average CPU Time over 25 Independent Runs. The Wilcoxon's Rank-sum Test at a 0.05 Significance Level Is Performed between DE/rand/1/bin-MIP and DE/rand/1/bin-CaR,DE/rand/2/bin-MIP and DE/rand/2/bin-CaR, DE/current-to-rand/bin-MIP and DE/current-to-rand/bin-CaR, and DE/rand-to-best/bin-MIP and DE/rand-to-best/bin-CaR.

Problem	Status	DE/rand/1/bin-MIP		DE/rand/1/bin-CaR	DE/rand/2/bin-MIP		DE/rand/2/bin-CaR	DE/current-to-rand/bin-MIP		DE/current-to-rand/bin-CaR	DE/rand-to-best/bin-MIP		DE/rand-to-best/bin-CaR
F1	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	0.00%		**100.00%**	0.00%		**100.00%**	0.00%		**100.00%**	0.00%		**100.00%**
	Ave	17.0000		**13.0000**	17.0000		**13.0000**	17.0000		**13.0000**	17.0000		**13.0000**
	Std Dev	0.0000	$+$	0.0000	0.0000	$+$	0.0000	0.0000	$+$	0.0000	0.0000	$+$	0.0000
	ACT	4.16		5.05	4.13		5.04	4.22		5.07	4.27		5.18
F2	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	0.00%		**100.00%**	0.00%		**100.00%**	0.00%		**100.00%**	0.00%		**100.00%**
	Ave	2.0000		**1.0000**	2.0000		**1.0000**	2.0000		**1.0000**	2.0000		**1.0000**
	Std Dev	0.0000	$+$	0.0000	0.0000	$+$	0.0000	0.0000	$+$	0.0000	0.0000	$+$	0.0000
	ACT	4.55		6.25	4.54		6.19	4.62		6.33	4.69		6.35
F3	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	24.00%		**100.00%**	32.00%		**100.00%**	12.00%		**100.00%**	8.00%		**100.00%**
	Ave	-3.6200		**-4.0000**	-3.6600		**-4.0000**	-3.5800		**-4.0000**	-3.5400		**-4.0000**
	Std Dev	0.2179	$+$	0.0000	0.2380	$+$	0.0000	0.1871	$+$	0.0000	0.1384	$+$	0.0000
	ACT	4.73		5.43	4.77		5.40	4.72		5.39	4.80		5.82
F4	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	Ave	**-6.0000**		**-6.0000**	**-6.0000**		**-6.0000**	**-6.0000**		**-6.0000**	**-6.0000**		**-6.0000**
	Std Dev	0.0000	$\approx$	0.0000	0.0000	$\approx$	0.0000	0.0000	$\approx$	0.0000	0.0000	$\approx$	0.0000
	ACT	4.71		5.68	4.74		5.67	4.74		5.71	4.77		5.69
F5	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	4.00%		**100.00%**	0.00%		**100.00%**	0.00%		**100.00%**	0.00%		**100.00%**
	Ave	1.2005		**0.2500**	1.2401		**0.2500**	1.2401		**0.2500**	1.2401		**0.2500**
	Std Dev	0.1980	$+$	0.0000	0.0000	$+$	0.0000	0.0000	$+$	0.0000	0.0000	$+$	0.0000
	ACT	4.17		5.34	4.17		5.33	4.24		5.39	4.31		5.36
F6	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	52.00%		**100.00%**	88.00%		**100.00%**	24.00%		**100.00%**	96.00%		**100.00%**
	Ave	-6205.8268		**-6783.5818**	-6657.8460		**-6783.5818**	-5875.0374		**-6783.5818**	-6741.6698		**-6783.5818**
	Std Dev	661.3729	$+$	0.0000	347.5154	$\approx$	0.0000	602.9786	$+$	0.0000	209.5596	$\approx$	0.0000
	ACT	4.73		5.89	4.72		5.85	4.81		5.95	4.87		5.97
F7	*FR*	**100.00%**		**100.00%**	84.00%		**92.00%**	92.00%		**100.00%**	**100.00%**		**100.00%**
	SR	**0.00%**		**0.00%**	**0.00%**		**0.00%**	**0.00%**		**0.00%**	0.00%		**4.00%**
	Ave	0.9917		**0.9311**	*NA*		*NA*	*NA*		**0.9945**	0.9862		**0.8503**
	Std Dev	0.0229	$+$	0.1192	*NA*	$\approx$	*NA*	*NA*	$+$	0.0191	0.0281	$+$	0.2309
	ACT	4.99		6.69	4.59		5.80	4.43		5.77	5.26		7.22
F8	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**	96.00%		**100.00%**
	SR	0.00%		**48.00%**	0.00%		**100.00%**	0.00%		**28.00%**	**0.00%**		**0.00%**
	Ave	7327.6789		**7097.5704**	7097.8528		**7055.5543**	7235.9826		**7081.6662**	*NA*		**7400.7217**
	Std Dev	235.1243	$+$	47.0169	41.8793	$+$	0.0000	160.2011	$+$	31.4088	*NA*	$+$	831.5231
	ACT	5.03		6.19	4.70		5.38	5.01		5.90	5.33		6.76
F9	FR	**100.00%**		**100.00%**	**100.00%**		**100.00%**	96.00%		**100.00%**	92.00%		**100.00%**
	SR	0.00%		**20.00%**	0.00%		**32.00%**	0.00%		**52.00%**	8.00%		**16.00%**
	Ave	7912.2222		**7226.4436**	7567.3332		**7120.1722**	*NA*		**7138.8877**	*NA*		**7645.3336**
	Std Dev	451.1959	$+$	170.5967	172.0754	$+$	44.8031	*NA*	$+$	93.4454	*NA*	$+$	791.3634
	ACT	5.08		6.36	4.89		5.69	5.15		6.25	5.33		6.74
F10	*FR*	96.00%		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**	88.00%		**100.00%**
	SR	4.00%		**56.00%**	0.00%		**36.00%**	0.00%		**80.00%**	12.00%		**24.00%**
	Ave	*NA*		**7338.6658**	8600.0000		**7413.3327**	7772.0000		**7268.4890**	*NA*		**7682.6703**
	Std Dev	*NA*	$+$	236.4248	474.3416	$+$	233.3341	249.1987	$+$	329.7006	*NA*	$+$	565.1227
	ACT	5.14		6.38	5.08		5.93	5.20		6.45	5.34		6.87
F11	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	4.00%		**16.00%**	24.00%		**28.00%**	**0.00%**		**0.00%**	**0.00%**		**0.00%**
	Ave	74.5033		**43.3592**	40.9682		**36.6424**	73.5353		**46.1278**	79.2500		**77.1227**
	Std Dev	53.1960	$+$	6.9833	7.2714	$+$	4.4466	61.9544	$\approx$	9.2244	58.0917	$\approx$	41.5648
	ACT	10.57		11.88	10.35		11.33	10.75		11.54	10.78		12.50
F12	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	0.00%		**4.00%**	28.00%		**44.00%**	0.00%		**4.00%**	**0.00%**		**0.00%**
	Ave	161.2159		**47.0600**	74.7723		**42.1174**	178.8217		**63.6321**	122.3971		**119.3997**
	Std Dev	89.3781	$+$	4.9272	26.8670	$+$	0.6135	63.1152	$+$	25.9556	44.4599	$\approx$	15.5185
	ACT	10.64		12.08	10.40		11.51	10.76		11.74	10.73		12.49
F13	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	8.00%		**56.00%**	0.00%		**20.00%**	0.00%		**8.00%**	8.00%		**24.00%**
	Ave	9002.8026		**8892.1970**	8966.8064		**8904.9290**	8934.5410		**8915.4794**	9001.2513		**8919.8126**
	Std Dev	125.6644	$+$	12.9662	83.7794	$+$	22.5612	24.1431	$+$	28.0801	111.2674	$+$	27.8651
	ACT	5.05		6.51	4.62		5.72	4.46		5.44	5.35		7.09
F14	*FR*	96.00%		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**	76.00%		**100.00%**
	SR	16.00%		**44.00%**	8.00%		**80.00%**	4.00%		**28.00%**	8.00%		**52.00%**
	Ave	*NA*		**8954.4795**	8981.1015		**8948.6209**	8962.6657		**8951.3441**	*NA*		**8958.8276**
	Std Dev	*NA*	$+$	8.8424	74.2260	$+$	2.1379	40.4182	$+$	2.3998	*NA*	$+$	41.0848
	ACT	5.05		6.59	4.73		5.90	4.46		5.77	5.34		7.13
F15	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	8.00%		**40.00%**	36.00%		**48.00%**	**8.00%**		**8.00%**	**0.00%**		**0.00%**
	Ave	30.0639		**28.6154**	28.9027		**28.6649**	**30.9901**		33.6415	31.5377		**29.4806**
	Std Dev	1.9622	$\approx$	0.2021	0.9408	$+$	0.1798	3.1633	$\approx$	13.6143	3.2770	$\approx$	1.6260
	ACT	4.63		6.13	4.58		5.86	4.79		5.68	4.62		6.44
F16	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	32.00%		**100.00%**	88.00%		**100.00%**	12.00%		**52.00%**	16.00%		**40.00%**
	Ave	32.6737		**28.4879**	28.7992		**28.4879**	33.8383		**29.6948**	42.8956		**30.9216**
	Std Dev	5.7933	$+$	0.0000	0.8604	$+$	0.0000	5.0148	$+$	2.0945	8.4415	$+$	3.4193
	ACT	4.86		6.43	4.62		6.24	4.80		6.39	4.65		6.70
$+/\approx/-$		14/2/0			13/3/0			13/3/0			11/5/0		

TABLE S-III

RESULTS OF DE/RAND/2/BIN-WOCR, DE/RAND/2/BIN-WOC, DE/RAND/2/BIN-WOR, AND DE/RAND/2/BIN-CAR OVER 25 INDEPENDENT RUNS. *FR* AND *SR* INDICATE THE FEASIBLE RATE AND SUCCESSFUL RATE, RESPECTIVELY. *Ave* AND *Std Dev* INDICATE THE AVERAGE AND STANDARD DEVIATION OF THE BEST FEASIBLE OBJECTIVE FUNCTION VALUES OVER 25 INDEPENDENT RUNS, RESPECTIVELY. *ACT* INDICATES THE AVERAGE CPU TIME OVER 25 INDEPENDENT RUNS. THE WILCOXON'S RANK-SUM TEST AT A 0.05 SIGNIFICANCE LEVEL IS PERFORMED BETWEEN DE/RAND/2/BIN-CAR AND EACH OF DE/RAND/2/BIN-WOCR, DE/RAND/2/BIN-WOC, AND DE/RAND/2/BIN-WOR.

Problem	Status	DE/rand/2/bin-WOCR		DE/rand/2/bin-WOC		DE/rand/2/bin-WOR		DE/rand/2/bin-CaR
F1	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	0.00%		**100.00%**		**100.00%**		**100.00%**
	Ave	17.0000		**13.0000**		**13.0000**		**13.0000**
	Std Dev	0.0000	$+$	0.0000	$\approx$	0.0000	$\approx$	0.0000
	ACT	4.13		4.98		4.38		5.04
F2	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	0.00%		0.00%		**100.00%**		**100.00%**
	Ave	2.0000		2.0000		**1.0000**		**1.0000**
	Std Dev	0.0000	$+$	0.0000	$+$	0.0000	$\approx$	0.0000
	ACT	4.54		6.00		5.09		6.19
F3	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	32.00%		60.00%		**100.00%**		**100.00%**
	Ave	-3.6600		-3.8000		**-4.0000**		**-4.0000**
	Std Dev	0.2380	$+$	0.2500	$+$	0.0000	$\approx$	0.0000
	ACT	4.77		5.61		5.26		5.40
F4	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	Ave	**-6.0000**		**-6.0000**		**-6.0000**		**-6.0000**
	Std Dev	0.0000	$\approx$	0.0000	$\approx$	0.0000	$\approx$	0.0000
	ACT	4.74		5.42		5.23		5.67
F5	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	0.00%		**100.00%**		0.00%		**100.00%**
	Ave	1.2401		**0.2500**		1.2401		**0.2500**
	Std Dev	0.0000	$+$	0.0000	$\approx$	0.0000	$+$	0.0000
	ACT	4.17		5.07		4.62		5.33
F6	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	88.00%		96.00%		**100.00%**		**100.00%**
	Ave	-6657.8460		-6741.6698		**-6783.5818**		**-6783.5818**
	Std Dev	347.5154	$\approx$	209.5596	$\approx$	0.0000	$\approx$	0.0000
	ACT	4.72		5.52		5.19		5.85
F7	*FR*	84.00%		**92.00%**		88.00%		**92.00%**
	SR	**0.00%**		**0.00%**		**0.00%**		**0.00%**
	Ave	*NA*		*NA*		*NA*		*NA*
	Std Dev	*NA*	$\approx$	*NA*	$\approx$	*NA*	$\approx$	*NA*
	ACT	4.59		5.69		5.03		5.80
F8	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	0.00%		0.00%		**100.00%**		**100.00%**
	Ave	7097.8528		7093.8274		**7055.5543**		**7055.5543**
	Std Dev	41.8793	$+$	18.3067	$+$	0.0000	$\approx$	0.0000
	ACT	4.70		5.38		4.94		5.38
F9	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	0.00%		0.00%		32.00%		**48.00%**
	Ave	7567.3332		7481.7776		7132.4432		**7120.1722**
	Std Dev	172.0754	$+$	193.4869	$+$	48.1578	$+$	44.8031
	ACT	4.89		5.74		5.20		5.69
F10	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	0.00%		0.00%		20.00%		**36.00%**
	Ave	8600.0000		8364.0000		7506.6663		**7413.3327**
	Std Dev	474.3416	$+$	494.8737	$+$	190.5165	$+$	233.3341
	ACT	5.08		6.21		5.32		5.93
F11	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	24.00%		32.00%		28.00%		**36.00%**
	Ave	40.9682		37.4760		40.9630		**36.6424**
	Std Dev	7.2714	$+$	6.4018	$\approx$	7.2769	$+$	4.4466
	ACT	10.35		11.51		11.26		11.33
F12	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	28.00%		36.00%		40.00%		**44.00%**
	Ave	74.7723		73.8780		47.5951		**42.1174**
	Std Dev	26.8670	$+$	26.3449	$+$	14.8921	$\approx$	0.6135
	ACT	10.40		11.18		10.95		11.51
F13	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	0.00%		0.00%		0.00%		**20.00%**
	Ave	8966.8064		8946.1031		8930.6730		**8904.9290**
	Std Dev	83.7794	$+$	73.9180	$+$	27.5154	$+$	22.5612
	ACT	4.62		5.27		5.26		5.72
F14	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	8.00%		28.00%		0.00%		**80.00%**
	Ave	8981.1015		8956.6418		8972.9400		**8948.6209**
	Std Dev	74.2260	$+$	9.1689	$+$	55.1837	$+$	2.1379
	ACT	4.73		5.47		5.30		5.90
F15	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	40.00%		40.00%		0.00%		**48.00%**
	Ave	28.9027		28.5329		28.7639		**28.4709**
	Std Dev	0.9408	$+$	0.1798	$\approx$	0.0000	$+$	0.1787
	ACT	4.58		5.69		4.98		5.86
F16	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	88.00%		**100.00%**		**100.00%**		**100.00%**
	Ave	28.7992		**28.4879**		**28.4879**		**28.4879**
	Std Dev	0.8604	$+$	0.0000	$\approx$	0.0000	$\approx$	0.0000
	ACT	4.62		5.74		4.98		6.24
$+/\approx/-$		13/3/0		8/8/0		7/9/0		

TABLE S-IV
RESULTS OF JADE-MIP, JADE-CAR, FROFI-MIP, AND FROFI-CAR OVER 25 INDEPENDENT RUNS. *FR* AND *SR* INDICATE THE FEASIBLE RATE AND SUCCESSFUL RATE, RESPECTIVELY. *Ave* AND *Std Dev* INDICATE THE AVERAGE AND STANDARD DEVIATION OF THE BEST FEASIBLE OBJECTIVE FUNCTION VALUES OVER 25 INDEPENDENT RUNS, RESPECTIVELY. *ACT* INDICATES THE AVERAGE CPU TIME OVER 25 INDEPENDENT RUNS. THE WILCOXON'S RANK-SUM TEST AT A 0.05 SIGNIFICANCE LEVEL IS PERFORMED BETWEEN JADE-MIP AND JADE-CAR, AND FROFI-MIP AND FROFI-CAR.

Problem	Status	JADE-MIP		JADE-CaR	FROFI-MIP		FROFI-CaR
F1	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	0.00%		**100.00%**	0.00%		**100.00%**
	Ave	17.0000		**13.0000**	17.0000		**13.0000**
	Std Dev	0.0000	+	0.0000	0.0000	+	0.0000
	ACT	7.86		8.08	5.11		5.49
F2	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	0.00%		**100.00%**	**100.00%**		**100.00%**
	Ave	2.0000		**1.0000**	**1.0000**		**1.0000**
	Std Dev	0.0000	+	0.0000	0.0000	≈	0.0000
	ACT	8.43		9.80	4.77		5.14
F3	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	0.00%		**100.00%**	96.00%		**100.00%**
	Ave	-3.5000		**-4.0000**	-3.9800		**-4.0000**
	Std Dev	0.0000	+	0.0000	0.1000	≈	0.0000
	ACT	8.75		8.96	5.24		5.62
F4	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	Ave	**-6.0000**		**-6.0000**	**-6.0000**		**-6.0000**
	Std Dev	0.0000	≈	0.0000	0.0000	≈	0.0000
	ACT	8.78		9.07	5.21		5.58
F5	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	0.00%		**100.00%**	48.00%		**100.00%**
	Ave	1.2401		**0.2500**	0.7648		**0.2500**
	Std Dev	0.0000	+	0.0000	0.5049	+	0.0000
	ACT	7.90		8.64	4.58		4.96
F6	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	Ave	**-6783.5818**		**-6783.5818**	**-6783.5818**		**-6783.5818**
	Std Dev	0.0000	≈	0.0000	0.0000	≈	0.0000
	ACT	8.54		9.26	5.28		5.64
F7	*FR*	**100.00%**		**100.00%**	92.00%		**100.00%**
	SR	**0.00%**		**0.00%**	84.00%		**100.00%**
	Ave	1.0000		**0.9954**	*NA*		**0.2114**
	Std Dev	0.0000	+	0.0158	*NA*	+	0.0000
	ACT	7.72		8.22	5.35		5.70
F8	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	**0.00%**		**0.00%**	56.00%		**100.00%**
	Ave	7187.4492		**7159.7941**	7073.9490		**7055.5543**
	Std Dev	104.4648	≈	80.1203	24.0099	+	0.0000
	ACT	7.11		7.73	5.35		5.71
F9	FR	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	**0.00%**		**0.00%**	24.00%		**100.00%**
	Ave	7217.8317		**7214.2284**	7273.3326		**7083.3317**
	Std Dev	136.7341	≈	128.0025	211.4586	+	0.0000
	ACT	7.21		8.01	5.53		5.89
F10	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	4.00%		**12.00%**	28.00%		**100.00%**
	Ave	7487.3189		**7402.7927**	7541.3329		**7133.3317**
	Std Dev	450.2467	≈	272.9971	259.6800	+	0.0000
	ACT	7.84		8.20	5.48		5.86
F11	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	4.00%		**72.00%**	36.00%		**92.00%**
	Ave	37.7755		**33.6079**	33.5850		**33.5171**
	Std Dev	2.6320	+	0.2254	0.0653	+	0.0362
	ACT	13.43		14.87	10.70		11.07
F12	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	0.00%		**16.00%**	60.00%		**72.00%**
	Ave	61.7478		**41.9968**	44.0349		**41.8516**
	Std Dev	26.4311	+	0.1817	10.8712	≈	0.1662
	ACT	13.99		14.75	11.07		11.41
F13	*FR*	60.00%		**68.00%**	**100.00%**		**100.00%**
	SR	0.00%		**0.00%**	24.00%		**68.00%**
	Ave	*NA*		*NA*	8893.4324		**8886.8089**
	Std Dev	*NA*	≈	*NA*	13.3607	+	4.0494
	ACT	8.07		7.96	6.41		6.78
F14	FR	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	8.00%		**12.00%**	20.00%		**96.00%**
	Ave	9053.1344		**8985.8192**	8955.0741		**8947.7481**
	Std Dev	125.3470	+	75.4249	7.5813	+	0.9561
	ACT	7.44		7.64	6.46		6.86
F15	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	0.00%		**12.00%**	28.00%		**100.00%**
	Ave	29.5538		**28.6939**	28.9173		**28.3514**
	Std Dev	1.6882	+	0.5253	1.3815	+	0.0000
	ACT	7.49		8.07	6.07		6.40
F16	*FR*	**100.00%**		**100.00%**	**100.00%**		**100.00%**
	SR	0.00%		**8.00%**	68.00%		**100.00%**
	Ave	30.2381		**28.8030**	29.3181		**28.4879**
	Std Dev	1.1629	+	0.8591	1.2351	+	0.0000
	ACT	8.02		8.57	6.16		6.54
+/≈/−		10/6/0			11/5/0		

TABLE S-V

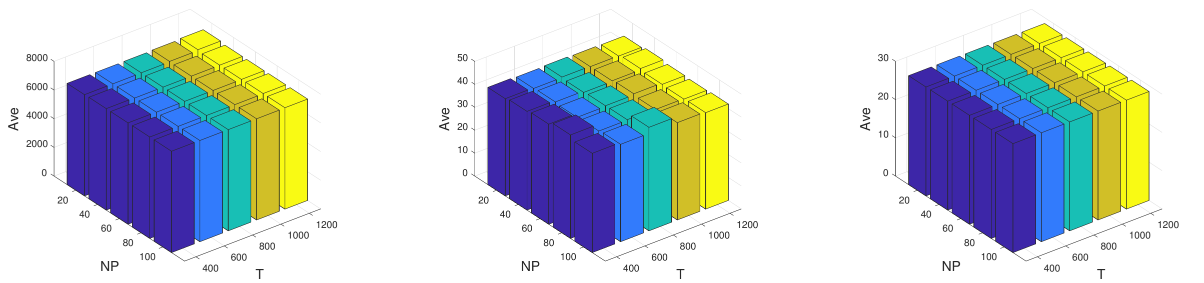
RESULTS OF MDE-LS, MDE-IHS, EMDE, AND FROFI-CAR OVER 25 INDEPENDENT RUNS. *FR* AND *SR* INDICATE THE FEASIBLE RATE AND SUCCESSFUL RATE, RESPECTIVELY. *Ave* AND *Std Dev* INDICATE THE AVERAGE AND STANDARD DEVIATION OF THE BEST FEASIBLE OBJECTIVE FUNCTION VALUES OVER 25 INDEPENDENT RUNS, RESPECTIVELY. *ACT* INDICATES THE AVERAGE CPU TIME OVER 25 INDEPENDENT RUNS. THE WILCOXON'S RANK-SUM TEST AT A 0.05 SIGNIFICANCE LEVEL IS PERFORMED BETWEEN FROFI-CAR AND EACH OF THE THREE COMPETITORS.

Problem	Status	MDE-LS		MDE-IHS		EMDE		FROFI-CaR
F1	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	0.00%		0.00%		0.00%		**100.00%**
	Ave	17.0000		17.0000		17.0000		**13.0000**
	Std Dev	0.0000	$+$	0.0000	$+$	0.0000	$+$	0.0000
	ACT	2.01		2.06		1.71		5.49
F2	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	**100.00%**		**100.00%**		0.00%		**100.00%**
	Ave	**1.0000**		**1.0000**		2.0000		**1.0000**
	Std Dev	0.0000	$\approx$	0.0000	$\approx$	0.0000	$+$	0.0000
	ACT	2.44		2.46		1.97		5.14
F3	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	12.00%		24.00%		4.00%		**100.00%**
	Ave	-3.5600		-3.7998		-3.5200		**-4.0000**
	Std Dev	0.1658	$+$	0.2499	$+$	0.1000	$+$	0.0000
	ACT	2.55		2.61		2.40		5.62
F4	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	Ave	**-6.0000**		**-6.0000**		**-6.0000**		**-6.0000**
	Std Dev	0.0000	$\approx$	0.0000	$\approx$	0.0000	$\approx$	0.0000
	ACT	2.51		2.51		2.40		5.58
F5	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	8.00%		16.00%		0.00%		**100.00%**
	Ave	1.1609		1.0817		1.2500		**0.2500**
	Std Dev	0.2742	$+$	0.3705	$+$	0.0000	$+$	0.0000
	ACT	2.00		1.84		1.76		4.96
F6	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	92.00%		64.00%		**100.00%**		**100.00%**
	Ave	-6699.7579		-6657.2603		**-6783.5818**		**-6783.5818**
	Std Dev	290.1222	$\approx$	347.2974	$+$	0.0000	$\approx$	0.0000
	ACT	2.66		2.52		2.57		5.64
F7	*FR*	12.00%		96.00%		76.00%		**100.00%**
	SR	0.00%		0.00%		0.00%		**100.00%**
	Ave	*NA*		*NA*		*NA*		**0.2114**
	Std Dev	*NA*	$+$	*NA*	$+$	*NA*	$+$	0.0000
	ACT	3.24		3.25		2.91		5.70
F8	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	8.00%		0.00%		0.00%		**100.00%**
	Ave	7126.4658		7545.4781		7567.5140		**7055.5543**
	Std Dev	44.2469	$+$	497.3655	$+$	607.1346	$+$	0.0000
	ACT	3.31		3.33		3.05		5.71
F9	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	0.00%		0.00%		4.00%		**100.00%**
	Ave	7346.6662		7732.1012		7695.5554		**7083.3317**
	Std Dev	211.1115	$+$	404.3210	$+$	307.3518	$+$	0.0000
	ACT	3.45		3.49		3.06		5.89
F10	*FR*	**100.00%**		**100.00%**		96.00%		**100.00%**
	SR	16.00%		0.00%		8.00%		**100.00%**
	Ave	7645.3331		7852.3596		*NA*		**7133.3317**
	Std Dev	290.0962	$+$	508.5911	$+$	*NA*	$+$	0.0000
	ACT	3.42		3.45		3.05		5.86
F11	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	0.00%		0.00%		12.00%		**92.00%**
	Ave	50.6751		106.2005		42.9037		**33.5171**
	Std Dev	8.1057	$+$	55.7755	$+$	8.1029	$+$	0.0362
	ACT	13.97		13.72		14.14		11.07
F12	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	0.00%		0.00%		0.00%		**72.00%**
	Ave	101.5348		131.4884		111.7425		**41.8516**
	Std Dev	3.8791	$+$	52.5734	$+$	69.8873	$+$	0.1662
	ACT	13.82		13.75		13.82		11.41
F13	*FR*	88.00%		72.00%		12.00%		**100.00%**
	SR	0.00%		4.00%		0.00%		**68.00%**
	Ave	*NA*		*NA*		*NA*		**8886.8089**
	Std Dev	*NA*	$+$	*NA*	$+$	*NA*	$+$	4.0494
	ACT	3.29		3.23		3.02		6.78
F14	*FR*	**100.00%**		92.00%		0.00%		**100.00%**
	SR	16.00%		16.00%		0.00%		**96.00%**
	Ave	8980.6825		*NA*		*NA*		**8947.7481**
	Std Dev	74.4063	$+$	*NA*	$+$	*NA*	$+$	0.9561
	ACT	3.31		3.27		3.08		6.86
F15	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	0.00%		0.00%		0.00%		**100.00%**
	Ave	31.7988		36.2163		32.7862		**28.3514**
	Std Dev	2.7397	$+$	6.3672	$+$	2.7529	$+$	0.0000
	ACT	3.52		3.54		3.13		6.40
F16	*FR*	**100.00%**		**100.00%**		**100.00%**		**100.00%**
	SR	8.00%		0.00%		4.00%		**100.00%**
	Ave	32.5555		40.9216		35.1298		**28.4879**
	Std Dev	4.0209	$+$	7.3693	$+$	5.7166	$+$	0.0000
	ACT	3.57		3.50		3.14		6.54
$+/\approx/-$		13/3/0		14/2/0		14/2/0		

S-III. Additional Results and Discussions

A. Study of The Parameter Settings in CaR

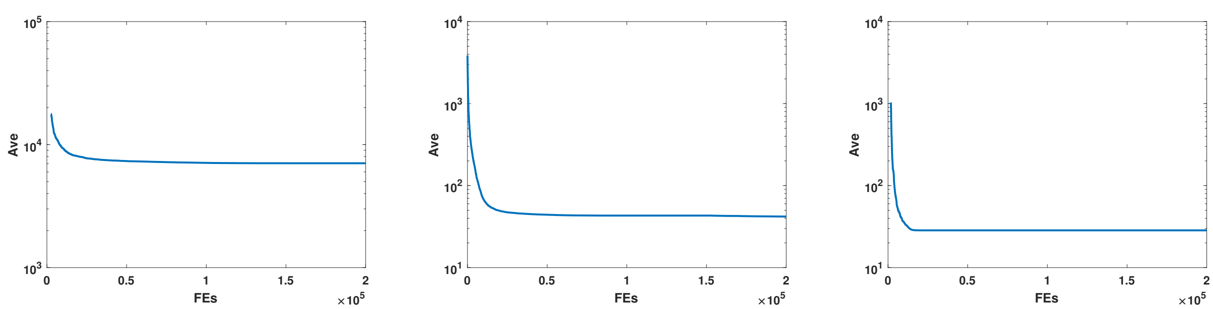
We firstly discussed the sensitivity of two parameters (i.e., NP and T) introduced in Section IV-A. To this end, DE/rand/2/bin-CaR was selected as the instance algorithm, and its performance was tested on three test problems: F8, F12, and F16. NP and T were selected from the following two sets: {20, 40, 60, 80, 100} and {400, 600, 800, 1000, 1200}, respectively. Fig. S-1 records the *Ave* values provided by the 25 different combinations of NP and T. It can be observed from Fig. S-1 that the influences of these two parameters are not obvious.



(a) (b) (c)

Fig. S-1. *Ave* values provided by DE/rand/2/bin-CaR with 25 different combinations of NP and T on F8, F12, and F16. (a) F8 (b) F12 (c) F16

Subsequently, we used DE/rand/2/bin-CaR to solve F8, F12, and F16, with the aim of investigating the settings of $MaxFEs$. The convergence curves are plotted in Fig. S-2, in which the horizontal axis represents the number of FEs and the vertical axis represents the *Ave* value. From Fig. S-2, we can observe that CaR can converge within 5.0E+04 FEs for these three problems. However, in this paper, we still set $MaxFEs$ to 2.0E+05 to ensure that our algorithm can converge completely when facing with an unknown optimization problem.



(a) (b) (c)

Fig. S-2. Evolution of the *Ave* value on F8, F12, and F16. (a) F8 (b) F12 (c) F16

B. Comparison with The Commercial Solver

We also compared FROFI-CaR with a commercial solver, i.e., branch-and-deduce optimization navigator (BARON). The results derived from BARON and FROFI-CaR are recorded in Table S-VI. Since both of them can provide 100% *FR*, we do not exhibit *FR* in Table S-VI. From Table S-VI, FROFI-CaR provides similar *Ave* values as BARON on F1-F10, F15, and F16. Moreover, BARON takes less *ACT* than FROFI-CaR to find the optimal solutions of these test problems. In addition, BARON provides better *Ave* and *ACT* values than FROFI-CaR on F11 and F12. The superiority of BARON on F1-F12, F15, and F16 is not difficult to understand since it contains several deterministic mathematical programming algorithms and is good at solving optimization problems with simple nonlinearity. When solving these test problems, BARON usually adopts the gradient information, thus greatly enhancing the efficiency. However, BARON cannot solve F13 and F14 in any run. Note that compared with F1-F12, F15, and F16, F13 and F14 are with high nonlinearity (e.g., trigonometric function terms $\sin(x)$ and $\cos(x)$), which signifies that the performance of BARON may be limited when solving highly nonlinear MIP problems. In

TABLE S-VI
RESULTS OF BARON AND FROFI-CAR OVER 25 INDEPENDENT RUNS. *SR* INDICATE THE SUCCESSFUL RATE. *Ave* AND *Std Dev* INDICATE THE AVERAGE AND STANDARD DEVIATION OF THE BEST FEASIBLE OBJECTIVE FUNCTION VALUES OVER 25 INDEPENDENT RUNS, RESPECTIVELY. *ACT* INDICATES THE AVERAGE CPU TIME OVER 25 INDEPENDENT RUNS.

Problem	Status	BARON	FROFI-CaR	Problem	Status	BARON	FROFI-CaR
F1	*SR*	**100.00%**	**100.00%**	F9	*SR*	**100.00%**	**100.00%**
	Ave	**13.0000**	**13.0000**		*Ave*	**7083.3317**	**7083.3317**
	Std Dev	0.0000	0.0000		*Std Dev*	0.0000	0.0000
	ACT	0.09	5.04		*ACT*	0.34	5.69
F2	*SR*	**100.00%**	**100.00%**	F10	*SR*	**100.00%**	**100.00%**
	Ave	**1.0000**	**1.0000**		*Ave*	**7133.3317**	**7133.3317**
	Std Dev	0.0000	0.0000		*Std Dev*	0.0000	0.0000
	ACT	0.08	6.19		*ACT*	0.14	5.93
F3	*SR*	**100.00%**	**100.00%**	F11	*SR*	**100.00%**	92.00%
	Ave	**-4.0000**	**-4.0000**		*Ave*	**33.5066**	33.5171
	Std Dev	0.0000	0.0000		*Std Dev*	0.0000	0.0362
	ACT	0.03	5.40		*ACT*	0.61	11.33
F4	*SR*	**100.00%**	**100.00%**	F12	*SR*	**100.00%**	72.00%
	Ave	**-6.0000**	**-6.0000**		*Ave*	**41.7399**	41.8516
	Std Dev	0.0000	0.0000		*Std Dev*	0.0000	0.1662
	ACT	0.03	5.67		*ACT*	0.62	11.51
F5	*SR*	**100.00%**	**100.00%**	F13	*SR*	*NA*	**68%**
	Ave	**0.2500**	**0.2500**		*Ave*	*NA*	**8886.8089**
	Std Dev	0.0000	0.0000		*Std Dev*	*NA*	4.0494
	ACT	0.19	5.33		*ACT*	*NA*	5.72
F6	*SR*	**100.00%**	**100.00%**	F14	*SR*	*NA*	**96%**
	Ave	**-6783.5818**	**-6783.5818**		*Ave*	*NA*	**8947.7481**
	Std Dev	0.0000	0.0000		*Std Dev*	*NA*	0.9561
	ACT	1.33	5.85		*ACT*	*NA*	5.90
F7	*SR*	**100.00%**	**100.00%**	F15	*SR*	**100%**	**100%**
	Ave	**0.2114**	**0.2114**		*Ave*	**28.3514**	**28.3514**
	Std Dev	0.0000	0.0000		*Std Dev*	0.0000	0.0000
	ACT	0.20	5.80		*ACT*	0.23	5.86
F8	*SR*	**100.00%**	**100.00%**	F16	*SR*	**100%**	**100%**
	Ave	**7055.5543**	**7055.5543**		*Ave*	**28.4879**	**28.4879**
	Std Dev	0.0000	0.0000		*Std Dev*	0.0000	0.0000
	ACT	0.50	5.38		*ACT*	0.23	6.24

contrast, FROFI-CaR can achieve 68% *SR* and 96% *SR* on F13 and F14, respectively. Therefore, the advantage of FROFI-CaR against BARON is its capability to deal with MIP problems with high nonlinearity.

On the other hand, BARON needs some mathematical properties of test problems, such as the gradient information; therefore, it cannot solve black-box optimization problems. However, FROFI-CaR does not depend on any mathematical property of test problems; thus, another advantage of FROFI-CaR is that it has the potential to handle black-box optimization problems.