Composite Differential Evolution for Constrained Evolutionary Optimization

Bing-Chuan Wang, Han-Xiong Li, *Fellow, IEEE*, Jia-Peng Li, and Yong Wang, *Senior Member, IEEE*

***Abstract*—When solving constrained optimization problems by evolutionary algorithms, the search algorithm plays a crucial role. In general, we expect that the search algorithm has the capability to balance not only diversity and convergence but also constraints and objective function during the evolution. For this purpose, this paper proposes a composite differential evolution for constrained optimization, which includes three different trial vector generation strategies with distinct advantages. In order to strike a balance between diversity and convergence, one of these three trial vector generation strategies is able to increase diversity, and the other two exhibit the property of convergence. In addition, to accomplish the tradeoff between constraints and objective function, one of the two trial vector generation strategies for convergence is guided by the individual with the least degree of constraint violation in the population, and the other is guided by the individual with the best objective function value in the population. After producing offspring by the proposed composite differential evolution, the feasibility rule and the ε constrained method are combined elaborately for selection in this paper. Moreover, a restart scheme is proposed to help the population jump out of a local optimum in the infeasible region for some extremely complicated constrained optimization problems. By assembling the above techniques together, a constrained composite differential evolution is proposed. The experiments on two sets of benchmark test functions with various features, i.e., 24 test functions from IEEE CEC2006 and 18 test functions with 10 dimensions and 30 dimensions from IEEE CEC2010, have demonstrated that the proposed method shows better or at least competitive performance against other state-of-the-art methods.**

***Index Terms*—constrained optimization, evolutionary algorithm, composite differential evolution, constraint-handling technique**

I. Introduction

This work was supported in part by a General Research Fund (GRF) project from Research Grant Council (RGC) of Hong Kong SAR (CityU: 11205615), in part by the Project from the City University of Hong Kong under Grant 7004666, in part by the Innovation-Driven Plan in Central South University under Grant 2018CX010, in part by the National Natural Science Foundation of China under Grant 61673397, and in part by the Hunan Provincial Natural Science Fund for Distinguished Young Scholars (Grant No. 2016JJ1018). (*Corresponding author: Yong Wang*).

B.-C. Wang is with the Department of Systems Engineering and Engineering Management, City University of Hong Kong, Hong Kong. (e-mail: bingcwang3-c@my.cityu.edu.hk).

H.-X. Li is with the Department of Systems Engineering and Engineering Management, City University of Hong Kong, Hong Kong, and also with the State Key Laboratory of High Performance Complex Manufacturing, Central South University, Changsha 410083, China. (e-mail: mehxli@cityu.edu.hk).

J.-P. Li is with the School of Information Science and Engineering, Central South University, Changsha 410083, China. (e-mail: ljpcsu@csu.edu.cn).

Y. Wang is with the School of Information Science and Engineering, Central South University, Changsha 410083, China, and also with the School of Computer Science and Electronic Engineering, University of Essex, Colchester CO4 3SQ, UK. (Email: ywang@csu.edu.cn).

CONSTRAINTS are everywhere. Many practical optimization problems, such as vehicle configuration design [1], scheduling [2], [3], digital circuit structure design [4], mixed-model two-sided assembly line [5], and antenna design [6], can be formulated as constrained optimization problems (COPs). Hence, how to solve COPs is of great practical significance.

As a kind of population-based heuristic optimization algorithms, evolutionary algorithms (EAs) [7] have attracted increasing interest in solving COPs. As a result, a variety of constrained EAs has been proposed [8], [9], [10]. A constrained EA includes two main components: 1) search algorithm and 2) constraint-handling technique. Search algorithm plays the role of generating new candidate solutions, and thus has a significant impact on the performance of constrained EAs. During the past two decades, differential evolution (DE) [11] has become one of the most popular EA paradigms. DE has numerous attractive advantages. First of all, its structure is simple and it can be implemented easily in any programming language. In addition, it includes few control parameters. Moreover, it has already achieved top ranks in a lot of competitions at IEEE Congress on Evolutionary Computation (IEEE CEC). Note that no other single algorithm can accomplish this [12]. More importantly, its search ability has been demonstrated in many real-world applications [13], [14], [15].

Due to the above advantages, DE has been frequently applied to solve COPs. Two primary ways of utilizing DE for constrained optimization can be summarized as: 1) designing a new DE, or 2) extending an existing DE originally designed for global optimization to deal with constrained optimization. In terms of case 2), many DE variants for global optimization have been tailored to tackle COPs [16], [17], [18], [19]. As an outstanding global optimizer, composite differential evolution (CoDE) [20] exhibits a few strengths, including ease of implementation, powerful search ability, integrating the strengths of different trial vector generation strategies, etc. However, few current studies investigate CoDE for constrained optimization.

Motivated by the above consideration, this paper seeks to make use of the idea of CoDE to solve COPs. The underlying idea behind CoDE is the utilization of three different trial vector generation strategies of DE with a variety of characteristics to address the key issue of global optimization, i.e., the tradeoff between diversity and convergence. In order to extend CoDE to tackle COPs, the tradeoff between constraints and objective function should also be taken into account. To this end, this paper proposes a constrained composite differential evolution, called C^{2}oDE, to address these two issues.

Similar to CoDE, C^{2}oDE also contains three different trial vector generation strategies with distinct advantages. Specifically, one trial vector generation strategy for diversity and two trial vector generation strategies for convergence are employed to balance diversity and convergence. In addition, one of the two trial vector generation strategies for convergence is guided by the individual with the least degree of constraint violation while the other is guided by the individual with the best objective function value, with the aim of balancing constraints and objective function. During the evolution, these three trial vector generation strategies are used to generate three offspring for each target vector. Afterward, a new comparison rule, which combines the feasibility rule with the ε constrained method, is proposed. Herein, the feasibility rule is applied to preselect the best one from the three offspring as the trial vector. Due to the fact that the feasibility rule prefers constraints, the ε constrained method, which can incorporate the information of objective function to a certain degree, is used to compare each target vector with its trial vector. Therefore, the new comparison rule can further promote the balance between constraints and objective function. Moreover, a restart scheme is designed to help the population jump out of a local optimum in the infeasible region for some extremely complex COPs.

By combining the strengthes of the above-mentioned techniques, C^{2}oDE achieves a reasonable tradeoff between diversity and convergence as well as between constraints and objective function. The contributions of this paper are summarized as follows:

- The principle of CoDE is successfully applied to design a search algorithm for constrained optimization.
- The feasibility rule and the ε constrained method are integrated in an effective way to select promising individuals for the next population.
- A restart scheme is proposed to cope with COPs with extremely complicated constraints.
- Systematic experiments have demonstrated that C^{2}oDE provides state-of-the-art performance on two benchmark test suites.

The rest of this paper is organized as follows. Section II introduces the preliminary knowledge. The related work on constrained DE is reviewed in Section III. Section IV illustrates the proposed method in detail. Extensive experiments and discussions are carried out in Section V. Section VI concludes this paper.

II. Preliminary Knowledge

A. Constrained Optimization Problems (COPs)

Without loss of generality, a COP [21], [22] can be described as follows:

$$\begin{aligned} &\text{minimize} && f(\vec{x}),\ \vec{x}=(x_1,...,x_D)\in S,\ L_i\le x_i\le U_i \\ &\text{subject to}: && g_j(\vec{x})\le 0,\ j=1,...,l \\ &&& h_j(\vec{x})=0,\ j=l+1,...,m \end{aligned}$$

where $f(\vec{x})$ is the objective function, $\vec{x}$ is the decision vector, x_i is the ith dimension of $\vec{x}$, L_i and U_i are the upper and lower bounds of x_i, respectively, D is the number of dimensions, $S=\prod_{i=1}^{D}[L_i,U_i]$ represents the decision space, $g_j(\vec{x})$ is the jth inequality constraint, l is the number of inequality constraints, $h_j(\vec{x})$ is the $(j-l)$th equality constraint, and $(m-l)$ is the number of equality constraints.

For COPs, the degree of constraint violation of the decision vector $\vec{x}$ can be expressed as follows:

$$G(\vec{x})=\sum_{j=1}^{m}G_j(\vec{x}) \tag{1}$$

where $G_j(\vec{x})$ is the degree of constraint violation on the jth constraint and calculated as follows:

$$G_j(\vec{x})=\begin{cases}\max(0,g_j(\vec{x})), & 1\le j\le l\\ \max(0,|h_j(\vec{x})|-\delta), & l+1\le j\le m\end{cases} \tag{2}$$

In Equation (2), δ is a positive tolerance value to relax equality constraints to a certain extent. $\vec{x}$ is called a feasible solution if $G(\vec{x})=0$. The aim of solving COPs is to locate the optimum in the feasible region.

B. Differential Evolution (DE)

The unique feature of DE is to make use of differential vectors to generate offspring [12], [23]. In general, DE consists of four stages, i.e., initialization, mutation, crossover, and selection.

Firstly, an initial population including NP target vectors (also called NP individuals) is randomly generated from the decision space. In the mutation stage, a mutation operator is implemented to generate a mutant vector for each target vector $\vec{x}_i^t$ ($i\in\{1,\ldots,NP\}$) at generation t. Several mutation operators have been proposed. As a representative, DE/rand/1 is described as follows:

$$\vec{v}_i^t=\vec{x}_{r_1}^t+F\cdot(\vec{x}_{r_2}^t-\vec{x}_{r_3}^t) \tag{3}$$

where $\vec{v}_i^t$ is the mutant vector of the ith target vector $\vec{x}_i^t$, $\vec{x}_{r_1}^t$, $\vec{x}_{r_2}^t$, and $\vec{x}_{r_3}^t$ are three mutually distinct target vectors randomly selected from the population, and F is the scaling factor. Some other popular mutation operators are enumerated as follows:

- DE/rand/2

$$\vec{v}_i^t=\vec{x}_{r_1}^t+F\cdot(\vec{x}_{r_2}^t-\vec{x}_{r_3}^t)+F\cdot(\vec{x}_{r_4}^t-\vec{x}_{r_5}^t) \tag{4}$$

- DE/rand-to-best/1

$$\vec{v}_i^t=\vec{x}_{r_1}^t+F\cdot(\vec{x}_{best}^t-\vec{x}_{r_1}^t)+F\cdot(\vec{x}_{r_2}^t-\vec{x}_{r_3}^t) \tag{5}$$

- DE/current-to-best/1

$$\vec{v}_i^t=\vec{x}_i^t+F\cdot(\vec{x}_{best}^t-\vec{x}_i^t)+F\cdot(\vec{x}_{r_1}^t-\vec{x}_{r_2}^t) \tag{6}$$

- DE/current-to-rand/1

$$\vec{v}_i^t=\vec{x}_i^t+rand\cdot(\vec{x}_{r_1}^t-\vec{x}_i^t)+F\cdot(\vec{x}_{r_2}^t-\vec{x}_{r_3}^t) \tag{7}$$

where $\vec{x}_{r_1}^t$, $\vec{x}_{r_2}^t$, $\vec{x}_{r_3}^t$, $\vec{x}_{r_4}^t$, and $\vec{x}_{r_5}^t$ are five mutually distinct target vectors randomly selected from the population, $\vec{x}_{best}^t$ is the best target vector in the current population, and $rand$ is a uniformly distributed random number between 0 and 1.

Different mutation operators have distinct characteristics. DE/rand/1 is the most commonly used mutation operator in the

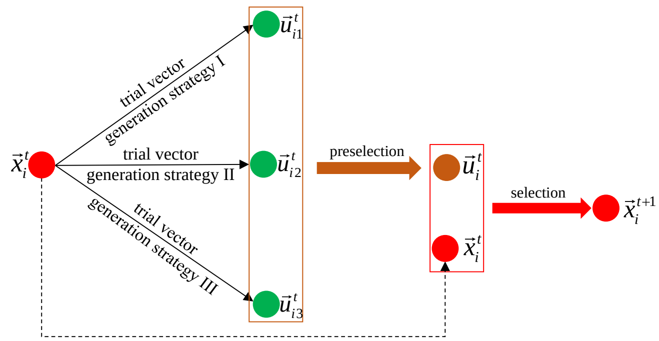


Fig. 1. Framework of CoDE.

DE community [20], in which all individuals are selected in a random manner for mutation. Due to the fact that an additional differential vector is utilized, DE/rand/2/ can provide a better perturbation than DE/rand/1. By making use of the information of the best individual, both DE/rand-to-best/1 and DE/current-to-best/1 can speed up the convergence. In DE/current-to-rand/1, each target vector learns from a randomly selected individual, thus promoting the diversity.

In the crossover stage, a crossover operator is conducted on each pair of $\vec{x}_i^t$ and $\vec{v}_i^t$ to produce a trial vector $\vec{u}_i^t$. The frequently used binomial crossover is introduced below:

$$u_{i,j}^t = \begin{cases} v_{i,j}^t, & \text{if } rand_j < CR \text{ or } j = j_{rand} \\ x_{i,j}^t, & \text{otherwise} \end{cases} \tag{8}$$

where $u_{i,j}^t$, $x_{i,j}^t$, and $v_{i,j}^t$ are the *j*th dimension of $\vec{u}_i^t$, $\vec{x}_i^t$, and $\vec{v}_i^t$, respectively, $rand_j$ is a random number uniformly generated between 0 and 1, CR is the crossover control parameter, and j_{rand} is a random integer uniformly generated between 1 and *D*.

Finally, a selection operator is performed on $\vec{x}_i^t$ and $\vec{u}_i^t$, and the better one is selected as the target vector of the $(t+1)th$ generation.

$$\vec{x}_i^{t+1} = \begin{cases} \vec{u}_i^t, & \text{if } f(\vec{u}_i^t) \le f(\vec{x}_i^t) \\ \vec{x}_i^t, & \text{otherwise} \end{cases} \tag{9}$$

In DE, a combination of a mutation operator and a crossover operator is called a trial vector generation strategy.

Currently, DE has been successfully applied to solve optimization problems in a considerable number of fields, such as electrical and power systems [24], [25], manufacturing science and operational research [26], [27], automotive design [28], and controller design [13], [14], [15].

C. CoDE

CoDE is one of the top DE variants proposed by Wang *et al.* [20] for global optimization. The main idea of CoDE is to combine several effective trial vector generation strategies with several appropriate DE parameter settings, which show complementary characteristics, to improve DE's performance.

In CoDE, a strategy pool comprised of three well-studied trial vector generation strategies, i.e., DE/rand/1/bin, DE/rand/2/bin, and DE/current-to-rand/1, is constructed in advance. On the other hand, a parameter pool involving three pairs of F and CR is constructed beforehand: [*F*=0.8; *CR*=0.2], [*F*=1.0; *CR*=0.1], and [*F*=1.0; *CR*=0.9]. As depicted in Fig. 1, three offspring, i.e., $\vec{u}_{i_1}^t$, $\vec{u}_{i_2}^t$, and $\vec{u}_{i_3}^t$, are generated for each target vector $\vec{x}_i^t$ via implementing the three trial vector generation strategies in the strategy pool one by one. Moreover, each trial vector generation strategy is associated with a pair of F and CR randomly chosen from the parameter pool. Subsequently, the best one among the three offspring is preselected as the trial vector $\vec{u}_i^t$. Finally, the better one between $\vec{x}_i^t$ and $\vec{u}_i^t$ is selected as the potential individual for the next generation.

By utilizing distinct advantages of different trial vector generation strategies and parameter settings, CoDE accomplishes outstanding performance. Owing to its simple structure, ease of implementation, and effectiveness, CoDE is fully investigated for constrained optimization in this paper.

D. Feasibility Rule

The feasibility rule proposed by Deb [29] is a well-known constraint-handling technique. It compares pairwise individuals as follows:

1) Between two infeasible individuals, the one with less degree of constraint violation is preferred.
2) If one individual is feasible and the other is infeasible, the feasible one is preferred.
3) Between two feasible individuals, the one with a smaller objective function value is preferred.

E. ε Constrained Method

The ε constrained method proposed by Takahama and Sakai [30], [31] is another representative constraint-handling technique. When comparing two individuals, say $\vec{x}_i^t$ and $\vec{x}_j^t$, $\vec{x}_i^t$ is better than $\vec{x}_j^t$ if and only if the following conditions are satisfied:

$$\begin{cases} f(\vec{x}_i) < f(\vec{x}_j), & \text{if } G(\vec{x}_i) \le \varepsilon \wedge G(\vec{x}_j) \le \varepsilon \\ f(\vec{x}_i) < f(\vec{x}_j), & \text{if } G(\vec{x}_i) = G(\vec{x}_j) \\ G(\vec{x}_i) < G(\vec{x}_j), & \text{otherwise} \end{cases} \tag{10}$$

In Equation (10), ε declines as the generation increases:

$$\varepsilon = \begin{cases} \varepsilon_0 (1 - \frac{t}{T})^{cp}, & \text{if } \frac{t}{T} \le p \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

$$cp = -\frac{\log \varepsilon_0 + \lambda}{\log(1-p)} \tag{12}$$

where ε_0 is the initial threshold and set to be the maximum degree of constraint violation of the initial population, T is the maximum generation number, λ is set to 6 in this paper, and p controls the degree that the information of objective function is exploited.

III. Related Work

DE has become a very popular search engine for constrained optimization and this paper focuses mainly on constrained DE (CDE). In this section, we survey the development of CDE primarily during the last five years and classify CDE into three classes: 1) single-strategy CDE, 2) multi-strategy CDE, and 3) CDE coupled with other operators. For a more comprehensive review, the interested reader can refer to [32].

A. Single-Strategy CDE

As suggested by the name, single-strategy CDE signifies that CDE just includes one trial vector generation strategy.

For example, De Melo and Carosio [33] conducted an empirical analysis on five classical trial vector generation strategies which are separatively integrated with a simple penalty function. According to the experimental results, they claimed that classical DE with a simple penalty function is still very competitive.

In [19], the famous global optimizer JADE [34] is combined with an archiving-based adaptive tradeoff model [35] for constrained optimization.

Gao *et al.* [36] suggested a dual population scheme in which one population is responsible for tackling constraints and the other for optimizing objective function. Moreover, a modified DE/rand/1/bin is designed to share the information between two populations.

Takahama and Sakaia [37] presented an efficient CDE. Through utilizing kernel regression, this method has the capability to find approximately optimal solutions with a very small number of function evaluations. In addition, the ε constrained method serves as the constraint-handling technique and DE/rand/1 with exponential crossover operator serves as the search algorithm. Yi *et al.* [38] presented an ε constrained DE with pre-estimated comparison based on gradient-based approximation for solving COPs.

Wang and Cai [39] proposed a dynamic hybrid framework referred as DyHF for constrained optimization. In DyHF, the global and local search models are dynamically implemented according to the feasibility proportion of the current population. In the same year, Wang and Cai [40] combined multiobjective optimization with DE and proposed CMODE. In CMODE, an infeasible solution replacement mechanism based on multiobjective optimization is devised to guide the population toward promising solutions and the feasible region simultaneously. Note that both DyHF and CMODE exploit Pareto dominance [41] to compare individuals.

Hamza *et al.* [42] integrated a DE with multi-constraint consensus. In this method, the constraint consensus [43] aims at moving the infeasible solutions along the parallel direction to the violated constraint, thus making them feasible quickly. The constraint consensus has also been used in [44].

In the self-adaptive interior penalty based DE [45], the scaling factor F and crossover control parameter CR of DE/rand/1/bin are adjusted according to the success rate. Fan and Yan [46] also developed a self-adaptive penalty based DE. However, the two DE control parameters, i.e., F and CR, together with the penalty factor, are adapted in the manner of coevolution by the alopex algorithm [47]. In [48], a fuzzy rule based penalty function approach is designed. Li and Zhang [49] showed that a modified penalty method, called minimum penalty method, is effective to handle constraints.

It is necessary to emphasize that for [38], [39], [40], [42], [45], [46], [48], and [49], DE/rand/1/bin is directly employed as the search algorithm.

B. Multi-Strategy CDE

In contrast to the first class, a number of CDE involves multiple trial vector generation strategies as pinpointed by the name.

For instance, Dong *et al.* [17] combined CoDE [20] with oracle penalty function to solve COPs. Herein, CoDE is treated as the search algorithm in a straightforward way.

Long *et al.* [50] integrated three trial vector generation strategies, i.e., DE/rand/1/bin, DE/best/1/bin, and DE/current-to-rand/1 to evolve the population. In this method, the initial population is divided into three sub-populations with equal size, and then each sub-population is assigned with a trial vector generation strategy to update the individuals.

De Melo and Carosio [51] provided a systematic way to ensemble five trial vector generation strategies, in which each trial vector generation strategy is applied to generate a corresponding solution and winner-take-all paradigm is utilized to select the best one as the trial vector.

By taking advantage of the concept of multi-population evolution, a cultural DE is developed in [52], in which each population is managed by its private cultural DE.

In [53], DE/rand/1/bin is employed in the early stage for exploration while DE/rand/1 with exponential crossover operator is adopted in the later stage for exploitation.

Jia *et al.* [35] divided the evolutionary process into three situations, i.e., the infeasible situation, the semi-feasible situation, and the feasible situation. In different situations, different constraint-handling techniques are developed: multiobjective optimization for the infeasible situation and adaptive penalty function for the semi-feasible situation.

In [8], Wang *et al.* made use of DE/rand-to-best/1/bin to introduce information of objective function into the population. Meanwhile, DE/current-to-rand/1 is used to cope with rotated optimization problems.

Ghasemishabankareh *et al.* [54] exploited a popular DE variant (i.e., SaDE [55]) in a coevolution fashion and an improved augmented Lagrangian to deal with constraints.

Adaptive mechanisms are also used in multi-strategy CDE [56], where each trial vector generation strategy is adaptively selected according to its performance.

C. CDE Coupled with Other Operators

Recently, CDE coupled with other operators has also attracted much attention.

Dong and Wang [57] proposed a memetic DE for constrained optimization, in which DE/rand/1/bin serves as the global search operator while the simplex crossover [58] plays the role of local search. To handle constraints, a weight sum method which somehow likes penalty method is designed.

In [59], the mixed-integer hybridizing DE is combined with the Nelder-Mead simplex method [60] to solve mixed-integer constrained optimization. Additionally, the Lagrange method and self-adaptive penalty function are incorporated to deal with constraints.

Zhao *et al.* [61] integrated three algorithms, in which DE is responsible for accelerating the convergence at the later iteration process of the backtracking search algorithm [62],

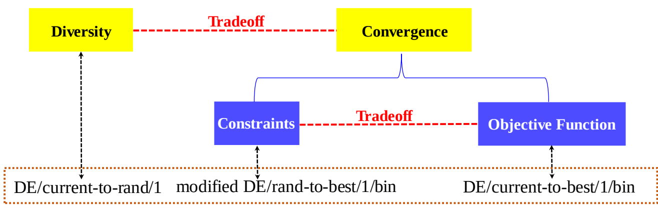


Fig. 2. Principle of the designed search algorithm.

the mutation operator of the breeder genetic algorithm [63] is employed to improve the population diversity, and the parameter-free penalty method is used to handle constraints.

Parouha and Das [64] hybridized DE with particle swarm optimization for constrained optimization. In this method, the optimized population is divided into three parts. Afterward, DE is used to evolve two of them and particle swarm optimization is used for the remaining one.

In [65], DE is combined with an improved teaching-learning-based optimization algorithm to solve constrained engineering design problems.

Tran *et al.* [66] hybridized DE with artificial bee colony for solving resource-constrained project scheduling problems.

Our work in this paper falls in the second class, i.e., attempting to design a search algorithm with multiple trial vector generation strategies to solve COPs.

IV. Proposed Method

A. Motivation

When applying EAs to solve COPs, two issues deserve much attention in order to obtain outstanding performance: 1) the tradeoff between diversity and convergence, and 2) the tradeoff between constraints and objective function. At present, more and more DE variants originally proposed for global optimization have been extended to search for the optimal solutions of COPs, due to their excellent search ability. Note, however, that in global optimization the essential purpose of the search algorithm is to balance diversity and convergence. As a consequence, the performance of most current CDE is limited due to the fact that the tradeoff between constraints and objective function has been neglected unreasonably in the search algorithm.

In view of the above drawback, this paper aims to make use of the idea of CoDE [20], a state-of-the-art DE variant, to design a search algorithm for constrained optimization. As pointed out previously, the search algorithm and constraint-handling technique are two important aspects of a constrained EA. Therefore, we also present a constraint-handling technique to suit the characteristics of CoDE. Additionally, a restart scheme is designed to tackle COPs with extremely complicated constraints. By assembling the above techniques together, an alternative CDE, i.e., C^{2}oDE, is proposed in this paper.

Next, the search algorithm, constraint-handling technique, restart scheme, and framework of C^{2}oDE are introduced one by one.

Algorithm 1: Search Algorithm

/*DE/current-to-rand/1*/
Select $\vec{x}^t_{r_1}$, $\vec{x}^t_{r_2}$, and $\vec{x}^t_{r_3}$ from the population;
Randomly choose a F value from F_{pool};
$\vec{v}^t_{i1} = \vec{x}^t_i + rand \cdot (\vec{x}^t_{r_1} - \vec{x}^t_i) + F \cdot (\vec{x}^t_{r_2} - \vec{x}^t_{r_3})$;
$\vec{u}^t_{i1} = \vec{v}^t_{i1}$;
/*Modified DE/rand-to-best/1/bin*/
Select $\vec{x}^t_{Gbest}$ (i.e., the individual with the least degree of constraint violation), $\vec{x}^t_{r_1}$, $\vec{x}^t_{r_2}$, $\vec{x}^t_{r_3}$, and $\vec{x}^t_{r_4}$ from the population;
Randomly choose a F value from F_{pool} and a CR value from CR_{pool};
$\vec{v}^t_{i2} = \vec{x}^t_{r_1} + F \cdot (\vec{x}^t_{Gbest} - \vec{x}^t_{r_2}) + F \cdot (\vec{x}^t_{r_3} - \vec{x}^t_{r_4})$;
Generate $\vec{u}^t_{i2}$ by applying the binomial crossover on $\vec{v}^t_{i2}$ and $\vec{x}^t_i$;
/*DE/current-to-best/1/bin*/
Select $\vec{x}^t_{fbest}$ (i.e., the individual with the best objective function value), $\vec{x}^t_{r_1}$, and $\vec{x}^t_{r_2}$ from the population ;
Randomly choose a F value from F_{pool} and a CR value from CR_{pool};
$\vec{v}^t_{i3} = \vec{x}^t_i + F \cdot (\vec{x}^t_{fbest} - \vec{x}^t_i) + F \cdot (\vec{x}^t_{r_1} - \vec{x}^t_{r_2})$;
Generate $\vec{u}^t_{i3}$ by applying the binomial crossover on $\vec{v}^t_{i3}$ and $\vec{x}^t_i$;

B. Search Algorithm

An ideal search algorithm for constrained optimization should not only reach a balance between diversity and convergence, but also between constraints and objective function. For this purpose, similar to CoDE, the designed search algorithm depicted in Fig. 2 involves three different trial vector generation strategies with distinct advantages. They are DE/current-to-rand/1, modified DE/rand-to-best/1/bin, and DE/current-to-best/1/bin.

As mentioned before, with respect to DE/current-to-rand/1 shown in Equation (7), each target vector $\vec{x}^t_i$ learns the information from a randomly selected individual $\vec{x}^t_{r_1}$; therefore, this trial vector generation strategy is able to promote the diversity of the population. In principle, DE/current-to-rand/1 can be decomposed into two steps: 1) implementing DE/rand/1 to generate the mutant vector $\vec{v}^t_i$ for $\vec{x}^t_i$, and 2) applying the arithmetic crossover on $\vec{x}^t_i$ and $\vec{v}^t_i$ as follows:

$$\vec{u}^t_i = \vec{x}^t_i + rand \cdot (\vec{v}^t_i - \vec{x}^t_i) \tag{13}$$

where $rand$ is a uniformly distributed random number on the interval [0,1]. As introduced in [20], [55], and [67], both DE/rand/1 and the arithmetic crossover are independent on the coordinate system and thus are rotation-invariant processes. As a result, DE/current-to-rand/1 is also beneficial to solve rotated optimization problems.

In terms of both the modified DE/rand-to-best/1/bin and DE/current-to-best/1/bin, the information of the "best" individual in the population is utilized to guide the search, thus accelerating the convergence. As shown in Equation (14), the modified DE/rand-to-best/1/bin is derived by replacing the second $\vec{x}^t_{r_1}$ in Equation (5) with a randomly selected individual $\vec{x}^t_{r_2}$ from the population:

$$\vec{v}^t_i = \vec{x}^t_{r_1} + F \cdot (\vec{x}^t_{best} - \vec{x}^t_{r_2}) + F \cdot (\vec{x}^t_{r_3} - \vec{x}^t_{r_4}) \tag{14}$$

The reason for this modification is explained as follows. There are two trial vector generation strategies for convergence and one trial vector generation strategy for diversity in the search algorithm, which might result in more biases toward convergence than diversity. By this modification, the modified DE/rand-to-best/1/bin has the potential to produce more disturbances than the original one. Thus, the tradeoff between diversity and convergence can be achieved in the search algorithm.

In addition, the "best" individual in the modified DE/rand-to-best/1/bin is chosen as the individual with the least degree of constraint violation while the "best" individual in DE/current-to-best/1/bin is selected as the individual with the best objective function value, with the aim of balancing constraints and objective function. Needless to say, the above balance is very important. It is because if the search is biased only toward constraints, the population might enter the feasible region with a very fast speed but subsequently converge to a local optimum in the feasible region due to the lack of diversity. On the other hand, the search biased only toward objective function would be very likely to get stuck in the infeasible region and could not find any feasible solution in the end. It should be noted that if multiple solutions have the same least degree of constraint violation or the same best objective function value, a random one is selected from them.

Overall, the proposed search algorithm provides an effective way to achieve the two desired tradeoffs in constrained optimization, the details of which are given in **Algorithm 1**. As shown in **Algorithm 1**, three offspring will be generated for each target vector. Moreover, similar to [8], we establish two parameter pools F_{pool} and CR_{pool} for the scaling factor F and the crossover control parameter CR, respectively.

C. Constraint-Handling Technique

In constrained evolutionary optimization, the constraint-handling technique is in charge of how to compare individuals. According to the characteristics of CoDE, the constraint-handling technique should include two phases as shown in Fig. 1: 1) how to preselect the best one from the three offspring as the trial vector, and 2) how to compare the target vector with its trial vector. According to the no free lunch theorem [68], [69] and [70], it is better to employ two different constraint-handling techniques rather than just one in the above two phases.

The feasibility rule is selected as one candidate owing to its attractive advantages, i.e., no additional parameters and the ability to rapidly motivate the population toward the feasible region. However, it is necessary to note that the feasibility rule prefers constraints to objective function and is a relatively greedy constrain-handling technique. Thus, we introduce the ε constrained method as the other candidate. From Equation (10), it can be seen that the ε constrained method also considers the information of objective function when comparing two individuals.

Obviously, there are two options to arrange these two constraint-handling techniques: 1) the ε constrained method in the first phase and the feasibility rule in the second phase, or 2) the feasibility rule in the first phase and the ε constrained method in the second phase. As shown in Fig. 1, the constraint-handling technique in the second phase determines which solution will survive into the next generation. In the case of option 1), the feasibility rule in the second phase might discard an individual with promising objective function value selected by the ε constrained method in the first phase. That is,

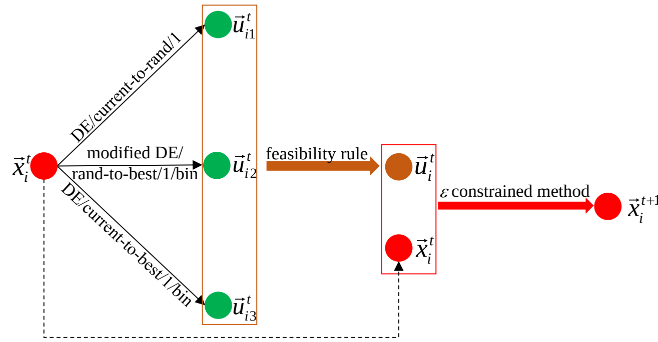


Fig. 3. Framework of C^{2}oDE.

option 1) would make the population bias toward constraints ultimately. In the case of option 2), although some biases are introduced by the feasibility rule in the first phase due to its preference to constraints, the ε constrained method in the second phase attempts to balance such biases by exploiting the information of objective function. Moreover, the degree that the information of objective function is exploited can be controlled by the parameter p in Equation (12). In summary, option 2) is adopted in this paper.

D. Restart Scheme

For some COPs with extremely complicated constraints, the infeasible region is highly nonlinear and always exhibits multimodal property. Under this condition, the population is very easy to stagnate in the infeasible region. To address this issue, a restart scheme is proposed in this paper.

Prior to applying the restart scheme, we need to judge whether the population has already been trapped into a local optimum in the infeasible region. It is intuitive that if the population clusters in a very small search range of the infeasible region, which means the difference/similarity among infeasible individuals is very tiny/high, then we can claim that premature convergence occurs in the infeasible region. However, how to measure the similarity among infeasible individuals should be studied in depth.

A possible way is to compute the average Euclidean distance among all the individuals or the average standard deviation of all the dimensions of the population. If such indicator is less than a specified threshold, then one can conclude that the similarity among all the individuals is very high. Nevertheless, it is not trivial to set an appropriate threshold since different problems possess different dimensions and search spaces. Considering this, we use a unitary indicator, i.e., the degree of constraint violation or objective function value, to measure the similarity of the population. It is believed that this unitary indicator is less sensitive to different problems.

Consequently, if the standard deviation of the degree of constraint violation or the standard deviation of objective function values of the population is less than a predefined threshold μ and if the population is infeasible, the restart scheme is triggered – all the individuals in the population are regenerated from the decision space randomly without any special skills.

Algorithm 2: C^{2}oDE

Input: NP: the population size
$MaxFEs$: the maximum number of function evaluations
F_{pool}: the pool of the scaling factor F
CR_{pool}: the pool of the crossover control parameter CR

1 t=1; /*t denotes the generation number*/
2 Randomly generate an initial population $\boldsymbol{P_t}$=$\{\vec{x}_1^t, ..., x_{NP}^t\}$ from the decision space S;
3 Evaluate the objective function values and the degree of constraint violation of $\boldsymbol{P_t}$;
4 $FEs = NP$; /*FEs denotes the number of fitness evaluations*/
5 Tune the ε value of the ε constrained method according to Equation (11);
6 $\boldsymbol{P_{t+1}} = \varnothing$;
7 **for** $i = 1 : NP$ **do**
8   Implement the search algorithm (**Algorithm 1**) to generate three offspring $\vec{u}_{i1}^t$, $\vec{u}_{i2}^t$, and $\vec{u}_{i3}^t$ for the target vector $\vec{x}_i^t$;
9   Evaluate the objective function values and the degree of constraint violation of $\vec{u}_{i1}^t$, $\vec{u}_{i2}^t$, and $\vec{u}_{i3}^t$;
10   Apply the feasibility rule to select the best one among $\vec{u}_{i1}^t$, $\vec{u}_{i2}^t$, and $\vec{u}_{i3}^t$ as the trial vector $\vec{u}_i^t$ for $\vec{x}_i^t$;
11   Apply the ε constrained method to compare $\vec{x}_i^t$ and $\vec{u}_i^t$, and store the better one into $\boldsymbol{P_{t+1}}$;
12     $FEs = FEs + 3$;
13 Implement the restart scheme;
14 $t = t + 1$;
15 **Stopping Criterion**: If $FEs \geq MaxFEs$, then stop and output the best individual in $\boldsymbol{P^t}$, otherwise go to Step 5.

E. C^{2}oDE

By integrating three important components, i.e., the search algorithm, constraint-handling technique, and restart scheme, C^{2}oDE is obtained. The framework of C^{2}oDE is given in Fig. 3. C^{2}oDE maintains a population consisting of NP target vectors: $\boldsymbol{P_t}$=$\{\vec{x}_1^t, \vec{x}_2^t, ..., \vec{x}_{NP}^t\}$, their objective function values: $f(\vec{x}_1^t), f(\vec{x}_2^t), ..., f(\vec{x}_{NP}^t)$, and their degree of constraint violation: $G(\vec{x}_1^t), G(\vec{x}_2^t), ..., G(\vec{x}_{NP}^t)$. As shown in Fig. 3, at generation t, three trial vector generation strategies are employed to generate three offspring ($\vec{u}_{i1}^t$, $\vec{u}_{i2}^t$, and $\vec{u}_{i3}^t$) for each target vector $\vec{x}_i^t$. Afterward, the feasibility rule is used to preselect the best offspring as the trial vector $\vec{u}_i^t$. And then, the ε constrained method is utilized to compare $\vec{x}_i^t$ and $\vec{u}_i^t$. Finally, the restart scheme is executed. The above procedure is repeated until the maximum number of fitness evaluations ($MaxFES$) is reached. The details of C^{2}oDE are presented in **Algorithm 2**. From Fig. 3 and **Algorithm 2**, it can be seen that the implementation of C^{2}oDE is simple.

Remark 1: Compared with other existing multi-strategy CDE, the advantages of C^{2}oDE are summarized from the following three aspects:

- The search algorithm of C^{2}oDE takes both the tradeoff between constraints and objective function and the tradeoff between diversity and convergence into account.
- Two well-known constraint-handling techniques with complementary properties are combined in an effective way for selection.
- Its computational time complexity is the same with the classical DE without any additional computation burden.

V. Experimental Study

A. Benchmark Test Functions and Parameter Settings

Two sets of benchmark test functions were selected to assess the performance of C^{2}oDE. The first set contains 24 test functions at IEEE CEC2006 [21], and the second set contains 18 test functions with 10 dimensions (10D) and 30 dimensions (30D) at IEEE CEC2010 [22]. These 60 test functions can systematically investigate the performance of a constrained EA since they exhibit a variety of characteristics such as different dimensions of decision space, different types of objective function (i.e., linear, nonlinear, quadratic, polynomial, and cubic), and different kinds of constraints (i.e., linear/nonlinear and equality/inequality). All these test functions are minimization problems and their details can be found in [21] and [22].

TABLE I
Maximum Number of Function Evaluations $MaxFEs$ and Population Size NP

Test Functions	$MaxFEs$	NP
24 test functions from IEEE CEC2006	2.4E+05	50
18 test functions with 10D from IEEE CEC2010	2.0E+05	35
18 test functions with 30D from IEEE CEC2010	6.0E+05	60

For the experiments in this paper, the settings of $MaxFEs$ and the population size NP are given in Table I. Note that a proper setting of NP is related to the benchmark test suite as well as the dimension of a test function. In addition, 25 independent runs were performed for each test function and the tolerance value δ for equality constraints was set to 10^{-4}. As the same with [8], F_{pool} = [0.6, 0.8, 1.0] and CR_{pool} = [0.1, 0.2, 1.0]. Meanwhile, p in the ε constrained method and μ in the restart scheme were set to 0.5 and 10^{-8}, respectively.

B. Experiments on IEEE CEC2006 Test Suite

Firstly, C^{2}oDE was applied to solve 24 test functions from IEEE CEC2006. The performance of C^{2}oDE was compared with that of four state-of-the-art CDE (i.e., CMODE [40], FROFI [8], NDE [71], and DW [72]). From [21], we know that it is extremely difficult to find a feasible solution for g22 and there are no feasible solutions for g20. Thus, we excluded these two functions and focused on the remaining 22 test functions. The experimental results are given in Table II, where "Mean OFV" and "Std Dev" denote the average and standard deviation of the objective function values obtained over 25 independent runs, respectively. For each test function, a run is successful if the following success condition is satisfied: $f(\vec{x}_{best})$-$f(\vec{x}^*) \leq$ 0.0001, where $\vec{x}^*$ is the best-known solution and $\vec{x}_{best}$ is the best feasible solution provided by a method. In Table II, "*" means that a method can satisfy the success condition in all 25 runs for a test function.

As shown in Table II, among the five compared CDE, CMODE, FROFI, and C^{2}oDE successfully solve all the 22 test functions. NDE fails to consistently find the optimal solution of g02. DW cannot attain the optimal solution of g17 consistently. The experimental results demonstrate that, overall, C^{2}oDE presents better or similar performance compared with the four competitors on the 22 test functions from IEEE CEC2006.

C. Experiments on IEEE CEC2010 Test Suite

In this subsection, the performance of C^{2}oDE was further tested by making use of other 36 test functions from IEEE

TABLE II
EXPERIMENTAL RESULTS OF C^{2}oDE AND OTHER FOUR SELECTED METHODS OVER 25 INDEPENDENT RUNS ON 22 TEST FUNCTIONS FROM IEEE CEC2006

IEEE CEC2006	CMODE Mean OFV±Std Dev	FROFI Mean OFV±Std Dev	NDE Mean OFV±Std Dev	DW Mean OFV±Std Dev	C^{2}oDE Mean OFV±Std Dev
g01	-1.5000E+01±7.62E-12*	-1.5000E+01±0.00E+00*	-1.50000E+01±0.00E+00*	-1.5000E+01±5.02E-14*	-1.5000E+01±0.00E+00*
g02	-8.0362E-01±2.28E-07*	-8.0362E-01±1.02E-05*	-8.01809E-01±5.10E-04	-8.0362E-01±9.99E-08*	-8.0362E-01±3.82E-07*
g03	-1.0005E+00±9.78E-09*	-1.0005E+00±1.25E-12*	-1.0005E+00±0.00E+00*	-1.0005E+00±4.27E-12*	-1.0005E+00±4.67E-16*
g04	-3.0666E+04±2.34E-26*	-3.066553E+04±3.71E-12*	-3.066553E+04±0.00E+00*	-3.066553E+04±0.00E+00*	-3.066553E+04±3.71E-12*
g05	5.1265E+03±1.36E-26*	5.1264967E+03±5.68E-11*	5.1264967E+03±0.00E+00*	5.1264967E+03±4.22E-10*	5.1265E+03±2.78E-12*
g06	-6.9618E+03±1.32E-26*	-6.961813E+03±1.37E-25*	-6.961813E+03±0.00E+00*	-6.961813E+03±0.00E+00*	-6.961813E+03±0.00E+00*
g07	2.4306E+01±6.41E-09*	2.430621E+01±5.32E-14*	2.430621E+01±1.35E-14*	2.430621E+01±5.28E-10*	2.4306E+01±5.01E-13*
g08	-9.5825E-02±1.32E-14*	-9.5825E-02±3.58E-15*	-9.5825E-02±0.00E+00*	-9.5825E+02±2.78E-18*	-9.5825E-02±1.42E-17*
g09	6.8063E+02±9.85E-14*	6.8063006E+02±2.73E-11*	6.8063006E+02±0.00E+00*	6.8063006E+02±2.23E-11*	6.8063006E+02±3.20E-13*
g10	7.0492480E+03±3.15E-09*	7.0492480E+03±8.32E-11*	7.0492480E+03±3.41E-09*	7.0492480E+03±3.26E-12*	7.0492480E+03±7.33E-09*
g11	7.499E-01±0.00E+00*	7.499E-01±8.54E-15*	7.499E-01±0.00E+00*	7.499E-01±1.13E-16*	7.499E-01±1.13E-16*
g12	-1.00E+00±0.00E+00*	-1.00E+00±0.00E+00*	-1.00E+00±0.00E+00*	-1.00E+00±0.00E+00*	-1.00E+00±0.00E+00*
g13	5.3942E-02±1.25E-17*	5.3942E-02±2.68E-15*	5.3942E-02±0.00E+00*	5.3942E-02±6.03E-14*	5.3942E-02±1.69E-17*
g14	-4.776489E+01±3.79E-11*	-4.776489E+01±5.12E-12*	-4.776489E+01±5.14E-15*	-4.776489E+01±3.47E-10*	-4.776489E+01±2.24E-14*
g15	9.617150E+02±0.00E+00*	9.617150E+02±5.03E-10*	9.617150E+02±0.00E+00*	9.617150E+02±4.47E-13*	9.617150E+02±5.80E-13*
g16	-1.90516E+00±2.48E-16*	-1.90516E+00±9.33E-15*	-1.90510E+00±0.00E+00*	-1.90516E+00±0.00E+00*	-1.90516E+00±4.53E-16*
g17	8.853533E+03±2.06E-06*	8.853533E+03±4.32E-15*	8.853533E+03±0.00E+00*	8.880233E+03±3.63E+01	8.853533E+03±8.92E-06*
g18	-8.66025E-01±3.45E-11*	-8.66025E-01±2.43E-08*	-8.66025E-01±0.00E+00*	-8.66025E-01±3.30E-07*	-8.66025E-01±1.66E-07 *
g19	3.265559E+01±7.79E-06*	3.265559E+01±1.19E-09*	3.265559E+01±3.73E-05*	3.265559E+01±3.37E-07*	3.265559E+01±1.01E-06*
g21	1.937245E+02±8.32E-05*	1.937245E+02±3.86E-09*	1.937245E+02±6.26E-11*	1.937245E+02±3.66E-09*	1.937245E+02±2.87E-10*
g23	-4.000551E+02±1.10E-05*	-4.000551E+02±1.61E-09*	-4.000551E+02±3.45E-09*	-4.000551E+02±6.49E-06*	-4.000551E+02±5.24E-08*
g24	-5.50801E+00±1.23E-25*	-5.50801E+00±1.36E-13*	-5.50801E+00±0.00E+00*	-5.50801E+00±0.00E+00*	-5.50801E+00±1.35E-16*
*	**22**	**22**	**21**	**21**	**22**

TABLE III
EXPERIMENTAL RESULTS OF C^{2}oDE AND OTHER FIVE SELECTED METHODS OVER 25 INDEPENDENT RUNS ON 18 TEST FUNCTIONS WITH 10D FROM IEEE CEC2010

IEEE CEC2010 with 10D	CMODE Mean OFV±Std Dev	FROFI Mean OFV±Std Dev	ECHT-DE Mean OFV±Std Dev	AIS-IRP Mean OFV±Std Dev	Co-CLPSO Mean OFV±Std Dev	C^{2}oDE Mean OFV±Std Dev
C01	-7.47E-01±2.35E-13 +	-7.47E-01±1.35E-03 +	-7.47E-01±1.40E-03+	-7.47E-01±1.30E-03+	-7.34E-01±1.78E-02−	-7.44E-01±7.39E-03
C02	-1.48E+00±4.88E-01∇ −	-2.02E+00±1.41E-01 −	-2.27E+00±6.70E-03 ≈	-2.27E+00±2.00E-03≈	-2.27E+00±1.46E-02≈	-2.26E+00±4.64E-02
C03	2.84E+00±4.23E+00 −	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	3.75E-09±4.81E-04−	3.55E-01±1.78E+00−	0.00E+00±0.00E+00
C04	-9.99E-04±2.90E-08 −	-1.00E-05±0.00E+00 ≈	-1.00E-05±0.00E+00≈	-9.97E-06±4.28E-03−	9.34E-06±1.07E-06−	-1.00E-05±0.00E+00
C05	-4.50E+02±1.61E+02∇ −	-4.84E+02±8.09E-07≈	-4.11E+02±7.63E+01−	-4.80E+02±6.30E+00−	-4.84E+02±1.98E-02≈	-4.84E+02±3.48E-13
C06	-5.78E+02±1.60E-02 −	-5.79E+02±5.04E-04 ≈	-5.62E+02±4.51E+01−	-5.80E+02±7.30E-08+	-5.79E+02±5.73E-04≈	-5.79E+02±6.17E-02
C07	6.69E-15±8.95E-15 −	0.00E+00±0.00E+00≈	1.33E-01±7.28E-01−	1.17E-08±2.70E+00−	7.97E-01±1.63E+00−	0.00E+00±0.00E+00
C08	8.94E+00±3.98E+00 −	7.11E+00±4.79E+00≈	6.16E+00±6.45E+00+	4.09E+00±1.46E+00+	6.09E-01±1.43E+00+	7.30E+00±5.18E+00
C09	2.13E+06±1.04E+07∇ −	2.50E+01±3.92E+01−	1.47E-01±8.05E-01+	2.70E+01±7.50E+01−	1.99E+10±9.97E+10−	5.17E+00±5.19E+01
C10	1.35E+05±1.61E+06∇ −	4.17E+01±8.69E-06−	1.71E+00±7.66E+00+	1.62E+03±5.00E+02−	4.97E+10±2.49E+11−	3.67E+01±1.38E+01
C11	-7.7E-02±2.85E-02∇ −	-1.52E-03±5.63E-14≈	-4.40E-03±1.57E-02∇ −	-9.20E-04±8.23E-04−	-1.61E-01±6.60E-01∇ −	-1.52E-03±4.89E-13
C12	-6.14E+02±2.74E+02∇ −	-3.84E+02±2.17E+02+	-1.72E+02±2.21E+02∇ −	-4.36E+02±6.02E+01+	-2.34E+00±2.43E+01−	-7.63E+01±1.22E+02
C13	-5.79E+01±4.09E+00 −	-6.84E+01±2.52E-09≈	-6.51E+01±2.38E+00−	-6.79E+01±3.11E-01−	-6.53E+01±2.58E+00−	-6.84E+01±2.77E-14
C14	8.18E-09±1.64E-08 −	0.00E+00±0.00E+00≈	7.02E+05±3.19E+06−	1.22E-04±2.90E-08−	3.19E-01±1.10E+00−	0.00E+00±0.00E+00
C15	1.20E+02±3.48E+02 −	3.09E+00±1.37E+00 +	2.34E+13±5.30E+13−	5.19E-09±1.10E-08+	2.99E+00±3.31E+00+	3.71E+00±1.65E-01
C16	6.82E-05±1.49E-04 −	1.19E-02±2.07E-02 −	3.93E-02±4.28E-02−	9.96E-18±6.27E-15−	5.99E-03±1.33E-02−	0.00E+00±0.00E+00
C17	4.37E-02±1.12E-01 −	7.83E-02±2.25E-01 −	1.12E-01±3.32E-01−	2.93E+00±2.29E+00−	3.80E-01±4.53E-01−	1.61E-02±8.04E-02
C18	5.75E+00±2.64E+02 −	5.23E-26±1.71E-25 −	0.00E+00±0.00E+00≈	1.66E+00±1.27E+00−	2.32E-01±9.96E-01−	0.00E+00±0.00E+00
−	**17**	**6**	**10**	**12**	**13**	/
+	**1**	**3**	**4**	**5**	**2**	/
≈	**0**	**9**	**4**	**1**	**3**	/

TABLE IV
RESULTS OF THE MULTIPLE-PROBLEM WILCOXON'S TEST FOR C^{2}oDE AND OTHER FIVE SELECTED METHODS ON 18 TEST FUNCTIONS WITH 10D FROM IEEE CEC2010

C^{2}oDE VS	R^+	R^-	p-value	α=0.1	α=0.05
CMODE	158.5	12.5	3.3000E-05	Yes	Yes
FROFI	98.0	55.0	2.5616E-01	No	No
ECHT-DE	123.5	29.5	5.5410E-03	Yes	Yes
AIS-IRP	111.0	42.0	1.1649E-02	Yes	Yes
Co-CLPSO	139.0	14.0	3.6000E-04	Yes	Yes

TABLE V
RANKING OF C^{2}oDE AND OTHER FIVE SELECTED METHODS BY THE FRIEDMAN'S TEST ON 18 TEST FUNCTIONS WITH 10D FROM IEEE CEC2010

Algorithm	Ranking
C^{2}oDE	**2.2778**
FROFI	2.7222
AIS-IRP	3.2222
ECHT-DE	3.8889
Co-CLPSO	4.25
CMODE	4.6389

CEC2010 (18 test functions with 10D and 18 test functions with 30D), which are more complicated than the 24 test functions from IEEE CEC2006. For the purpose of comparison, five competitive methods were selected. Among them, three are CDE (i.e., CMODE [40], FROFI [8], and ECHT-DE [73]), and two are other constrained EAs (i.e., AIS-IRP [74] and Co-CLPSO [75]).

Due to the fact that the optimal solutions of this test suite cannot be known *a priori*, the average and standard deviation of the objective function values derived from a method over 25 independent runs were recorded. Afterward, statistical tests were implemented to compare C^{2}oDE with each competitor. Specifically, we applied the Wilcoxon's rank sum test at a 0.05 significance level to compare C^{2}oDE with each of CMODE and FROFI. It is because the objective function values of CMODE and FROFI in 25 runs can be available from our previous study. In addition, only the average and standard deviation of objective function values can be obtained from the original papers of ECHT-DE, AIS-IRP and Co-CLPSO.

TABLE VI

EXPERIMENTAL RESULTS OF C^{2}oDE AND OTHER FIVE SELECTED METHODS OVER 25 INDEPENDENT RUNS ON 18 TEST FUNCTIONS WITH 30D FROM IEEE CEC2010

IEEE CEC2010 with 30D	CMODE Mean OFV±Std Dev	FROFI Mean OFV±Std Dev	ECHT-DE Mean OFV±Std Dev	AIS-IRP Mean OFV±Std Dev	Co-CLPSO Mean OFV±Std Dev	C^{2}oDE Mean OFV±Std Dev
C01	-8.21E-01±3.3E-03 ≈	-8.21E-01±2.36E-03≈	-8.00E-01±1.79E-02−	-8.20E-01±3.25E-04≈	-7.16E-01±5.03E-02−	-8.20E-01±2.52E-03
C02	9.75E-01±6.25E+01−	-2.00E+00±4.35E-02−	-1.99E+00±2.10E-01−	-2.21E+00±2.84E-03≈	-2.20E+00±1.93E-01≈	-2.22E+00±5.20E-02
C03	2.18E+01±1.25E+01 ≈	2.87E+01±6.24E-08≈	9.89E+01±6.26E+01−	6.68E+01±4.26E+02−	3.51E+01±3.31E+01∇ −	3.06E+01±2.12E+01
C04	6.72E-04±4.24E-04 −	-3.33E-06±4.13E-10+	-1.03E-06±9.01E-03+	1.98E-03±1.61E-03−	1.13E-01±5.63E-01∇ −	5.46E-06±2.75E-05
C05	2.77E+02±2.03E+02∇ −	-4.81E+02±2.84E+00≈	-1.06E+02±1.67E+02−	-4.36E+02±2.51E+01−	-3.12E+02±8.83E+01−	-4.82E+02±7.02E-01
C06	-4.96E+02±2.15E+02 ∇ −	-5.29E+02±5.71E-01−	-1.38E+02±9.89E+01−	-4.54E+02±4.79E+01−	-2.45E+02±3.95E+01−	-5.31E+02±8.97E-02
C07	5.24E-05±5.89E-05 −	0.00E+00±0.00E+00≈	1.33E-01±7.28E-01−	1.07E+00±1.61E+00−	1.12E+00±1.83E+00−	0.00E+00±0.00E+00
C08	3.68E-01±2.62E-01 −	0.00E+00±0.00E+00≈	3.36E+01±1.11E+02−	1.65E+00±6.41E-01−	4.75E+01±1.13E+02−	0.00E+00±0.00E+00
C09	1.72E+13±1.07E+13∇ −	4.30E+01±3.27E+01−	4.24E+01±1.38E+02−	1.57E+00±1.96E+00≈	1.48E+08±2.45E+08−	1.85E+00±4.90E+00
C10	1.60E+13±7.00E+12∇ −	3.13E+01±8.22E-02≈	5.34E+01±8.83E+01≈	1.78E+01±1.88E+01+	1.40E+09±5.84E+09−	3.13E+01±5.73E-06
C11	9.5E-03±9.7E-03∇ −	-3.92E-04±2.64E-06≈	2.60E-03±6.00E-03∇ −	-1.58E-04±4.67E-05−	2.82E-02±3.21E-02∇ −	-3.92E-04±1.60E-06
C12	-3.46E+00±7.35E+02∇ −	-1.99E-01±1.42E-06≈	2.51E+01±1.37E+02∇ −	4.29E-06±4.52E-04−	-1.99E-01±1.18E-04∇ −	-1.99E-01±3.09E-07
C13	-3.89E+01±2.17E+00 −	-6.83E+01±1.95E-01≈	-6.46E+01±1.67E+00−	-6.62E+01±2.27E-01−	-6.08E+01±1.12E+00−	-6.81E+01±6.25E-01
C14	9.31E+00±2.46E+00 −	9.80E-29±4.90E-28≈	1.24E+05±6.77E+05−	8.68E-07±3.14E-07−	1.28E+00±1.90E+00−	0.00E+00±0.00E+00
C15	1.51E+13±8.26E+12 −	2.16E+01±8.03E-05≈	1.94E+11±4.35E+11−	3.41E+01±3.82E+01−	5.11E+01±9.18E+01−	2.16E+01±2.92E-07
C16	6.30E-02±2.72E-02 −	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	8.21E-02±1.12E-01−	5.24E-16±4.67E-16−	0.00E+00±0.00E+00
C17	3.12E+02±2.75E+02 ∇ −	1.59E-01±3.82E-01−	2.75E-01±3.78E-01−	3.61E+00±2.54E+00−	1.39E+00±4.26E+00−	6.58E-02±1.46E-01
C18	7.36E+03±3.12E+03 −	4.87E-01±1.25E+00−	0.00E+00±0.00E+00+	4.02E+01±1.80E+01−	1.09E+01±3.72E+01−	4.47E-20±2.24E-19
−	**16**	**5**	**14**	**14**	**17**	/
+	**0**	**1**	**2**	**1**	**0**	/
≈	**2**	**12**	**2**	**3**	**1**	/

TABLE VII

RESULTS OF THE MULTIPLE-PROBLEM WILCOXON'S TEST FOR C^{2}oDE AND OTHER FIVE SELECTED METHODS ON 18 TEST FUNCTIONS WITH 30D FROM IEEE CEC2010

C^{2}oDE VS	R^+	R^-	p-value	α=0.1	α=0.05
CMODE	169.5	1.5	1.91E-05	Yes	Yes
FROFI	111.5	41.5	7.77E-02	Yes	No
ECHT-DE	166.0	5.0	7.63E-05	Yes	Yes
AIS-IRP	148.5	4.5	1.30E-04	Yes	Yes
Co-CLPSO	153.0	0.0	1.53E-05	Yes	Yes

TABLE VIII

RANKING OF C^{2}oDE AND OTHER FIVE SELECTED METHODS BY THE FRIEDMAN'S TEST ON 18 TEST FUNCTIONS WITH 30D FROM IEEE CEC2010

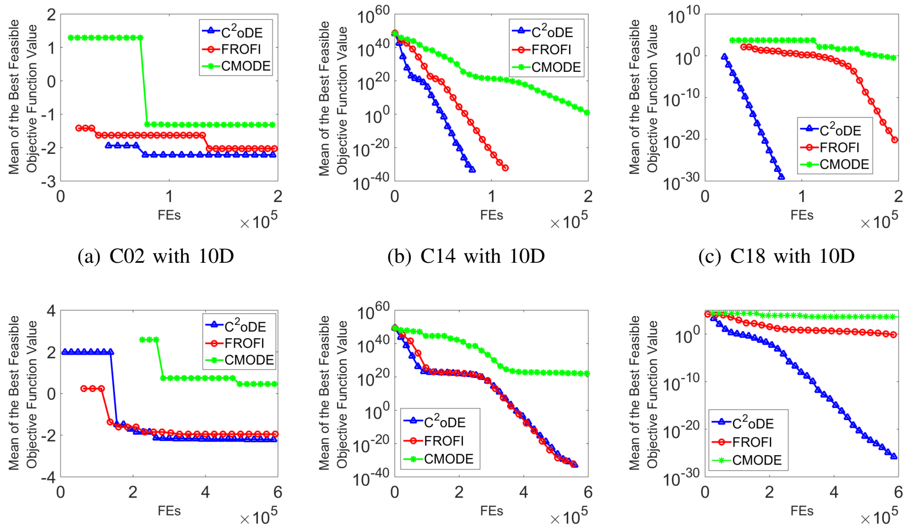
Algorithm	Ranking
C^{2}oDE	**1.6944**
FROFI	2.1111
AIS-IRP	3.4444
ECHT-DE	4.1111
Co-CLPSO	4.7222
CMODE	4.9167

As a result, the t-test at a 0.05 significance level was used to compare C^{2}oDE with each of ECHT-DE, AIS-IRP and Co-CLPSO. When a method obtains the smallest average objective function value on a test function, the corresponding experimental results are highlighted in gray. Furthermore, the multiple-problem Wilcoxon's test and the Friedman's test were implemented via KEEL software [76]. Note that the *post-hoc* test of the Friedman's test is based on the Bonferroni-Dunn method.

In terms of the 18 test functions with 10D from IEEE CEC2010, Tables III, IV, and V summarize the average and standard deviation of objective function values, results of the multiple-problem Wilcoxon's test, and results of the Friedman's test, respectively. In Table III, "∇" means that feasible solutions cannot be found by the corresponding method at the end of some runs. Additionally, "$-$", "$+$" and "$\approx$" denote that the performance of the corresponding method is worse than, better than, and similar to that of C^{2}oDE, respectively, according to the Wilcoxon's rank sum test/t-test. From Table III, it can be seen that C^{2}oDE outperforms CMODE, FROFI, ECHT-DE, AIS-IRP, and Co-CLPSO on 17, six, 10, 12, and 13 test functions, respectively. In contrast, CMODE, FROFI, ECHT-DE, AIS-IRP, and Co-CLPSO perform better than C^{2}oDE on one, three, four, five, and two test functions, respectively. According to the multiple-problem Wilcoxon's test in Table IV, the R^+ values are bigger than the R^- values in all cases. Moreover, the significant differences can be observed in four cases at α=0.05, i.e., C^{2}oDE versus CMODE, C^{2}oDE versus ECHT-DE, C^{2}oDE versus AIS-IRP, and C^{2}oDE versus Co-CLPSO. As far as the Friedman's test is concerned, C^{2}oDE achieves the first rank followed by FROFI. Taking all these results into consideration, we can conclude that C^{2}oDE has an edge over the five competitors on the 18 test functions with 10D from IEEE CEC2010.

In terms of the 18 test functions with 30D from IEEE CEC2010, Tables VI, VII, and VIII record the average and standard deviation of objective function values, results of the multiple-problem Wilcoxon's test, and results of the Friedman's test, respectively. As shown in Table VI, C^{2}oDE surpasses CMODE, FROFI, ECHT-DE, AIS-ISP, and Co-CLPSO on 16, five, 14, 14, and 17 test functions, respectively. However, the performance of FROFI, ECHT-DE, and AIS-ISP is better than that of C^{2}oDE on only one, two, and one test function, respectively. In particular, CMODE and Co-CLPSO cannot beat C^{2}oDE even on one test function. Regarding the multiple-problem Wilcoxon's test, C^{2}oDE provides higher R^+ values than R^- values in all cases. Moreover, the p-values are less than 0.1 in all cases and less than 0.05 in four cases, i.e., C^{2}oDE versus CMODE, C^{2}oDE versus ECHT-DE, C^{2}oDE versus AIS-IRP, and C^{2}oDE versus Co-CLPSO. With respect to the Friedman's test, C^{2}oDE ranks the first followed by FROFI. In conclusion, C^{2}oDE provides superior results on the 18 test functions with 30D from IEEE CEC2010. Moreover, it seems that the advantage of C^{2}oDE over the five competitors increases as the number of dimension increases.

To visualize the experimental results, the convergence graphs of C^{2}oDE, FROFI, and CMODE were plotted on six



(a) C02 with 10D (b) C14 with 10D (c) C18 with 10D

(d) C02 with 30D (e) C14 with 30D (f) C18 with 30D

Fig. 4. Convergence graphs of C^{2}oDE, FROFI, and CMODE on six representative test functions from IEEE CEC2010.

TABLE IX
EXPERIMENTAL RESULTS OF C^{2}ODE AND CODE OVER 25 INDEPENDENT RUNS ON 36 TEST FUNCTIONS FROM IEEE CEC2010

Instance	10D		30D	
	C^{2}oDE Mean OFV±Std Dev (feasible rate)	CoDE Mean OFV±Std Dev (feasible rate)	C^{2}oDE Mean OFV±Std Dev (feasible rate)	CoDE Mean OFV±Std Dev (feasible rate)
C01	-7.44E-01±7.39E-03	-7.47E-01±1.88E-03 ≈	-8.20E-01±2.52E-03	-8.11E-01±1.69E-03 −
C02	-2.26E+00±4.64E-02	-1.34E+00±7.11E-01 −	-2.22E+00±5.20E-02	9.21E-01±1.06E+00 −
C03	0.00E+00±0.00E+00	3.55E-01±1.78E+00 −	3.06E+01±2.12E+01	(0%) −
C04	-1.00E-05±0.00E+00	-6.74E-06±2.63E-06 −	5.46E-06±2.75E-05	(0%) −
C05	-4.84E+02±3.48E-13	(36%) −	-4.82E+02±7.02E-01	(0%) −
C06	-5.79E+02±6.17E-02	(28%) −	-5.31E+02±8.97E-02	(0%) −
C07	0.00E+00±0.00E+00	7.37E-25±2.41E-24 −	0.00E+00±0.00E+00	1.49E+01±2.51E+00 −
C08	7.30E+00±5.18E+00	1.71E+00±3.99E+00 +	0.00E+00±0.00E+00	6.56E+01±4.65E+01 −
C09	5.17E+00±5.19E+01	4.13E-24±5.38E-24 +	1.85E+00±4.90E+00	(24%) −
C10	3.67E+01±1.38E+01	2.17E+01±2.13E+01 +	3.13E+01±5.73E-06	(4%) −
C11	-1.52E-03±4.89E-13	-1.52E-03±1.39E-07 ≈	-3.92E-04±1.60E-06	(0%) −
C12	-7.63E+01±1.22E+02	(84%) −	-1.99E-01±3.09E-07	(4%) −
C13	-6.84E+01±2.77E-14	-6.84E+01±1.83E-02≈	-6.81E+01±6.25E-01	-5.92E+01±8.71E-01 −
C14	0.00E+00±0.00E+00	5.09E-25±8.48E-25−	0.00E+00±0.00E+00	2.12E+01±1.93E+00 −
C15	3.71E+00±1.65E-01	2.06E+00±1.86E+00+	2.16E+01±2.92E-07	4.39E+12±2.64E+12 −
C16	0.00E+00±0.00E+00	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00	3.39E-05±2.09E-05 −
C17	1.61E-02±8.04E-02	1.17E-04±1.86E+00+	6.58E-02±1.46E-01	(60%) −
C18	0.00E+00±0.00E+00	2.50E+01±1.10E+02−	4.47E-20±2.24E-19	1.19E+04±8.70E+03 −
−	/	**9**	/	**18**
+	/	**5**	/	**0**
≈	/	**4**	/	**0**

representative test functions from IEEE CEC2010, i.e., C02 with 10D, C14 with 10D, C18 with 10D, C02 with 30D, C14 with 30D, and C18 with 30D. Since the source codes of ECHT-DE, AIS-IRP, and Co-CLPSO cannot be available, their convergence graphs are not provided. Fig. 4 depicts the evolution of the mean of the best feasible objective function value. As shown in Fig. 4, C^{2}oDE converges faster than CMODE on all these six test functions. In addition, C^{2}oDE shows faster convergence speed than FROFI on all these six test functions except for C14 with 30D. In terms of C14 with 30D, C^{2}oDE and FROFI have similar convergence speed.

According to the above comprehensive experiments on two benchmark test sets, C^{2}oDE exhibits very competitive performance when tackling COPs.

D. Comparing C^{2}oDE with the Original CoDE for Constrained Optimization

The aim of this subsection is to ascertain whether the original CoDE designed for global optimization can be directly applied to solve COPs. To this end, the search algorithm of C^{2}oDE was replaced with the original CoDE. Subsequently, the 36 test functions from IEEE CEC2010 were used to produce the experimental results for CoDE. The average and standard deviation of objective function values over 25 runs are summarized in Table IX. It is noteworthy that the feasible rate, i.e., percentage of runs where at least one feasible solution is found, is recorded if an algorithm fails to consistently provide feasible solutions over all 25 runs. In addition, the Wilcoxon's rank sum test at a 0.05 significance level was executed to compare C^{2}oDE with CoDE. The cell with the smaller average objective function value is highlighted in gray.

As shown in Table IX, overall, CoDE performs better than, similar to, and worse than C^{2}oDE on five, four, and 27 test functions, respectively. More importantly, CoDE cannot consistently find feasible solutions in 12 cases. Therefore, the above comparison indicates that the original CoDE without any modifications is not a good choice as the search algorithm for constrained optimization, which verifies the motivation of this paper.

E. Contribution of the Feasibility Rule and the ε Constrained Method

In this paper, our constraint-handing technique includes two phases. Moreover, the feasibility rule and the ε constrained method are used for the first and second phases, respectively. In order to identify their main contribution, two C^{2}oDE variants, i.e., C^{2}oDE-FR and C^{2}oDE-ECM, were implemented. To be specific, in C^{2}oDE-FR, the feasibility rule was utilized in both phases while in C^{2}oDE-ECM, the ε constrained method was utilized in both phases. The 18 test functions with 30D from IEEE CEC2010 were employed to produce the experimental results.

The average and standard deviation of objective function values over 25 runs, and the feasible rate are summarized

TABLE X
EXPERIMENTAL RESULTS OF C²oDE, C²oDE-FR, AND C²oDE-ECM OVER 25 INDEPENDENT RUNS ON 18 TEST FUNCTIONS WITH 30D FROM IEEE CEC2010

Instance	C²oDE Mean OFV±Std Dev (feasible rate)	C²oDE-FR Mean OFV±Std Dev (feasible rate)	C²oDE-ECM Mean OFV±Std Dev (feasible rate)
C01	-8.20E-01±2.52E-03	-8.16E-01±6.59E-03≈	-8.20E-01±2.14E-03≈
C02	-2.22E+00±5.20E-02	2.70E+00±9.16E-01−	-2.23E+00±5.00E-02≈
C03	3.06E+01±2.12E+01	1.74E+13±5.10E+13 −	3.35E+01±2.14E+01≈
C04	5.46E-06±2.75E-05	-3.28E-06±1.57E-07 +	6.23E-06±1.35E-05≈
C05	-4.82E+02±7.02E-01	4.49E+02±1.18E+02−	-4.81E+02±5.86E-01≈
C06	-5.31E+02±8.97E-02	4.88E+02±1.14E+02−	-5.31E+02±1.02E-01≈
C07	0.00E+00±0.00E+00	5.43E-28±2.72E-27≈	1.37E-27±6.86E-27≈
C08	0.00E+00±0.00E+00	3.21E-29±1.21E-28 ≈	5.43E-28±2.72E-27 ≈
C09	1.85E+00±4.90E+00	8.07E+13±1.84E+13−	1.13E+01±2.13E+01 −
C10	3.13E+01±5.73E-06	8.01E+13±2.77E+13−	3.13E+01±3.94E-06≈
C11	-3.92E-04±1.60E-06	-3.92E-04±1.11E-09≈	(92%) −
C12	-1.99E-01±3.09E-07	(88%)−	-1.99E-01±8.41E-08≈
C13	-6.81E+01±6.25E-01	-6.80E+01±7.95E-01≈	-6.83E+01±3.59E-01≈
C14	0.00E+00±0.00E+00	1.60E-01±7.97E-01−	0.00E+00±0.00E+00≈
C15	2.16E+01±2.92E-07	3.59E+14±2.01E+14−	2.18E+01±1.14E+00 −
C16	0.00E+00±0.00E+00	1.10E+00±3.83E-02−	0.00E+00±0.00E+00≈
C17	6.58E-02±1.46E-01	2.04E+03±7.35E+02−	2.30E-01±4.35E-01 −
C18	4.47E-20±2.24E-19	4.39E+04±2.07E+04−	5.60E-18±2.16E-17≈
−	/	**12**	**4**
+	/	**1**	**0**
≈	/	**5**	**14**

in Table X. Besides, the Wilcoxon's rank sum test at a 0.05 significance level was applied to compare C²oDE with each of C²oDE-FR and C²oDE-ECM. If a method obtains the smallest average objective function value on a test function, the corresponding experimental results are highlighted in gray. As shown in Table X, C²oDE outperforms C²oDE-FR and C²oDE-ECM on 12 and four test functions, respectively. In contrast, C²oDE-FR and C²oDE-ECM cannot perform better than C²oDE on more than one test function.

Therefore, the experimental results reveal the contribution of the feasibility rule and the ε constrained method for the first and second phases, respectively.

F. Investigation on How to Select the Best Individual

In the search algorithm of C²oDE, the individual with the least degree of constraint violation is chosen as the "best" individual in the modified DE/rand-to-best/1/bin while the individual with the best objective function value is selected as the "best" individual in DE/current-to-best/1/bin. In this subsection, we empirically investigated how to select the "best" individual. To this end, three C²oDE variants, i.e., C²oDE-Exc, C²oDE-Obj, and C²oDE-Const, were implemented. In C²oDE-Exc, the manners of selecting the "best" individual in the modified DE/rand-to-best/1/bin and DE/current-to-best/1/bin were exchanged. Specifically, the "best" individual in the modified DE/rand-to-best/1/bin was selected in terms of the objective function value while the "best" individual in the DE/current-to-best/1/bin was selected according to the degree of constraint violation. In C²oDE-Obj, both the modified DE/rand-to-best/1/bin and DE/current-to-best/1/bin selected the "best" individual according to the objective function value. On the contrary, both of them selected the "best" individual in terms of the degree of constraint violation in C²oDE-Const. The 18 test functions with 30D from IEEE CEC2010 were adopted for comparison.

The average and standard deviation of objective function values over 25 runs, and the feasible rate are summarized in Table XI. Also, the Wilcoxon's rank sum test at a 0.05

TABLE XI
EXPERIMENTAL RESULTS OF C²oDE, C²oDE-EXC, C²oDE-OBJ, AND C²oDE-CONST OVER 25 INDEPENDENT RUNS ON 18 TEST FUNCTIONS WITH 30D FROM IEEE CEC2010

Instance	C²oDE Mean OFV±Std Dev (feasible rate)	C²oDE-Exc Mean OFV±Std Dev (feasible rate)	C²oDE-Obj Mean OFV±Std Dev (feasible rate)	C²oDE-Const Mean OFV±Std Dev (feasible rate)
C01	-8.20E-01±2.52E-03	-8.20E-01±2.51E-03≈	-8.18E-01±3.65E-03≈	-8.20E-01±2.67E-03≈
C02	-2.22E+00±5.20E-02	-2.11E+00±8.73E-02≈	-2.20E+00±7.06E-02≈	-2.07E+00±1.07E-01−
C03	3.06E+01±2.12E+01	3.05E+01±6.49E+00 ≈	3.67E+01±2.63E+01 ≈	2.87E+01±1.57E-09 ≈
C04	5.46E-06±2.75E-05	2.32E-04±6.13E-04 −	2.89E-05±1.19E-05 −	1.79E-03±1.96E-03 −
C05	-4.82E+02±7.02E-01	-3.77E+02±2.10E+02−	-4.82E+02±5.24E-01 ≈	-2.63E+02±2.60E+02 −
C06	-5.31E+02±8.97E-02	-5.30E+02±2.51E-02≈	-5.31E+02±2.50E-02≈	-5.29E+02±1.23E+00≈
C07	0.00E+00±0.00E+00	2.49E-24±3.75E-24−	0.00E+00±0.00E+00 ≈	25.21E-20±1.85E-19−
C08	0.00E+00±0.00E+00	1.84E-20±5.22E-20 −	5.43E-28±2.72E-27 ≈	2.46E-16±9.08E-16−
C09	1.85E+00±4.90E+00	1.42E+01±2.37E+01−	7.87E+00±1.88E+01−	2.65E+01±2.95E+01−
C10	3.13E+01±5.73E-06	3.13E+01±2.63E-06≈	3.13E+01±3.82E-06≈	3.13E+01±4.70E-06≈
C11	-3.92E-04±1.60E-06	-3.92E-04±1.11E-09≈	(84%) −	-3.92E-04±2.35E-09≈
C12	-1.99E-01±3.09E-07	(80%)−	-1.99E-01±1.81E-08 ≈	-1.99E-01±4.96E-06 ≈
C13	-6.81E+01±6.25E-01	-6.77E+01±5.30E-01≈	-6.82E+01±5.38E-01≈	-6.69E+01±7.63E-01≈
C14	0.00E+00±0.00E+00	9.74E-22±1.65E-21−	0.00E+00±0.00E+00 ≈	8.38E-18±1.96E-17−
C15	2.16E+01±2.92E-07	2.16E+01±1.10E-07≈	2.16E+01±2.79E-07≈	2.16E+01±1.78E-07≈
C16	0.00E+00±0.00E+00	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈
C17	6.58E-02±1.46E-01	1.90E-01±4.62E-01−	4.62E-01±1.69E+00−	(96%) −
C18	4.47E-20±2.24E-19	6.48E-20±2.28E-19≈	6.72E-04±3.35E-03 −	4.99E-05±2.49E-04 −
−	/	**8**	**5**	**9**
+	/	**0**	**0**	**0**
≈	/	**10**	**13**	**9**

TABLE XII
EXPERIMENTAL RESULTS OF C²oDE AND C²oDE-WOR OVER 25 INDEPENDENT RUNS ON THREE TEST FUNCTIONS WITH 10D (C11 WITH 10D, C12 WITH 10D, AND C17 WITH 10D) AND ONE TEST FUNCTION WITH 30D (C12 WITH 30D) FROM IEEE CEC2010

Instance	C²oDE Mean OFV±Std Dev (feasible rate)	C²oDE-WoR Mean OFV±Std Dev (feasible rate)
C11 with 10D	-1.52E-03±4.89E-13	(4%)
C12 with 10D	-7.63E+01±1.22E+02	(0%)
C17 with 10D	1.61E-02±8.04E-02	(76%)
C12 with 30D	-1.99E-01±3.09E-07	(92%)

significance level was used to compare C²oDE with each of C²oDE-Exc, C²oDE-Obj, and C²oDE-Const. The experimental results with the smallest average objective function value among the four compared methods are highlighted in gray on each test function. As shown in Table XI, C²oDE surpasses C²oDE-Exc, C²oDE-Obj, and C²oDE-Const on eight, five, and nine test functions, respectively. However, C²oDE-Exc, C²oDE-Obj, and C²oDE-Const cannot beat C²oDE on any test function.

The above experimental results suggest that the manner of selecting the "best" individual in C²oDE is reasonable.

G. Effectiveness of the Restart Scheme

In order to analyze the effectiveness of the proposed restart scheme, a method called C²oDE-WoR was implemented by removing the restart scheme from C²oDE. The 36 test functions from IEEE CEC2010 were selected for experiments.

The average and standard deviation of objective function values resulting from C²oDE-WoR were computed. The experimental results of those test functions, for which C²oDE and C²oDE-WoR do not have significant performance difference based on the Wilcoxon's rank sum test at a 0.05 significance level, were omitted. As a result, Table XII provides the experimental results for four test functions. In Table XII, the feasible rate is also provided if a method cannot attain feasible solutions consistently.

As shown in Table XII, the restart scheme plays a very important role in the performance of C11 with 10D, C12 with

10D, C17 with 10D, and C12 with 30D. Without the restart scheme, C^{2}oDE-WoR tends to converge to a local optimum in the infeasible region. Especially, for C11 with 10D, C^{2}oDE-WoR can just find feasible solutions in one run, and for C12 with 10D, C^{2}oDE-WoR is unable to find any feasible solution. It is interesting to observe that C^{2}oDE-WoR performs similarly to C^{2}oDE on C11 with 30D and C17 with 30D. This is not difficult to understand because the relatively larger $MaxFEs$ and the population size were specified under this condition.

Therefore, C^{2}oDE gets great benefit from the restart scheme to jump out the infeasible region once the population searches to stall.

Remark 2: We also presented the parameter sensitivity analysis of C^{2}oDE in Section S-I of the supplementary file.

VI. Conclusions

This paper extended an outstanding global optimizer, i.e., CoDE, to tackle COPs. Firstly, the principle of CoDE was inspired to design a search algorithm, which includes three complementary trial vector generation strategies. Among them, one was responsible for diversity and the other two facilitated convergence, thus achieving a tradeoff between diversity and convergence. In order to balance constraints and objective function, one of the two trial vector generation strategies for convergence was guided by the individual with the least degree of constraint violation and the other was guided by the individual with the best objective function value. In addition, a constraint-handling technique consisting of the feasibility rule and the ε constrained method was developed. The constraint-handling technique was coupled with the search algorithm in a natural way. Furthermore, a restart scheme was designed to deal with complex constraints. By the above procedure, a new constrained DE, i.e., C^{2}oDE, was proposed. Systematic experiments on two benchmark test suites demonstrated that:

1) C^{2}oDE showed better or at least competitive performance against other state-of-the-art constrained EAs.
2) C^{2}oDE had a great advantage over the original CoDE for solving COPs.
3) The restart scheme was able to enhance C^{2}oDE's ability to reach feasible solutions on some extremely difficult COPs.

In the future, it is interesting to generalize C^{2}oDE for solving constrained multiobjective optimization problems (CMOPs). When solving a CMOP, a set of solutions, which is uniformly distributed on the feasible Pareto front, is desired. Thus, diversity is a critical factor which affects the performance of an algorithm for CMOPs. C^{2}oDE already contains a trial vector generation strategy for diversity, i.e., DE/current-to-rand/1. In order to further enhance the diversity for solving CMOPs, C^{2}oDE can be improved from the following two aspects: 1) since C^{2}oDE is an open framework, it is easy to add more trial vector generation strategies for diversity to C^{2}oDE, such as DE/rand/2/bin; and 2) the polynomial mutation [29], [77] and the improved BGA mutation [9], [78], which have been proven to be effective for promoting the diversity of population, can be incorporated into C^{2}oDE.

The Matlab source code of C^{2}oDE can be downloaded from Y. Wang's homepage: http://www.escience.cn/people/yongwang1/index.html

References

[1] Y. Miao, G. M. Fadel, and V. B. Gantovnik, "Vehicle configuration design with a packing genetic algorithm," *International Journal of Heavy Vehicle Systems*, vol. 15, no. 2-4, pp. 433–448, 2008.

[2] G. Onwubolu and D. Davendra, "Scheduling flow shops using differential evolution algorithm," *European Journal of Operational Research*, vol. 171, no. 2, pp. 674–692, 2006.

[3] Y.-H. Jia, W.-N. Chen, T. Gu, H. Zhang, H. Yuan, Y. Lin, W.-J. Yu, and J. Zhang, "A dynamic logistic dispatching system with set-based particle swarm optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2017, in press.

[4] C. K. Hen, "Design and development of automated digital circuit structure base on evolutionary algorithm method," *International Journal of Electronics, Computer and Communications Technologies*, vol. 2, no. 1, pp. 1–8, 2011.

[5] D. Li, C. Zhang, G. Tian, X. Shao, and Z. Li, "Multiobjective program and hybrid imperialist competitive algorithm for the mixed-model two-sided assembly lines subject to multiple constraints," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 48, no. 1, pp. 119–129, Jan 2018.

[6] D. S. Linden and E. E. Altshuler, "Automating wire antenna design using genetic algorithms," *Microwave Journal*, vol. 39, no. 3, pp. 74–81, 1996.

[7] T. Bäck, D. Fogel, and Z. Michalewicz, "Handbook of evolutionary computation," *Release*, vol. 97, no. 1, p. B1, 1997.

[8] 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.

[9] Y. Wang and Z. Cai, "Constrained evolutionary optimization by means of (μ+λ)-differential evolution and improved adaptive trade-off model," *Evolutionary Computation*, vol. 19, no. 2, pp. 249–285, 2011.

[10] B. Tessema and G. G. Yen, "An adaptive penalty formulation for constrained evolutionary optimization," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 39, no. 3, pp. 565–578, 2009.

[11] 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.

[12] 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.

[13] W.-Y. Chiu, "Multiobjective controller design by solving a multiobjective matrix inequality problem," *IET Control Theory & Applications*, vol. 8, no. 16, pp. 1656–1665, 2014.

[14] H. G. Harno and I. R. Petersen, "Synthesis of linear coherent quantum control systems using a differential evolution algorithm," *IEEE Transactions on Automatic Control*, vol. 60, no. 3, pp. 799–805, 2015.

[15] W.-Y. Chiu, "Pareto optimal controller designs in differential games," in *2014 CACS International Automatic Control Conference (CACS)*. IEEE, 2014, pp. 179–184.

[16] J. Brest, B. Boškovič, and V. Žumer, "An improved self-adaptive differential evolution algorithm in single objective constrained real-parameter optimization," in *2010 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2010, pp. 1–8.

[17] M. Dong, N. Wang, X. Cheng, and C. Jiang, "Composite differential evolution with modified oracle penalty method for constrained optimization problems," *Mathematical Problems in Engineering*, vol. 2014, 2014.

[18] W. Wei, J. Wang, and M. Tao, "Constrained differential evolution with multiobjective sorting mutation operators for constrained optimization," *Applied Soft Computing*, vol. 33, pp. 207–222, 2015.

[19] K. Li, L. Zuo, W. Li, and L. Yang, "A novel differential evolution algorithm based on JADE for constrained optimization," in *International Symposium on Intelligence Computation and Applications*. Springer, 2015, pp. 84–94.

[20] Y. Wang, Z. Cai, and Q. Zhang, "Differential evolution with composite trial vector generation strategies and control parameters," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 55–66, 2011.

[21] J. Liang, T. P. Runarsson, E. Mezura-Montes, M. Clerc, P. Suganthan, C. A. Coello 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, 2006.

[22] R. Mallipeddi and P. N. Suganthan, "Problem definitions and evaluation criteria for the cec 2010 competition on constrained real-parameter optimization," *Nanyang Technological University, Singapore*, 2010.

[23] S. Das, S. S. Mullick, and P. Suganthan, "Recent advances in differential evolution–an updated survey," *Swarm and Evolutionary Computation*, vol. 27, pp. 1–30, 2016.

[24] A. Bhattacharya and P. K. Chattopadhyay, "Solving economic emission load dispatch problems using hybrid differential evolution," *Applied Soft Computing*, vol. 11, no. 2, pp. 2526–2537, 2011.

[25] Y. Wang, H. Liu, H. Long, Z. Zhang, and S. Yang, "Differential evolution with a new encoding mechanism for optimizing wind farm layout," *IEEE Transactions on Industrial Informatics*, vol. PP, no. 99, pp. 1–1, 2017.

[26] A. Ponsich and C. A. Coello Coello, "A hybrid differential evolution—tabu search algorithm for the solution of job-shop scheduling problems," *Applied Soft Computing*, vol. 13, no. 1, pp. 462–474, 2013.

[27] Q.-K. Pan, L. Wang, and B. Qian, "A novel differential evolution algorithm for bi-criteria no-wait flow shop scheduling problems," *Computers & Operations Research*, vol. 36, no. 8, pp. 2498–2511, 2009.

[28] Y. Wang, B. Xu, G. Sun, and S. Yang, "A two-phase differential evolution for uniform designs in constrained experimental domains," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 5, pp. 665–680, Oct 2017.

[29] K. Deb, "An efficient constraint handling method for genetic algorithms," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, no. 2, pp. 311–338, 2000.

[30] T. Takahama and S. Sakai, "Constrained optimization by the ε constrained differential evolution with an archive and gradient-based mutation," in *IEEE Congress on Evolutionary Computation*. IEEE, 2010, pp. 1–9.

[31] ——, "Efficient constrained optimization by the ε constrained rank-based differential evolution," in *2012 IEEE Congress on Evolutionary Computation*. IEEE, 2012, pp. 1–8.

[32] E. Mezura-Montes and C. A. Coello Coello, "Constraint-handling in nature-inspired numerical optimization: past, present and future," *Swarm and Evolutionary Computation*, vol. 1, no. 4, pp. 173–194, 2011.

[33] V. V. De Melo and G. L. C. Carosio, "Evaluating differential evolution with penalty function to solve constrained engineering problems," *Expert Systems with Applications*, vol. 39, no. 9, pp. 7860–7863, 2012.

[34] J. Zhang and A. C. Sanderson, "JADE: adaptive differential evolution with optional external archive," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958, 2009.

[35] 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.

[36] W.-F. Gao, G. G. Yen, and S.-Y. Liu, "A dual-population differential evolution with coevolution for constrained optimization," *IEEE Transactions on Cybernetics*, vol. 45, no. 5, pp. 1108–1121, 2015.

[37] T. Takahama and S. Sakai, "Efficient constrained optimization by the ε constrained differential evolution with rough approximation using kernel regression," in *2013 IEEE Congress on Evolutionary Computation*. IEEE, 2013, pp. 1334–1341.

[38] W. Yi, X. Li, L. Gao, Y. Zhou, and J. Huang, "ε constrained differential evolution with pre-estimated comparison using gradient-based approximation for constrained optimization problems," *Expert Systems With Applications*, vol. 44, pp. 37–49, 2016.

[39] Y. Wang and Z. Cai, "A dynamic hybrid framework for constrained evolutionary optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 1, pp. 203–217, 2012.

[40] ——, "Combining multiobjective optimization with differential evolution to solve constrained optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 1, pp. 117–134, 2012.

[41] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[42] N. M. Hamza, R. A. Sarker, and D. L. Essam, "Differential evolution with multi-constraint consensus methods for constrained optimization," *Journal of Global Optimization*, vol. 57, no. 2, pp. 583–611, 2013.

[43] A. Nedic, A. Ozdaglar, and P. A. Parrilo, "Constrained consensus and optimization in multi-agent networks," *IEEE Transactions on Automatic Control*, vol. 55, no. 4, pp. 922–938, 2010.

[44] N. M. Hamza, D. L. Essam, and R. A. Sarker, "Constraint consensus mutation-based differential evolution for constrained optimization," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 3, pp. 447–459, 2016.

[45] C. Cui, X. Yang, and T. Gao, "A self-adaptive interior penalty based differential evolution algorithm for constrained optimization," in *International Conference in Swarm Intelligence*. Springer, 2014, pp. 309–318.

[46] Q. Fan and X. Yan, "Differential evolution algorithm with co-evolution of control parameters and penalty factors for constrained optimization problems," *Asia-Pacific Journal of Chemical Engineering*, vol. 7, no. 2, pp. 227–235, 2012.

[47] E. Harth and E. Tzanakou, "Alopex: a stochastic method for determining visual receptive fields," *Vision Research*, vol. 14, no. 12, pp. 1475–1482, 1974.

[48] C. Saha, S. Das, K. Pal, and S. Mukherjee, "A fuzzy rule-based penalty function approach for constrained evolutionary optimization," *IEEE Transactions on Cybernetics*, vol. 46, no. 12, pp. 2953–2965, 2016.

[49] X. Li and G. Zhang, "Minimum penalty for constrained evolutionary optimization," *Computational Optimization and Applications*, vol. 60, no. 2, pp. 513–544, 2015.

[50] W. Long, X. Liang, Y. Huang, and Y. Chen, "A hybrid differential evolution augmented lagrangian method for constrained numerical and engineering optimization," *Computer-Aided Design*, vol. 45, no. 12, pp. 1562–1574, 2013.

[51] V. V. De Melo and G. L. Carosio, "Investigating multi-view differential evolution for solving constrained engineering design problems," *Expert Systems with Applications*, vol. 40, no. 9, pp. 3370–3377, 2013.

[52] W. Xu, R. Wang, L. Zhang, and X. Gu, "A multi-population cultural algorithm with adaptive diversity preservation and its application in ammonia synthesis process," *Neural Computing and Applications*, vol. 21, no. 6, pp. 1129–1140, 2012.

[53] M. Asafuddoula, T. Ray, and R. Sarker, "An adaptive hybrid differential evolution algorithm for single objective optimization," *Applied Mathematics and Computation*, vol. 231, pp. 601–618, 2014.

[54] B. Ghasemishabankareh, X. Li, and M. Ozlen, "Cooperative coevolutionary differential evolution with improved augmented lagrangian to solve constrained optimisation problems," *Information Sciences*, vol. 369, pp. 441–456, 2016.

[55] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398–417, 2009.

[56] S. M. Elsayed, R. A. Sarker, and D. L. Essam, "A self-adaptive combined strategies algorithm for constrained optimization using differential evolution," *Applied Mathematics and Computation*, vol. 241, pp. 267–282, 2014.

[57] N. Dong and Y. Wang, "A memetic differential evolution algorithm based on dynamic preference for constrained optimization problems," *Journal of Applied Mathematics*, vol. 2014, 2014.

[58] S. Tsutsui, M. Yamamura, and T. Higuchi, "Multi-parent recombination with simplex crossover in real coded genetic algorithms," in *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation-Volume 1*. Morgan Kaufmann Publishers Inc., 1999, pp. 657–664.

[59] Y. Lin, "Mixed-integer constrained optimization based on memetic algorithm," *Journal of Applied Research and Technology*, vol. 11, no. 2, pp. 242–250, 2013.

[60] J. A. Nelder and R. Mead, "A simplex method for function minimization," *The Computer Journal*, vol. 7, no. 4, pp. 308–313, 1965.

[61] W. Zhao, L. Wang, Y. Yin, B. Wang, Y. Wei, and Y. Yin, "An improved backtracking search algorithm for constrained optimization problems," in *International Conference on Knowledge Science, Engineering and Management*. Springer, 2014, pp. 222–233.

[62] P. Civicioglu, "Backtracking search optimization algorithm for numerical optimization problems," *Applied Mathematics and Computation*, vol. 219, no. 15, pp. 8121–8144, 2013.

[63] H. Mühlenbein and D. Schlierkamp-Voosen, "Predictive models for the breeder genetic algorithm I. continuous parameter optimization," *Evolutionary Computation*, vol. 1, no. 1, pp. 25–49, 1993.

[64] R. P. Parouha and K. N. Das, "Parallel hybridization of differential evolution and particle swarm optimization for constrained optimization with its application," *International Journal of System Assurance Engineering and Management*, pp. 1–20.

[65] K. Yu, X. Wang, and Z. Wang, "An improved teaching-learning-based optimization algorithm for numerical and engineering optimization problems," *Journal of Intelligent Manufacturing*, pp. 1–13, 2014.

[66] D.-H. Tran, M.-Y. Cheng, and M.-T. Cao, "Solving resource-constrained project scheduling problems using hybrid artificial bee colony with

differential evolution," *Journal of Computing in Civil Engineering*, p. 04015065, 2015.

[67] A. W. Iorio and X. Li, "Solving rotated multi-objective optimization problems using differential evolution," in *Australasian Joint Conference on Artificial Intelligence*. Springer, 2004, pp. 861–872.

[68] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.

[69] R. Mallipeddi and P. N. Suganthan, "Ensemble of constraint handling techniques," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 4, pp. 561–579, 2010.

[70] B. Y. Qu and P. N. Suganthan, "Constrained multi-objective optimization algorithm with an ensemble of constraint handling methods," *Engineering Optimization*, vol. 43, no. 4, pp. 403–416, 2011.

[71] A. W. Mohamed, "A novel differential evolution algorithm for solving constrained engineering optimization problems," *Journal of Intelligent Manufacturing*, 2017, in press. https://doi.org/10.1007/s10845-017-1294-6.

[72] C. Peng, H.-L. Liu, and F. Gu, "A novel constraint-handling technique based on dynamic weights for constrained optimization problems," *Soft Computing*, 2017, in press. https://doi.org/10.1007/s00500-017-2603-x.

[73] R. Mallipeddi and P. N. Suganthan, "Differential evolution with ensemble of constraint handling techniques for solving cec 2010 benchmark problems," in *2010 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2010, pp. 1–8.

[74] W. Zhang, G. G. Yen, and Z. He, "Constrained optimization via artificial immune system," *IEEE Transactions on Cybernetics*, vol. 44, no. 2, pp. 185–198, 2014.

[75] J. J. Liang, S. Zhigang, and L. Zhihui, "Coevolutionary comprehensive learning particle swarm optimizer," in *2010 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2010, pp. 1–8.

[76] J. Alcalá-Fdez, L. Sanchez, S. Garcia, M. J. del Jesus, S. Ventura, J. M. Garrell, J. Otero, C. Romero, J. Bacardit, V. M. Rivas *et al.*, "Keel: a software tool to assess evolutionary algorithms for data mining problems," *Soft Computing*, vol. 13, no. 3, pp. 307–318, 2009.

[77] H. Li and Q. Zhang, "Multiobjective optimization problems with complicated pareto sets, MOEA/D and NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 284–302, April 2009.

[78] Y. Wang, Z. Cai, G. Guo, and Y. Zhou, "Multiobjective optimization and hybrid evolutionary algorithm to solve constrained optimization problems," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 37, no. 3, pp. 560–575, June 2007.

Bing-Chuan Wang received the B.S. degree in automation and M.S. degree in control science and engineering both from Central South University, Changsha, China, in 2013 and 2016, respectively.

He is currently pursuing the Ph.D. degree at City University of Hong Kong, Hong Kong. His current research interests include evolutionary computation, modeling of distributed parameter systems, and battery management system.

Han-Xiong Li (S'94–M'97–SM'00–F'11) received the B.E. degree in aerospace engineering from the National University of Defense Technology, China, in 1982, the M.E. degree in electrical engineering from Delft University of Technology, The Netherlands, in 1991, and the Ph.D. degree in electrical engineering from the University of Auckland, New Zealand, in 1997.

He is a professor in the Department of Systems Engineering and Engineering Management, City University of Hong Kong. He has a broad experience in both academia and industry. He has authored 2 books and 7 patents, and published more than 200 SCI journal papers with h-index 37 (web of science). His current research interests include process modeling and control, system intelligence, distributed parameter systems, and battery management system.

Dr. Li serves as Associate Editor for *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *IEEE Transactions on Cybernetics* (2002-2016) and *IEEE Transactions on Industrial Electronics* (2009-2015). He was awarded the Distinguished Young Scholar (overseas) by the China National Science Foundation in 2004, a Chang Jiang professorship by the Ministry of Education, China in 2006, and a national professorship in China Thousand Talents Program in 2010. He serves as a distinguished expert for Hunan Government and China Federation of Returned Overseas Chinese. He is a Fellow of the IEEE.

Jia-Peng Li received the B.S. degree in intelligent science and technology from Central South University, Changsha, China, in 2015, where he is currently pursuing the M.S. degree in control science and engineering.

His current research interests include evolutionary constrained single- and multi-objective optimization, and machine learning.

Yong Wang (M'08–SM'17) received the B.S. degree in automation from the Wuhan Institute of Technology, Wuhan, China, in 2003, and the M.S. degree in pattern recognition and intelligent systems and the Ph.D. degree in control science and engineering both from the Central South University (CSU), Changsha, China, in 2006 and 2011, respectively.

He is currently an Associate Professor with the School of Information Science and Engineering, CSU. His current research interests include the theory, algorithm design, and interdisciplinary applications of computational intelligence.

Dr. Wang was awarded the Hong Kong Scholar by the Mainland-Hong Kong Joint Postdoctoral Fellows Program, China, in 2013, the Excellent Doctoral Dissertation by Hunan Province, China, in 2013, the New Century Excellent Talents in University by the Ministry of Education, China, in 2013, the 2015 IEEE Computational Intelligence Society Outstanding PhD Dissertation Award, the Hunan Provincial Natural Science Fund for Distinguished Young Scholars, in 2016, the EU Horizon 2020 Marie Sklodowska-Curie Fellowship, in 2016, and a Highly Cited Researcher in computer science by Clarivate Analytics, in 2017. He is currently serving as an associate editor for the *Swarm and Evolutionary Computation*.

Supplementary File for "Composite Differential Evolution for Constrained Evolutionary Optimization"

Bing-Chuan Wang, Han-Xiong Li, *Fellow, IEEE*, Jia-Peng Li, and Yong Wang, *Senior Member, IEEE*

B.-C. Wang is with the Department of Systems Engineering and Engineering Management, City University of Hong Kong, Hong Kong. (e-mail: bingcwang3-c@my.cityu.edu.hk).

H.-X. Li is with the Department of Systems Engineering and Engineering Management, City University of Hong Kong, Hong Kong, and also with the State Key Laboratory of High Performance Complex Manufacturing, Central South University, Changsha 410083, China. (e-mail: mehxli@cityu.edu.hk).

J.-P. Li is with the School of Information Science and Engineering, Central South University, Changsha 410083, China. (e-mail: ljpcsu@csu.edu.cn).

Y. Wang is with the School of Information Science and Engineering, Central South University, Changsha 410083, China, and also with the School of Computer Science and Electronic Engineering, University of Essex, Colchester CO4 3SQ, UK. (Email: ywang@csu.edu.cn).

TABLE S-I
EXPERIMENTAL RESULTS OF C^{2}oDE WITH SEVEN VARYING p OVER 25 INDEPENDENT RUNS ON 18 TEST FUNCTIONS WITH 30D FROM IEEE CEC2010

IEEE CEC2010 with 30D	$p = 0.0$ Mean OFV±Std Dev (feasible rate)	$p = 0.2$ Mean OFV±Std Dev (feasible rate)	$p = 0.4$ Mean OFV±Std Dev (feasible rate)	$p = 0.6$ Mean OFV±Std Dev (feasible rate)	$p = 0.8$ Mean OFV±Std Dev (feasible rate)	$p = 1.0$ Mean OFV±Std Dev (feasible rate)	$p = 0.5$ (C^{2}oDE) Mean OFV±Std Dev (feasible rate)
C01	-8.16E-01±6.59E-03≈	-8.18E-01±4.20E-03≈	-8.19E-01±3.50E-03≈	-8.21E-01±2.65E-03≈	-8.12E-01±4.45E-02−	-6.28E-01±2.32E-02−	-8.20E-01±2.52E-03
C02	2.70E+00±9.16E-01−	-1.51E+00±3.94E-01−	-2.16E+00±1.30E-01≈	-2.24E+00±5.20E-02≈	-2.24E+00±3.33E-02≈	(40%)−	-2.22E+00±5.20E-02
C03	1.74E+13±5.10E+13 −	2.87E+01±2.52E-09≈	2.87E+01±3.18E-08≈	3.85E+01±3.33E+01≈	4.32E+01±5.10E+01≈	(8%) −	3.06E+01±2.12E+01
C04	-3.28E-06±1.57E-07 +	-2.82E-06±1.42E-06+	-3.11E-06±4.52E-07+	2.77E-05±1.16E-04 −	4.77E-04±2.76E-04 −	(0%)−	5.46E-06±2.75E-05
C05	4.49E+02±1.18E+02−	1.39E+02±2.89E+02−	-4.41E+02±8.62E+01−	-4.82E+02±6.24E-01≈	-4.83E+02±3.75E-01≈	(44%)−	-4.82E+02±7.02E-01
C06	4.88E+02±1.14E+02−	-3.62E+02±2.18E+02−	-5.30E+02±1.33E+00≈	-5.31E+02±1.20E-02≈	-5.31E+02±1.47E-02≈	(0%)−	-5.31E+02±8.97E-02
C07	5.43E-28±2.72E-27≈	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	5.43E-28±2.72E-27≈	0.00E+00±0.00E+00≈	5.43E-28±2.72E-27≈	0.00E+00±0.00E+00
C08	3.21E-29±1.21E-28 ≈	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	5.63E-28±2.81E-27≈	0.00E+00±0.00E+00≈	7.79E+01±3.42E+02−	0.00E+00±0.00E+00
C09	8.07E+13±1.84E+13−	1.26E+13±2.46E+13−	1.51E+01±2.32E+01−	3.33E+00±1.39E+01≈	6.72E+00±1.96E+01≈	3.00E+00±1.41E+01≈	1.85E+00±4.90E+00
C10	8.01E+13±2.77E+13−	8.01E+13±1.73E+13−	3.13E+01±1.26E-05≈	3.13E+01±9.34E-06≈	3.13E+01±4.17E-05≈	3.13E+01±4.28E-02≈	3.13E+01±5.73E-06
C11	-3.92E-04±1.11E-09≈	-3.92E-04±8.82E-10≈	-3.92E-04±9.26E-10≈	-3.92E-04±1.73E-10≈	(0%) −	(0%) −	-3.92E-04±1.60E-06
C12	(88%)−	(76%)−	(88%)−	(80%)−	(92%) −	(0%) −	-1.99E-01±3.09E-07
C13	-6.80E+01±7.95E-01≈	-6.81E+01±8.66E-01≈	-6.84E+01±2.91E-01≈	-6.83E+01±4.10E-01≈	-6.80E+01±7.08E-01≈	(0%)−	-6.81E+01±6.25E-01
C14	1.60E-01±7.97E-01−	3.19E-01±1.10E+00−	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00
C15	3.59E+14±2.01E+14−	5.86E+10±1.15E+11−	2.16E+01±4.44E-07≈	2.16E+01±2.10E-07≈	2.16E+01±3.16E-07≈	2.16E+01±5.33E-07≈	2.16E+01±2.92E-07
C16	1.10E+00±3.83E-02−	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00
C17	2.04E+03±7.35E+02−	1.14E-01±2.95E-01−	1.38E-01±1.97E-01−	7.80E-02±1.49E-011≈	5.26E-02±2.04E-01≈	(96%)−	6.58E-02±1.46E-01
C18	4.39E+04±2.07E+04−	2.56E-03±7.97E-03−	2.54E-29±7.97E-28≈	4.32E-27±1.35E-26≈	1.146E-24±4.00E-24≈	1.14E-18±5.70E-18≈	4.47E-20±2.24E-19
−	**12**	**10**	**4**	**2**	**4**	**11**	/
+	**1**	**1**	**1**	**0**	**0**	**0**	/
≈	**5**	**7**	**13**	**16**	**14**	**7**	/

S-I. PARAMETER SENSITIVITY ANALYSIS

The sensitivity of the parameter p of the ε constrained method was investigated in this subsection. As introduced in Section II-E, p controls the extent that the information of objective function is utilized. Too much information of objective function will cause slow convergence speed toward the feasible region while the search with too little information of objective function may run the high risk of getting stuck in a local optimum. Hence, this parameter is vital to the tradeoff between constraints and objective function.

We ran C^{2}oDE with seven different values of p, i.e., p=0.0, p=0.2, p=0.4, p=0.6, p=0.8, p=1.0, and p=0.5 over 25 independent runs on the 18 test functions with 30D from IEEE CEC2010. It is noteworthy that in the original C^{2}oDE, p was equal to 0.5. The Wilcoxon's rank sum test at a 0.05 significance level was utilized to compare p=0.5 with each of p=0.0, p=0.2, p=0.4, p=0.6, p=0.8, and p=1.0. The average and standard deviation of objective function values are summarized in Table S-I. Similarly, the feasible rate is given in the case that a method cannot achieve 100% feasible rate for a test function. Besides, when a method obtains the smallest average objective function value on a test function, the corresponding experimental results are highlighted in gray.

As shown in the Table S-I, p=0.5 outperforms p=0.0, p=0.2, p=0.4, p=0.6, p=0.8, and p=1.0 on 12, 10, four, two, four, and 11 test functions, respectively. On the contrary, the six competitors cannot perform better than p=0.5 on more than one test function. Moreover, they suffer from infeasible convergence in the infeasible region for different number of test functions. Therefore, p=0.5 is recommended in this paper.