This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

A Dual-Population Differential Evolution with Coevolution for Constrained Optimization

Wei-feng Gao, Gary G. Yen, *Fellow, IEEE,* and San-yang Liu

***Abstract*—Inspired by the fact that in modern society, team cooperation and the division of labor play important roles in accomplishing a task, this paper proposes a dual-population differential evolution (DPDE) with coevolution for constrained optimization problems (COPs). The COP is treated as a bi-objective optimization problem where the first objective is the actual cost or reward function to be optimized, while the second objective accounts for the degree of constraint violations. At each generation during the evolution process, the whole population is divided into two based on the solution's feasibility to treat the both objectives separately. Each subpopulation focuses on only optimizing the corresponding objective which leads to a clear division of work. Furthermore, DPDE makes use of an information-sharing strategy to exchange search information between the different subpopulations similar to the team cooperation. The comparison of the proposed method on a number of benchmark functions with selected state-of-the-art constraint-handling algorithms indicates that the proposed technique performs competitively and effectively.**

***Index Terms*—Coevolutionary technique, constrained optimization, differential evolution, dual-population.**

I. Introduction

Many real-world optimization problems involve various kinds of constraints. The problems are called constrained optimization problems (COPs) [1]–[3]. In general, a COP can be expressed by the following equations:

$$\text{Minimize } f(X) \tag{1}$$

subject to q inequality constraints and $m-q$ equality constraints as follows:

$$\begin{cases} g_i(X) \le 0, i = 1, \ldots, q \\ h_i(X) = 0, i = q+1, \ldots, m \end{cases}$$

where $f(X)$ is the objective function; $X = (x_1, x_2, \ldots, x_n)$ denotes a n-dimensional vector of decision variables, each being defined by lower and upper bounds $L = (l_1, l_2, \ldots, l_n)$, $U = (u_1, u_2, \ldots, u_n)$, respectively. Any solution X which satisfies all the inequality constraints $g_i(X)$ and the equality constraints $h_i(X)$ is called a feasible solution; otherwise, X is called an infeasible solution. If an inequality constraint satisfies $g_i(X) = 0$, it is called active constraint. Hence, all equality constraints $h_i(X) (i = q+1, \ldots, m)$ are considered as active constraints.

When handling COPs, the equality constraints are usually transformed into the inequality form

$$|h_i(X)| - \delta \le 0, \qquad i = q+1, \ldots, m \tag{2}$$

where δ is a positive tolerance parameter for the equality constraints. Thus, the degree of constraint violation of a solution X on the ith constraint can be calculated by the following equation:

$$G_i(X) = \begin{cases} \max\{0, g_i(X)\}, i = 1, \ldots, q \\ \max\{0, |h_i(X)| - \delta\}, i = q+1, \ldots, m. \end{cases} \tag{3}$$

Then, $G(X) = \sum_{i=1}^{m} G_i(X)$ refers to the total degree of constraint violations of the solution X.

The main objective of constrained optimization algorithms is to locate the global optimal solution in the feasible region. One of the main challenges for handling COPs is how to maintain a delicate balance between feasible and infeasible solutions during the search process [4]. One way to deal with infeasible solutions is to ignore them completely and keep on creating new solutions until a population with only feasible ones is made available [6]. However, the deficiency of this method is that it does not provide an effective way to exploit the useful information hidden in the infeasible solutions. As a result, a variety of frameworks have been proposed to exploit the information hidden in infeasible solutions [6]. Please note in some exceptional cases, the information hidden in an infeasible solution is not available when a physical experiment violating some constraints cannot be performed [5]. Additionally, if the search space is discontinuous or consists of multiple separating feasible regions, the algorithm may be restricted to search only in one of the regions and may only locate a local optimal solution.

Since the 1970s [7], solving COPs has become an important research area of evolutionary algorithms (EAs). EAs are population-based search heuristics that draw the inspiration from the metaphor of natural selection and survival of the fittest in biology. Due to its simplicity and ease of implementation, EAs have attracted a growing interest and have been widely applied to solve COPs, which results in a large

Manuscript received December 22, 2013; revised May 23, 2014; accepted July 21, 2014. This work was supported in part by the National Nature Science Foundation of China under Grant 61373174 and Grant 11326188 and in part by the Fundamental Research Funds for the Central Universities under Grant 14CX02160A. This paper was recommended by Associate Editor Q. Zhao.

W. Gao is with the School of Science, China University of Petroleum, Qingdao 266580, China (e-mail: gaoweifeng2004@126.com).

G. G. Yen is with the School of Electrical and Computer Engineering, Oklahoma State University, Stillwater, MN 74078 USA (e-mail: gyen@okstate.edu).

S. Liu is with the School of Mathematics and Statistics, Xidian University, Xi'an 710071, China (e-mail: liusanyang@126.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2014.2345478

number of constrained optimization evolutionary algorithms (COEAs) [1]–[4]. It is necessary to point out that the original EAs are usually designed for unconstrained optimization problems in nature that need employ specific mechanisms to handle constraints when solving COPs. Meanwhile, different constraint-handling techniques combined with EAs have been developed. Three most frequently used constraint-handling techniques in COEAs include penalty functions, feasibility selection rule, and multiobjective optimization.

Penalty functions [6]–[9] are the most commonly used approaches for solving COPs using EAs. In these methods, the penalty factor is applied to penalize each infeasible solution that violates the constraints to drive infeasible solution toward feasible region. Due to their simplicity, static penalty functions are very popular. However, they usually require definition of the penalty factors which are problem-dependent parameters and often chosen heuristically, and thus form a major difficulty for the use of these methods. In order to overcome this concerning issue, the adaptive penalty functions have been proposed. In these methods, the information collocated in the search process is used to tune the penalty factors imposed upon infeasible solutions. Adaptive penalty methods are easy to implement and do not require users to define parameter values explicitly. Methods based on feasibility selection rules also do not require a careful fine-tuning of parameters [4], [10]. In these algorithms, feasible solutions are always considered preferred to infeasible solutions to a degree. Therefore, when population fitness assignment is implemented, feasible solutions will rank first followed by infeasible solutions with small degree of constraint violation.

Additionally, multiobjective optimization techniques have also been introduced to deal with COPs. The main idea of this kind of methods is that COPs are transformed into unconstrained multiobjective optimization problems, and multiobjective optimization techniques are employed to handle the converted problems. This kind of technique has attracted considerable interest in the community of constrained evolutionary optimization in the past several decades and numerous approaches have been developed. In these algorithms [55]–[57], Pareto dominance is applied to rank the individuals and assign fitness accordingly. However, as Pareto dominance provides only a partial-order relation, it is difficult to select individuals for the next generations to continue the evolution process. The resulted domination rank techniques often exert considerable computational time.

Unlike previous methods, inspired by the concept that in modern society, the team cooperation and the division of labor play important roles in accomplishing a task, a mechanism termed dual-population for COPs is proposed in this paper. The motivations lie in that a COP is treated as a bi-objective optimization problem, where the first objective is the actual objective, while the second objective is the degree of constraint violations. Since it is difficult to consider the two objectives (i.e., objective function and degree of constraint violation) as a whole in one population due to distinct nature of quality measures, we treat both objectives separately in different subpopulation. One subpopulation consists of the infeasible solutions to minimize the degree of constraint violations, while the other subpopulation consists of the feasible solutions to optimize the objective function value. Both subpopulations cooperate to approximate the feasible optimal solution. The size of either subpopulation varies dynamically based on the number of feasible solutions in the current population. In every generation, the solutions in either subpopulation calculate the two objective functions like that in traditional COEAs. However, when executing evolutionary operators like selection, the fitness value of a solution in each subpopulation is assigned by the corresponding objective function of the bi-objective optimization problem. This way, the solutions are guided by the corresponding objective to search different regions. Compared to multiobjective optimization techniques for COPs, the computational complexity is reduced without nondominated sorting. However, as one subpopulation focuses on optimizing the actual objective alone while the other subpopulation is concerned only on minimizing the degree of constraint violations, it may lead to a problem that the dual subpopulations work independently and are lack of coordination and communication between them. This will easily result in inefficient and ineffective approximation of the feasible optimal solution. In order to address this problem, an information-sharing strategy is proposed so that different subpopulations can share their search information and communicate with each other. Here, we call the method that uses the dual-population mechanism and the information-sharing strategy to deal with COPs as DPCO technique.

It should be noted that the dual-population approach is a type of multipopulation approaches and has been actively researched over years in combination with a GA [18]–[20] or a DE [21]. The DPCO can be considered to be a coevolutionary mechanism since it uses dual subpopulations to cooperatively deal with COPs. Coevolution has been regarded as a meaningful evolutionary mechanism in the biological world that has induced multiple interesting adaptations and made great contributions to biological diversity [33]. In the past two decades, the coevolutionary mechanism has also been successfully employed by researchers in the EAs community [13]. The main idea of this mechanism is that the complex problem is decomposed into subproblems and these different subproblems are optimized by multiple subpopulations cooperatively. In DPCO, the COP is transformed into the bi-objective optimization problem and these two objectives are considered separately. Thus, the problem need not be decomposed. Dual subpopulations optimize two different objectives and cooperatively approximate the feasible optimal solution of COPs. Therefore, DPCO can be considered as a coevolutionary technique for solving COPs.

As a general technique, it is straightforward to perform the DPCO technique which can be combined with any existing single-objective optimization algorithm in each subpopulation. In this paper, by considering differential evolution (DE) [34] which is a simple and powerful EA with ease to use, and fast convergence speed, we adopt DE for both subpopulations and design dual-population DE with coevolutionary, namely, DPDE, as an instantiation of DPCO to solve COPs.

The remainder of this paper is organized as follows. Section II discusses the DE and reviews related works. In

Section III, the proposed algorithm is presented in sufficient details. The problem definition and experimental results are presented and discussed in Section IV. Finally, the paper is concluded in Section V.

II. Basic Differential Evolution and Related Studies

A. Differential Evolution

The original DE proposed by Storn and Price [34] is a population-based optimization algorithm. After initialization, it implements a loop of evolutionary operations, namely, mutation, crossover, and selection operations to update the population. The one classic DE variant, called DE/rand/1/bin, is used in this paper.

The population of DE consists of *SN* solutions with n-dimensional parameter vectors

$$X_i = \{x_{i,1}, x_{i,2}, \ldots, x_{i,n}\}, \qquad i = 1, 2, \ldots, SN. \tag{4}$$

With respect to each solution X_i (called a target vector) in the current population, the mutation strategy is employed to produce a mutant vector $V_i = \{v_{i,1}, v_{i,2}, \ldots, v_{i,n}\}$ by adding the scaled difference of two randomly selected and distinct population members to a third solution

$$V_i = X_{r1} + F \cdot (X_{r2} - X_{r3}), \qquad i = 1, 2, \ldots, SN \tag{5}$$

where F is the mutation scaling factor, $r1$, $r2$, and $r3$ are integers randomly chosen from $\{1, 2, \ldots, SN\}$, which also satisfy $r1 \neq r2 \neq r3 \neq i$.

After the mutation operation, a binomial crossover operation is applied to the target vector X_i and the mutant vector V_i to generate a trial vector $U_i = \{u_{i,1}, u_{i,2}, \ldots, u_{i,n}\}$ as follows:

$$u_{i,j} = \begin{cases} v_{i,j} & \text{if } (rand_j(0,1) \leq CR) \text{ } or \text{ } (j = j_{rand}) \\ x_{i,j} & \text{otherwise} \end{cases} \tag{6}$$

where $i = 1, 2, \ldots, SN$, $j = 1, 2, \ldots, n$, j_{rand} is a randomly chosen integer from $\{1, 2, \ldots, n\}$ which guarantee that U_i differs from its target vector X_i, $rand_j(0, 1)$ is a random number in the range [0, 1] which is generated with respect to each j, and $CR \in [0, 1]$ is called the crossover control parameter.

Selection operation is implemented to select the better one from each pair of the target vector X_i and the trial vector U_i for the next generation. For the maximization problem

$$X_i = \begin{cases} U_i & \text{if } f(U_i) > f(X_i) \\ X_i & \text{otherwise.} \end{cases} \tag{7}$$

B. Coevolutionary Mechanism and Multiobjective DE

Being fascinated by the prospect and potential of coevolution, many researches have been devoted in developing coevolutionary mechanisms [11]–[21]. For example, Potter and De Jong [11] presented cooperative co-evolution as an explicit means of problem decomposition in EAs. Van den Bergh and Engelbrecht [12] were among the first to apply PSO to a cooperative coevolutionary framework (CPSO). Li and Yao [13] exploited an improved version of CPSO. Park and Ryu [18], [19] proposed a method which possesses two distinct populations: a main population and a reserve population. The main population evolves to search for good solutions while the reserve population evolves to maintain diversity in the main population. Yan *et al.* [20] exploited an improved genetic algorithm named DPAGA which also employs two populations. Zhong *et al.* [21] designed an enhanced DE with dual populations. In this method, two populations cooperate during the evolution, the first of which focuses on global search, while the second one focuses on speeding up convergence.

On the other hand, the idea of transforming a single objective optimization into a multiobjective optimization has been well received. For example, Abbass and Deb [24] firstly suggested that it might be beneficial to apply multiobjective technique to deal with single objective optimization. Jensen [25] introduced the added objectives which guide the search toward solutions containing good building blocks and help the algorithm to avoid local optima. Segura *et al.* [26] proposed a new scheme whose aim is to maintain a better diversity. More research findings about this line of works can be seen in [27].

Recently, some newly developed meta-heuristic models, such as DE, have been incorporated into the designs of multiobjective EAs, so called multiobjective DE (MODE). Abbass *et al.* [28] was the first to apply DE to multiobjective problems and proposed a Pareto-frontier DE. Babu and Jehan [29] used two test problems to verify the performance of MODE. Xue *et al.* [30] developed a Pareto-based MODE where a Pareto-based approach is proposed to implement the differential vectors. Mezura-Montes *et al.* [31] designed a novel DE which uses three simple selection criteria based on feasibility to guide the search toward the feasible regions. Iorio and Li [32] demonstrated that the self-adaptive technique in DE can be simply applied for solving a multiobjective optimization problem where parameters are interdependent.

C. DE for Constrained Optimization

Many researchers have attempted to extend DE to deal with COPs and a lot of DE-based designs (CODEs) have been proposed in the literatures.

In early time, Chiou and Wang [22] developed a hybrid method of DE by embedding two additional operations into the original DE. Lampinen and Zelinka [23] proposed a static-penalty approach, coupled with DE to solve engineering design problems. Storn [35] presented a CODE called CADE by incorporating the idea of constraint adaptation into DE. In the beginning, CADE considers all solutions in the population as feasible and then step-by-step tightens the degree of constraint violation. Lin *et al.* [36] combined DE with a multiplier updating method for solving COPs. In addition, this method applies an adaptive scheme to direct the search. Lampinen [37] designed a novel replacement criterion for DE to handle nonlinear constraint functions. Runarsson and Yao [38] developed a modified variant of the stochastic ranking method [39]. Moreover, a variant of the mutation operator is adopted in this method. Mezura-Montes *et al.* [40] introduced a diversity mechanism into DE. In this method, infeasible solutions with promising objective function values are able to enter the next population. Becerra and Coello [41] exploited a cultural-based DE for COPs. A cultural algorithm maintains two spaces, i.e., population space and belief space. DE works in the population space, while the belief space includes situational, topographical, normative, and history knowledge. The knowledge source of the belief space influences the mutation operation of DE.

In addition to the above algorithms, several related methods were proposed at the special session on COPs in the 2006 IEEE Congress on Evolutionary Computation. Takahama and Sakai [42] presented a method named εDE, which introduces an ε-constrained method to DE. In εDE, a gradient-based mutation is used to locate a feasible solution. Huang *et al.* [43] introduced a self-adaptive mechanism to DE for COPs. In this method, the suitable trial vector generation strategies and the two control parameters (*F* and *CR*) are dynamically changed according to their previous experiences of generating promising solutions. Tasgetiren and Suganthan [44] developed a multipopulation DE named MDE. In MDE, the population is divided into several subpopulations which conduct their search in parallel. Moreover, a regrouping schedule is introduced periodically by MDE to realize information exchange among the subpopulations. Kukkonen and Lampinen [45] presented a generalized DE named GDE to solve COPs. In this method, if it weakly dominates the target vector in the space of the degree of constraint violation or objective function, the target vector is replaced by its trial vector. Brest *et al.* [46] also exploited a self-adaptive DE called jDE. In jDE, three mutation strategies of DE are adopted and the DE control parameters *F* and *CR* are adaptively selected based on their previous experiences. Mezura-Montes *et al.* [47] proposed an improved DE with a new mutation operator. In this new mutation operator, the information on both the best solution in the current population and the current parent is combined to find promising search directions. Zielinski and Laur [48] introduced Deb's feasibility-based rule [10] into DE for solving COPs.

After then, more algorithms have sprung up. Liu *et al.* [62] proposed a memetic co-evolutionary DE for COPs in which two cooperative populations are constructed and evolved by independent DE. Tagawa and Nakajima [63] developed an island-based DE of which the population is divided into several sub-populations, or islands, according to the distributed population model. Mezura-Montes and Cecilia-López-Ramírez [49] conducted a comparative study of four population-based optimization algorithms which employ the same constraint-handling technique, (i.e., Deb's feasibility-based rule) to solve 24 benchmark test functions. These four population-based optimization algorithms include DE, Genetic Algorithm, Evolution Strategy, and Particle Swarm Optimization. The overall experimental results show that DE is the most competitive among the four compared algorithms for these 24 benchmark test functions. Besides, Gong and Cai [50] developed a multiobjective DE algorithm to handle COPs. In this method, the orthogonal design is applied to produce the initial population, and Pareto dominance technique is used to select solutions for the next generation. Takahama and Sakai [51] exploited an improved version of ε-constrained DE for COPs with equality constraints. This method uses dynamic control of the degree of relaxant constraint violation for equality constraints, and specifies the amount of the degree by the ε-level. Zhang *et al.* [52] integrated the multimember DE [40] with a dynamic stochastic selection scheme which is based on stochastic ranking [39] and proposed a new approach for COPs. Zielinski and Laur [53] presented another CODE called DSS-MDE which combines several stopping criteria for DE to solve COPs. In addition, DSS-MDE considers the distribution, improvement or movement of solutions in the population to determine when DE should be terminated. Yuan *et al.* [54] proposed chaotic hybrid cultural algorithm (CA) for COPs in which the population space in their study is DE and the belief space consists of normative and situational knowledge. Moreover, a logistic map function is incorporated into DE for faster convergence speed.

Most recently, based on GA, ES, and DE, Wang *et al*. [55], [56] developed a hybrid framework with global search model and local search model to solve COPs. In these approaches, Pareto dominance used in multiobjective optimization is introduced to compare the solutions in the population. Mallipeddi and Suganthan [57] proposed an ensemble of constraint handling techniques named ECHT for COPs where each constraint handling approach has its own population. In ECHT, the parent population of one constraint handling method competes with all the offspring populations. Wang *et al.* [59] exploited an adaptive tradeoff model called ATM. To satisfy different requirements in corresponding situations, ATM designs different tradeoff schemes during different situations of a search process. Based on ATM, Wang and Cai [60] proposed an improved ATM, named IATM, with each constraint violation first normalized. IATM is combined with (μ+λ)-DE to form the framework of (μ+λ)-constrained DE [namely (μ+λ)-CDE]. In this approach, a constraint-handling mechanism is designed in each situation based on the characteristics of the current population. To overcome the drawbacks of (μ+λ)-CDE, Jia *et al.* [61] presented an improved version of (μ+λ)-CDE named ICDE. ICDE consists of an improved (μ+λ)-DE and a novel archiving based ATM. The experimental results demonstrate that ICDE not only overcomes the main drawbacks of (μ+λ)-CDE, but also obtains very competitive performance compared with selected state-of-the-art designs for COPs.

Although the proposed algorithm in this paper also attempts to solve COPs by making use of DE, its methodology is completely different from the above works. It attempts to employ an information-sharing strategy and a dual-population mechanism. Unlike the above methods, this paper adopts the dual-population mechanism to separately deal with the objective function and the degree of constraint violations, and available information is extracted to promote the interactions between these two subpopulations by the effective information-sharing strategy.

III. Proposed Algorithm

The DPDE is based on the dual-population method that uses two subpopulations to optimize the degree of constraint violations and the objective function value, respectively. In this section, details of the evolutionary process for both subpopulations and the information-sharing strategy between two subpopulations are described. Afterwards, the complete DPDE process is presented.

A. DPDE Evolutionary Process

There are dual subpopulations based on their feasibility working in DPDE. One subpopulation consists of the infeasible solutions to minimize the degree of constraint violations; the other subpopulation consists of the feasible solutions to optimize the objective function value. The evolutionary process in either subpopulation is similar to that in a conventional DE that is used to optimize a single-objective function. Without loss of generality, we here consider only one of the both subpopulations, indicating by index of m ($m = 1, 2$), to describe the evolutionary process.

X_i^m represents solution i in the mth subpopulation. In every generation during the evolutionary process, the mutant vector V_i^m is generated with respect to each target vector X_i^m by the following equation:

$$V_i^m = X_{r1}^m + F \cdot (X_{r2} - X_{r3}^m) \tag{8}$$

where X_{r1}^m and X_{r3}^m are selected from the mth subpopulation in the same way as in (5), while X_{r2} is randomly chosen from the whole population, which is also different from X_i^m, X_{r1}^m, and X_{r3}^m. Here, DE which adopts the mutation equation (8) is named coevolutionary DE.

In the mutation equation, the term $F \cdot (X_{r2} - X_{r3}^m)$ conveys the sharing information. With the help of information in the whole population, the solution can use the search information not only from its own subpopulation but also from the other subpopulation. Specifically, as X_{r2} may come from the subpopulation consisting of feasible solutions, the resulted mutation operation could move an infeasible solution toward feasible regions, and then become feasible solutions to locate feasible optimal solution. Feasible solutions can make use of valuable information hidden in some infeasible solutions to track the better feasible solutions. The solution is expected to search for the feasible optimal solution by making use of the search information of the whole population instead of being attracted to the boundary of the constraint surface only by the search information of its own subpopulation. Therefore, the algorithm can approximate the feasible optimal solution fast with the help of the sharing information. X_{r2} is chosen by randomly selecting a solution from the whole population for the solution i. A random selection method is rapid, has advantages of preserving high diversity and low computational cost. Therefore, it is employed in this paper. At the same time, it is necessary to point out why only one solution is selected from the whole population. It is well know that the primary task of COEAs is to locate the feasible solutions. If more than one solution in (8) is selected from the whole population, there is a high probability that the feasible solution is guided toward the infeasible region. This is clearly unadvisable.

It can be seen from (8) that at least three solutions are needed to form a subpopulation. While, at the early stage, the whole population might contain less than three feasible solutions. As evolution continues, infeasible solutions become feasible solutions. At the later stage, the number of infeasible solutions could be less than three. Therefore, based on the number of feasible solutions NF, the population may inevitably experience three situations.

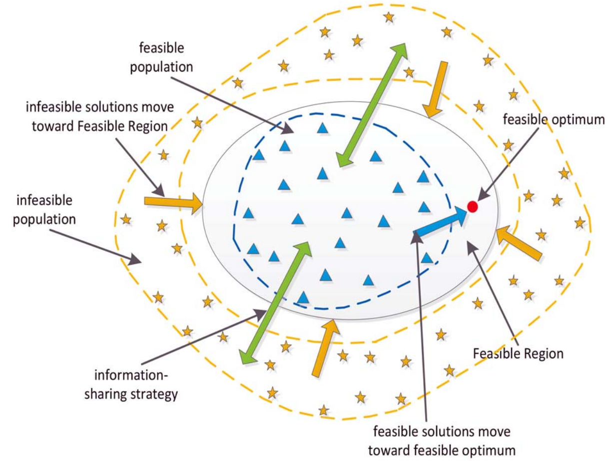

Fig. 1. Schematic diagram to illustrate the tendency of movement of solutions.

1) *Case One:* The population contains few feasible solutions ($NF < 3$).
2) *Case Two:* The population consists of enough feasible solutions and enough infeasible solutions to form dual populations ($3 \le NF \le SN - 3$).
3) *Case Three:* The population is composed of few infeasible solutions ($NF > SN - 3$).

In order to further explain the proposed constraint handling technique, the tendency of movement of solutions in the current population is shown in Fig. 1. Considering case two depicted that is a significant situation during the evolution process, we here take case two as an example to account for our idea which is inspired by the fact that in modern society, the team cooperation and the division of labor play important roles in completing a task. As shown in Fig. 1, there are two subpopulations, i.e., feasible population and infeasible population. Firstly, infeasible population which consists of infeasible solution just focuses on optimizing the degree of constraint violations and feasible population which consists of feasible solution only concentrates on minimizing the objective function value, which can implement a clear division of work. At the same time, the information-sharing strategy which plays a role in the team cooperation can promote the exchange of social information between the two subpopulations. In a word, based on the information-sharing strategy and the dual-population mechanism, the objective of the algorithm design to make infeasible solutions become feasible solutions and drive feasible solutions toward the feasible optimal solution can be successfully achieved. The experimental results of Section IV-B can support the above conclusion.

The detailed description is shown as follows. After computing the number of feasible solutions in the current population, we can determine which case the algorithm should go to. In case one, as $NF < 3$, the feasible solutions can not

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

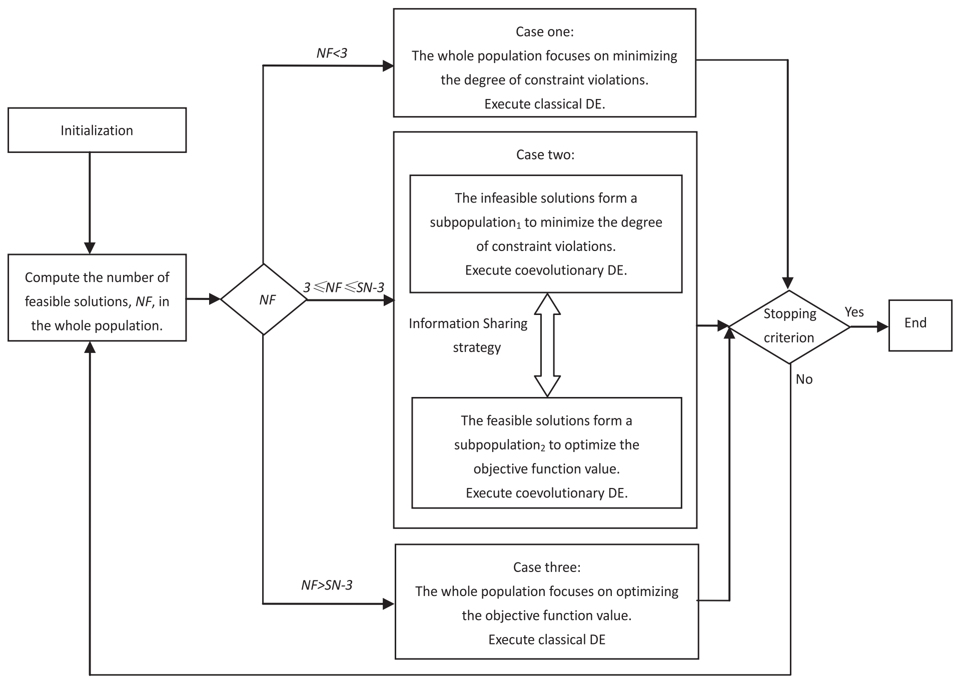

Fig. 2. Framework of DPDE.

form a subpopulation to implement the mutation operation. In this case, the feasibility of a solution is more important than the minimization of its objective function. A desirable search mechanism should guide the population toward feasibility from various directions. Thus, the whole population focuses on minimizing the degree of constraint violations toward the feasible region by DE/rand/1/bin. In case two, the current population is divided into dual subpopulations based on their feasibility. Infeasible solutions explore the search space in the direction of reduction of the constraint violation; whereas the feasible solutions search the feasible region for the exact optima through improving their fitness (i.e., objective value). The underlying mechanism that feasible solutions improve their affinity is based on coevolutionary DE. With regard to the infeasible solutions, the coevolutionary technique can lead the search toward feasibility boundary. It is commonly desired that the infeasible solutions can move toward the feasible regions and become feasible solutions, while feasible solutions search for better feasible solutions. Based on previous analysis, the coevolutionary dual-population mechanism can accomplish the above goal to maintain a delicate balance between feasible and infeasible solutions over the course of evolution. Case three refers to the situation where the current population is nearly composed of feasible solutions. It is obvious that the evolution of this case focuses on optimizing the objective function though DE/rand/1/bin.

B. Proposed Algorithm

The previous subsection describes the evolutionary process. A schematic illustration of the proposed DPDE is shown in Fig. 2. It should be noted that in cases one and three, the whole population might contain both feasible solutions and infeasible solutions. In this situation, if the target vectors are infeasible, the comparisons of solutions are based on the degree of constraint violations; if the target vectors and the trial vectors are feasible, the comparisons of solutions are based on their objective function values.

From Fig. 2, it is clear that the proposed method preserves the fine characteristics of the classical DE, such as simple structure, ease of use, and so on. In addition, since the dual-population mechanism and the information-sharing strategy are adopted to deal with COPs, the solutions in different subpopulations may have different focuses in search direction and information communication can become even more effective. In this way, the infeasible solutions can move toward the feasible regions and become feasible solutions, while feasible solutions locate better feasible solutions. Besides, the two operators are also able to insert into other EAs with minimal changes.

Moreover, DPDE does not significantly increase the overall complexity of the classical DE. Compared to the classical DE, the additional operator added in DPDE is the feasible solutions calculation, as shown in Fig. 1. The complexity of the feasible solutions calculation is $O(SN)$, where SN is the population size. Since the complexity of the classical DE is $O(n \cdot SN)$, where n is the dimension of decision space, the computational complexity of DPDE remains to be $O(n \cdot SN)$.

C. Differences Between DPDE and Other Subpopulation-Based Strategies

Zhong *et al.* [21], Yu and Zhang [17], and Tagawa and Nakajima [63] employed various subpopulation strategies in DE. However, these algorithms are designed specifically for unconstrained optimization problems. Zhong *et al.* [21] proposed an enhanced DE with dual populations—one population for global search, the other for speeding up convergence. The size of the subpopulation is determined in advance. The algorithm proposed in [17], named MPDEA, is similar to the one in [21]. The main difference is that different sub-populations in MPDEA exchange information via a mutation operation instead of migration used in [21]. In [63], the population is divided into several sub-populations, or islands, according to the distributed population model and migration technique is used to exchange information among subpopulations. On the other hand, our approach is designed explicitly for COPs and the size of the each subpopulation can be dynamically adjusted based on individual's feasibility status.

Liu *et al.* [62], Le Riche *et al.* [15], and Hajela and Lee [16] also used multiple subpopulations for COPs. However, there is a radical difference among them. In [62], a threshold parameter is used to judge the boundary of the two subpopulations; in [15], two groups are formed based on two penalty parameters; in [16], a traditional GA works on the whole population with the objective function as the only measure of fitness and the immune system is used to reduce the level of constraint violations with an additional function as the measure of fitness. The information between subpopulations is exchanged by the migration technique in [62], the hybridization in [15], and the immune operation in [16]. However, in our proposed method, the whole population is divided into two subpopulations based on their feasibility without explicit parameter to tune and the exchange of information is performed naturally by the inner mechanism of the information-sharing strategy.

TABLE I
PROPERTIES OF TEST PROBLEMS IN SET 1

Prob.	n	Type of function	ρ	LI	NI	LE	NE	a
g01	13	quadratic	0.0111%	9	0	0	0	6
g02	20	nonlinear	99.9971%	0	2	0	0	1
g03	10	polynomial	0.0000%	0	0	0	1	1
g04	5	quadratic	51.1230%	0	6	0	0	2
g05	4	cubic	0.0000%	2	0	0	3	3
g06	2	cubic	0.0066%	0	2	0	0	2
g07	10	quadratic	0.0003%	3	5	0	0	6
g08	2	nonlinear	0.8560%	0	2	0	0	0
g09	7	polynomial	0.5121%	0	4	0	0	2
g10	8	linear	0.0010%	3	3	0	0	0
g11	2	quadratic	0.0000%	0	0	0	1	1
g12	3	quadratic	4.7713%	0	1	0	0	0
g13	5	nonlinear	0.0000%	0	0	0	3	3
g14	10	nonlinear	0.0000%	0	0	3	0	3
g15	3	quadratic	0.0000%	0	0	1	1	2
g16	5	nonlinear	0.0204%	4	34	0	0	4
g17	6	nonlinear	0.0000%	0	0	0	4	4
g18	9	quadratic	0.0000%	0	13	0	0	0
g19	15	nonlinear	33.4761%	0	5	0	0	0
g20	24	linear	0.0000%	0	6	2	12	16
g21	7	linear	0.0000%	0	1	0	5	6
g22	22	linear	0.0000%	0	1	8	11	19
g23	9	linear	0.0000%	0	2	3	1	6
g24	2	linear	79.6556%	0	2	0	0	2

TABLE II
PROPERTIES OF TEST PROBLEMS IN SET 2

Prob.	n	I	E	$Range$	ρ
H01	10	1	2	[-50, 50]	0.000000
H02	10	2	1	[-5.12, 5.12]	0.000000
H03	10	1	1	[-100, 100]	0.000000
H04	10	0	2	[-100, 100]	0.000000
H05	10	2	0	[-100, 100]	0.008900
H06	10	1	0	[-100, 100]	0.000000
H07	10	2	1	[-100, 100]	0.000000
H08	10	0	1	[-500, 500]	0.000000
H09	10	3	0	[-500, 500]	0.000001
H10	10	1	0	[-10, 10]	0.500300
H11	10	0	2	[-5, 5]	0.000000
H12	10	0	1	[-50, 50]	0.000000
H13	10	2	0	[-100, 100]	0.259900

IV. EXPERIMENT STUDY

A. Experiment Settings

To evaluate the performance of the proposed algorithm, we use two sets of constrained benchmark problems.The set 1 includes 22 test functions (i.e., g01-g19, g21, g23, and g24) [57], [58]. The set 2 includes 13 test functions (i.e., H01-H13) [57]. It was proposed by Mallipeddi and Suganthan [57] who suggest these problems in set 2 are harder than those in set 1. The properties of these benchmark functions are listed in Tables I and II.

These test cases in Table I involve different types of functions such as linear, nonlinear, polynomial, cubic, and quadratic. These benchmark functions vary in the dimension of decision variables, n (i.e., between 2 and 24), and the number of constraints (i.e., between 1 and 38). In the table, ρ, referred to as the feasibility ratio, is the estimated proportion between the feasible region and the search space through 1 000 000 sample solutions, which varies from as low as 0.0000% to as high as 99.9971% in the test functions considered. Here, the ratio is rounded to six decimal places. Thus, 0.0000% is not exactly 0, but inferring an extremely small feasible region with respect to the search space. ρ cannot be equal to 1. If so, the problem will be an unconstrained optimization problem. The various types of constraints for each test function are also reported which include nonlinear inequality (NI), linear inequality (LI), nonlinear equality (NE), and linear equality (LE). a in Table I denotes the number of active constraint included at the optimal solution. For most of these test functions, it is not easy even locating a feasible region. The problems in Table II are considered harder problems. I and E represent the numbers of inequality and equality constraints, respectively. Range indicates the lower and upper bounds of the decision variable in the search space.

The parameters in DPDE are set as follows: $SN = 100$, $F = 0.8$, $CR = 0.9$ [60], [61]. For each test function, DPDE runs 25 times independently and is stopped when a maximum of 240 000 fitness evaluations (FES) is reached in each run.

In order to deal with equality constraints, each of them is converted into inequality constraints as $|h_i(X)| - \delta \leq 0$, $i = q+1, \ldots, m)$, where δ is a small tolerance value. A dynamic setting of the parameter δ, which is similarly used in [59]–[61] is adopted. The parameter decreases with respect to the current

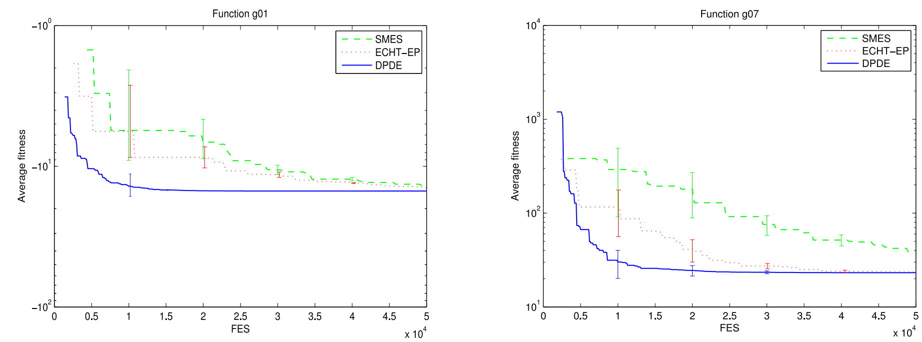

Fig. 3. Fitness curves and error bars averaged over 25 runs for test functions g01 and g07, respectively.

generation using the following expression:

$$\delta_{t+1} = \begin{cases} \frac{\delta_t}{\delta'} & \text{if } \delta_t > 0.0001 \\ 0.0001 & \text{otherwise} \end{cases} \tag{9}$$

where t is the generation number, the initial value of δ (i.e., δ_0) is set to $n \cdot (\log_{10}(\max_i(u_i - l_i)) + 1)$, and the change rate of δ (i.e., δ') is equal to 1.015.

B. Convergence Curves and Size of the Feasible Population

For each benchmark function with FES of 240 000, two graphs are plotted which investigate the changes of the population average fitness and the corresponding standard deviation, when compared with some state-of-the-art designs (i.e., ES+SR [39], HCOEA [55], SMES [64], ATMES [59], and ECHT-EP [57]). These graphs for two benchmark test functions (i.e., g01 and g07) are shown in Fig. 3. Please note vertical axis is plotted in log scale in order to show the appreciable differences. There are two observations that can be drawn from Fig. 3. On one hand, it can be observed that the proposed approach, DPDE, has better convergence ability. On the other hand, it can be seen obviously that the proposed approach is more stable than the other compared algorithms, obtaining minimal standard deviation consistently.

The size of the feasible population during the search process on function g03 is shown in Fig. 4. Equation (9) is applied to deal with the equality constraints. As function g03 involves equality constraints, its feasible region varies during the search. Thus, the size of the feasible population fluctuates widely throughout the search process before it reaches to the feasibility region.

C. Comparison With Other COEAs on Solution Accuracy

To compare the performance of DPDE with respect to some well-known findings in literatures, six state-of-the-art designs from different branches of evolutionary computation are chosen as competitors. These highly regarded algorithms are ES+SR [39], HCOEA [55], SMES [64], ATMES [59], ECHT-EP [57], and ECHT-DE [57], which are very popular and often cited by researchers in the COEAs community. The results of these compared algorithms are based on the reports in the original publications. The comparison results are shown in Tables III–V in terms of the best, mean, worst,

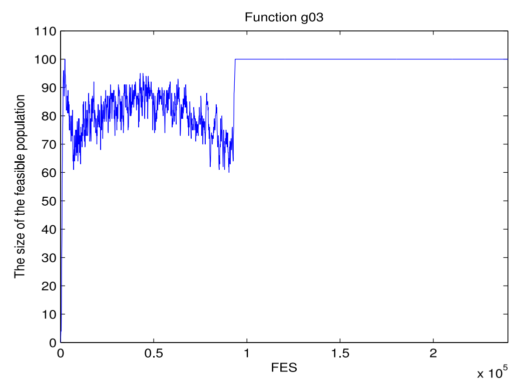

Fig. 4. Size of the feasible population during the search process.

and standard deviation (SD) of the objective value for the best-so-far solution given the budgeted FES over 25 independent runs. In addition, in order to show the significant differences between two algorithms, the Mann–Whitney–Wilcoxon rank sum test on the mean found by two competing algorithms at 5% significance level is also conducted and shown in Tables III–V. The result of the U-test is represented as "+/=/-" at the end of each mean value obtained by the corresponding competing algorithm, which means that DPDE is significantly better than, equal to, or worse than the compared algorithm, respectively. For example, DPDE performs significantly better than ES+SR on g02, g03, g04, g05, g06, g07, g09, g10, g11, and g13, and equally to ES+SR on g01, g08, and g12. The results of the compared algorithms are all derived directly from their corresponding references. For functions from test function set 1, the optimal value is also represented behind the corresponding test function in Tables III and IV. Please note there is no optimal solution ever reported for the test functions g20 and g22. Unlike functions from test function set 1, no exact optimal value is made available for test function set 2 by the authors in [57]. For ES+SR, HCOEA, SMES, and ATMES, the results about g14-g19, g21, g23-g24, and H01-H13 are not available in the corresponding references.

Some insightful observations can be drawn from Tables III–V.

1) DPDE could perform as well as or in most cases better than peer algorithms for all problems in set 1. For

TABLE III
COMPARISON BETWEEN VARIOUS STATE-OF-THE-ART METHODS ON TEST SET 1 (g01-g14)

Algorithms	FES	g01/-15.0000				g02/-0.803619			
		Best	Mean	Worst	SD	Best	Mean	Worst	SD
ES + SR	3.5E+5	**-15.0000**	**-15.0000**$^{=}$	**-15.0000**	**0.0E+00**	-0.803515	-0.781975^{+}	-0.726288	2.0E-02
HCOEA	2.4E+5	**-15.0000**	**-15.0000**$^{=}$	-14.9999	4.2E-07	-0.803550	-0.792610^{+}	-0.756938	1.0E-02
SMES	2.4E+5	**-15.0000**	**-15.0000**$^{=}$	**-15.0000**	**0.0E+00**	-0.803601	-0.785238^{+}	-0.751322	1.67E-02
ATMES	2.4E+5	**-15.0000**	**-15.0000**$^{=}$	**-15.0000**	1.6E-14	-0.803388	-0.790148^{+}	-0.756986	1.3E-02
ECHT-EP	2.4E+5	**-15.0000**	**-15.0000**$^{=}$	**-15.0000**	**0.0E+00**	**-0.803619**	-0.799822^{+}	-0.785182	6.29E-03
DPDE	2.4E+5	**-15.0000**	**-15.0000**	**-15.0000**	**0.0E+00**	**-0.803619**	**-0.802350**	**-0.795326**	**1.18E-03**
Algorithms	FES	g03/-1.0005				g04/-30665.5387			
		Best	Mean	Worst	SD	Best	Mean	Worst	SD
ES + SR	3.5E+5	-1.000	-1.000^{+}	-1.000	1.9E-04	-30665.539	-30665.539^{+}	-30665.539	2.0E-05
HCOEA	2.4E+5	-1.000	-1.000^{+}	-1.000	1.3E-12	-30665.539	-30665.539^{+}	-30665.539	5.4E-07
SMES	2.4E+5	-1.000	-1.000^{+}	-1.000	2.09E-04	-30665.539	-30665.539$^{=}$	-30665.539	0.0E+00
ATMES	2.4E+5	-1.000	-1.000^{+}	-1.000	5.9E-05	-30665.539	-30665.539^{+}	-30665.539	7.4E-12
ECHT-EP	2.4E+5	**-1.0005**	**-1.0005**$^{=}$	**-1.0005**	**0.0E+00**	**-30665.5387**	**-30665.5387**$^{=}$	**-30665.5387**	**0.0E+00**
DPDE	2.4E+5	**-1.0005**	**-1.0005**	**-1.0005**	**0.0E+00**	**-30665.5387**	**-30665.5387**	**-30665.5387**	**0.0E+00**
Algorithms	FES	g05/5126.4967				g06/-6961.8139			
		Best	Mean	Worst	SD	Best	Mean	Worst	SD
ES + SR	3.5E+5	**5126.497**	5128.881^{+}	5142.472	3.5E+00	-6961.814	-6875.940^{+}	-6350.262	1.6E+02
HCOEA	2.4E+5	5126.4981	5126.4981^{+}	5126.4981	1.7E-07	-6961.81388	-6961.81388^{+}	-6961.81388	8.5E-12
SMES	2.4E+5	5126.599	5174.492^{+}	5160.198	5.006E+01	-6961.814	-6961.284^{+}	-6952.482	1.85E+00
ATMES	2.4E+5	5126.498	5127.648^{+}	5135.256	1.8E+00	-6961.814	-6961.814^{+}	-6961.814	4.6E-12
ECHT-EP	2.4E+5	**5126.4967**	**5126.4967**$^{=}$	**5126.4967**	**0.0E+00**	**-6961.8139**	**-6961.8139**$^{=}$	**-6961.8139**	**0.00E+00**
DPDE	2.4E+5	**5126.4967**	**5126.4967**	**5126.4967**	**0.0E+00**	**-6961.8139**	**-6961.8139**	**-6961.8139**	**0.00E+00**
Algorithms	FES	g07/24.3062				g08/-0.095825			
		Best	Mean	Worst	SD	Best	Mean	Worst	SD
ES + SR	3.5E+5	24.307	24.374^{+}	24.642	6.6E-02	**-0.095825**	**-0.095825**$^{=}$	**-0.095825**	2.6E-17
HCOEA	2.4E+5	24.306	24.307^{+}	24.309	7.1E-04	**-0.095825**	**-0.095825**$^{=}$	**-0.095825**	2.1E-17
SMES	2.4E+5	24.327	24.475^{+}	24.843	1.32E-01	**-0.095825**	**-0.095825**$^{=}$	**-0.095825**	**0.0E+00**
ATMES	2.4E+5	24.306	24.316^{+}	24.359	1.1E-02	**-0.095825**	**-0.095825**$^{=}$	**-0.095825**	2.8E-17
ECHT-EP	2.4E+5	**24.3062**	24.3063^{+}	24.3063	3.19E-05	**-0.095825**	**-0.095825**$^{=}$	**-0.095825**	**0.0E+00**
DPDE	2.4E+5	**24.3062**	**24.3062**	**24.3062**	**6.25E-09**	**-0.095825**	**-0.095825**	**-0.095825**	**0.0E+00**
Algorithms	FES	g09/680.630057				g10/7049.248			
		Best	Mean	Worst	SD	Best	Mean	Worst	SD
ES + SR	3.5E+5	680.630	680.656^{+}	680.763	3.4E-02	7054.316	7559.192^{+}	8835.655	5.3E+02
HCOEA	2.4E+5	**680.630057**	**680.630057**$^{=}$	**680.630057**	9.4E-08	7049.286	7049.525^{+}	7049.984	1.5E-01
SMES	2.4E+5	680.632	680.643^{+}	680.719	1.55E-02	7051.903	7253.047^{+}	7638.366	1.36E+02
ATMES	2.4E+5	680.630	680.639^{+}	680.673	1.0E-02	7052.253	7250.437^{+}	7240.224	1.2E+02
ECHT-EP	2.4E+5	**680.630057**	**680.630057**$^{=}$	**680.630057**	2.61E-08	**7049.248**	7049.249^{+}	7049.250	6.60E-04
DPDE	2.4E+5	**680.630057**	**680.630057**	**680.630057**	**3.65E-14**	**7049.248**	**7049.248**	**7049.248**	**8.36E-08**
Algorithms	FES	g11/0.749900				g12/-1.000			
		Best	Mean	Worst	SD	Best	Mean	Worst	SD
ES + SR	3.5E+5	0.75	0.75^{+}	0.75	8.0E-05	**-1.0000**	**-1.0000**$^{=}$	**-1.0000**	**0.0E+00**
HCOEA	2.4E+5	0.75	0.75$^{=}$	0.75	1.5E-12	**-1.0000**	**-1.0000**$^{=}$	**-1.0000**	**0.0E+00**
SMES	2.4E+5	0.75	0.75^{+}	0.75	1.52E-04	**-1.0000**	**-1.0000**$^{=}$	**-1.0000**	**0.0E+00**
ATMES	2.4E+5	0.75	0.75^{+}	0.75	3.4E-04	-1.000	-1.000^{+}	-0.994	1.0E-03
ECHT-EP	2.4E+5	**0.7499**	**0.7499**$^{=}$	**0.7499**	**0.0E+00**	**-1.0000**	**-1.0000**$^{=}$	**-1.0000**	**0.0E+00**
DPDE	2.4E+5	**0.7499**	**0.7499**	**0.7499**	**0.0E+00**	**-1.0000**	**-1.0000**	**-1.0000**	**0.0E+00**
Algorithms	FES	g13/0.0539415				g14/-47.7649			
		Best	Mean	Worst	SD	Best	Mean	Worst	SD
ES + SR	3.5E+5	0.053957	0.067543^{+}	0.216915	3.1E-02				
HCOEA	2.4E+5	0.0539498	0.0539498^{+}	0.0539499	8.6E-08				
SMES	2.4E+5	0.053986	0.166385^{+}	0.468294	1.77E-01				
ATMES	2.4E+5	0.053950	0.053952^{+}	0.053959	1.3E-05				
ECHT-EP	2.4E+5	**0.0539415**	**0.0539415**$^{=}$	**0.0539415**	1.00E-12	**-47.7649**	-47.7648^{+}	-47.7648	2.72E-05
DPDE	2.4E+5	**0.0539415**	**0.0539415**	**0.0539415**	**1.16E-17**	**-47.7649**	**-47.7649**	**-47.7649**	**2.56E-09**

the most test functions in Table III, DPDE is superior to ES+SR, HCOEA, SMES, and ATMES in terms of the best, worst, mean, and standard deviation. For the 22 test problems in set 1, DPDE outperforms ECHT-EP on eight test problems (i.e., g02, g07, g10, g14, g17, g19, g21, g23). Although DPDE and ECHT-EP perform the same on the six test problems (i.e., g09, g13, g15, g16, g18, g24), DPDE improves the robustness in performance on these cases.

2) For the 13 test functions in set 2, DPDE performs better than ECHT-EP and ECHT-DE on nine cases (except H06, H07, H08, and H13) and eight cases (except

TABLE IV
COMPARISON BETWEEN VARIOUS STATE-OF-THE-ART METHODS ON TEST SET 1 (g15-g24)

Algorithms	FES	g15/ 961.715022				g16/-1.905155			
		Best	Mean	Worst	SD	Best	Mean	Worst	SD
ECHT-EP	2.4E+5	**961.715022**	**961.715022**$^{=}$	**961.715022**	2.01E-13	**-1.905155**	**-1.905155**$^{=}$	**-1.905155**	1.12E-10
DPDE	2.4E+5	**961.715022**	**961.715022**	**961.715022**	**0.0E+00**	**-1.905155**	**-1.905155**	**-1.905155**	**0.0E+00**
Algorithms	FES	g17/8853.5338748				g18/-0.86602540			
		Best	Mean	Worst	SD	Best	Mean	Worst	SD
ECHT-EP	2.4E+5	8853.5397	8853.5397^{+}	8853.5397	2.13E-08	**-0.866025**	**-0.866025**$^{=}$	**-0.866025**	1.00E-09
DPDE	2.4E+5	**8853.5338748**	**8853.5338748**	**8853.5338748**	**2.34E-12**	**-0.86602540**	**-0.86602540**	**-0.86602540**	**1.65E-12**
Algorithms	FES	g19/32.6556				g20			
		Best	Mean	Worst	SD	Best	Mean	Worst	SD
ECHT-EP	2.4E+5	32.6591	32.6623^{+}	32.6687	3.4E-03				
DPDE	2.4E+5	**32.6556**	**32.6556**	**32.6556**	**6.17E-08**	0.2098	0.2165	0.2201	1.2E-02
Algorithms	FES	g21/193.7245				g22			
		Best	Mean	Worst	SD	Best	Mean	Worst	SD
ECHT-EP	2.4E+5	193.7246	193.7438^{+}	193.7741	1.65E-02				
DPDE	2.4E+5	**193.7245**	**193.7262**	**193.7536**	**8.36E-04**	258.2308	269.8261	294.5372	1.96E+01
Algorithms	FES	g23/-400.0551				g24 /-5.5080			
		Best	Mean	Worst	SD	Best	Mean	Worst	SD
ECHT-EP	2.4E+5	-398.9731	-373.2178^{+}	-335.1145	3.37E+01	**-5.5080**	**-5.5080**$^{=}$	**-5.5080**	1.8E-15
DPDE	2.4E+5	**-400.0551**	**-399.2752**	**-396.9405**	**1.52E-00**	**-5.5080**	**-5.5080**	**-5.5080**	**0.0E+00**

TABLE V
COMPARISON BETWEEN VARIOUS STATE-OF-THE-ART METHODS ON TEST SET 2

Algorithms	FES	H01				H02			
		Best	Mean	Worst	SD	Best	Mean	Worst	SD
ECHT-EP	2.4E+5	5.58E-13	3.02E-11^{+}	1.04E-10	3.19E-11	**-2.2776**	-2.2764^{+}	-2.2709	0.0021
ECHT-DE	2.4E+5	8.29E-83	2.66E-78^{+}	7.41E-77	1.35E-77	**-2.2776**	-2.2516^{+}	-2.2168	0.0031
DPDE	2.4E+5	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**-2.2777**	**-2.2777**	**-2.2777**	**2.67E-10**
Algorithms	FES	H03				H04			
		Best	Mean	Worst	SD	Best	Mean	Worst	SD
ECHT-EP	2.4E+5	1.00E-15	3.26E-10^{+}	2.38E-09	7.43E-10	1.54E-13	1.89E-11^{+}	1.72E-10	5.40E-11
ECHT-DE	2.4E+5	1.19E-83	6.90E-81^{+}	3.68E-80	1.12E-80	4.91E-95	1.01E-92^{+}	7.98E-92	1.85E-92
DPDE	2.4E+5	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**
Algorithms	FES	H05				H06			
		Best	Mean	Worst	SD	Best	Mean	Worst	SD
ECHT-EP	2.4E+5	**-20.0780**	-19.3877^{+}	-18.0109	0.5997	**-8.3826**	**-8.3826**$^{=}$	**-8.3826**	1.77E-05
ECHT-DE	2.4E+5	**-20.0780**	-20.0774^{+}	-20.0599	0.0033	**-8.3826**	**-8.3826**$^{=}$	**-8.3826**	3.76E-15
DPDE	2.4E+5	**-20.0780**	**-20.0780**	**-20.0780**	**2.38E-08**	**-8.3826**	**-8.3826**	**-8.3826**	**0.0E+00**
Algorithms	FES	H07				H08			
		Best	Mean	Worst	SD	Best	Mean	Worst	SD
ECHT-EP	2.4E+5	**-7.6159**	**-7.6159**$^{=}$	**-7.6159**	3.18E-09	**-483.6106**	**-483.6106**$^{=}$	**-483.6106**	**0.00E+00**
ECHT-DE	2.4E+5	**-7.6159**	**-7.6159**$^{=}$	**-7.6159**	4.26E-10	**-483.6106**	**-483.6106**$^{=}$	**-483.6106**	**0.00E+00**
DPDE	2.4E+5	**-7.6159**	**-7.6159**	**-7.6159**	**0.0E+00**	**-483.6106**	**-483.6106**	**-483.6106**	**0.00E+00**
Algorithms	FES	H09				H10			
		Best	Mean	Worst	SD	Best	Mean	Worst	SD
ECHT-EP	2.4E+5	**-68.4294**	-64.9120^{+}	-63.5174	2.0358	**0.0000**	3.5970^{+}	8.9900	4.6416
ECHT-DE	2.4E+5	**-68.4294**	-67.9231^{+}	-63.5175	1.0938	**0.0000**	0.5993^{+}	8.9900	2.2808
DPDE	2.4E+5	**-68.4294**	**-68.1101**	**-68.0945**	**2.71E-01**	**0.0000**	**2.68E-02**	**4.3567**	**1.5102**
Algorithms	FES	H11				H12			
		Best	Mean	Worst	SD	Best	Mean	Worst	SD
ECHT-EP	2.4E+5	**580.7301**	580.7303^{+}	580.7310	0.0003	5.00E-07	1.95E-06^{+}	1.06E-05	3.06E-06
ECHT-DE	2.4E+5	**580.7301**	**580.7301**$^{=}$	**580.7301**	1.32E-11	1.54E-32	4.55E-31^{+}	1.75E-30	4.61E-31
DPDE	2.4E+5	**580.7301**	**580.7301**	**580.7301**	**0.00E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**
Algorithms	FES	H13							
		Best	Mean	Worst	SD				
ECHT-EP	2.4E+5	**-46.3755**	**-46.3755**$^{=}$	**-46.3755**	7.15E-10				
ECHT-DE	2.4E+5	**-46.3755**	**-46.3755**$^{=}$	**-46.3755**	9.47E-15				
DPDE	2.4E+5	**-46.3755**	**-46.3755**	**-46.3755**	**0.0E+00**				

H06, H07, H08, H11, and H13), respectively. Although DPDE perform the same with ECHT-EP and ECHT-DE on some cases (e.g., H06, H07, H08, and H13), DPDE shows more robustness in performance. As far as the computational cost (the number of FES) is concerned, HCOEA, SMES, ATMES, ECHT-EP, ECHT-DE, and DPDE have the minimum computational cost (240 000 FES) for all the test functions, while ES+SR has a higher computational cost (350 000 FES) for all the test functions.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE VI
NUMBER OF FES TO ACHIEVE THE SUCCESS CONDITION, SUCCESS RATE, FEASIBLE RATE, AND SUCCESS PERFORMANCE

Prob.	Best	Median	Worst	Mean	SD	Feasible Rate	Success Rate	Success Performance
g01	41,600	41,800	44,600	42,180	1,289.3	100%	100%	42,180
g02	86,400	91,400	112,100	95,102	6,567.3	100%	94%	95,102
g03	88,300	88,300	89,600	88,260	553.2	100%	100%	88,260
g04	25,500	26,200	27,100	25,160	510.3	100%	100%	25,160
g05	92,700	107,800	114,400	100,506	8,890.8	100%	100%	100,506
g06	13,800	14,200	15,700	13,400	738.2	100%	100%	13,400
g07	95,700	98,200	104,000	99,060	3,365.3	100%	100%	99,060
g08	1,800	1,900	2,300	1,960	239.2	100%	100%	1,960
g09	29,200	30,100	34,000	31,820	2,020.9	100%	100%	31,820
g10	133,400	136,700	161,400	143,300	10,256.4	100%	100%	143,300
g11	90,200	90,200	90,400	90,310	65.2	100%	100%	90,310
g12	4,200	4,300	7,000	5,626	2,368.9	100%	100%	5,626
g13	81,300	81,900	82,200	81,980	3,632.1	100%	100%	81,980
g14	90,000	103,200	134,100	107,480	20,790.6	100%	100%	107,480
g15	83,900	93,000	100,400	94,600	6,560.8	100%	100%	94,600
g16	17,700	18,000	20,400	18,650	1,106.5	100%	100%	18,650
g17	111,600	123,800	142,100	128,690	15,356.5	100%	100%	128,690
g18	79,000	80,200	83,600	80,280	2,090.4	100%	100%	80,340
g19	156,600	157,700	172,700	163,080	6,445.3	100%	100%	163,280
g21	153,200	166,100	173,300	164,068	10,023.3	100%	92%	164,068
g23	198,200	206,200	222,500	204,450	9,264.6	100%	94%	204,450
g24	5,000	6,800	6,400	5,860	542.2	100%	100%	5,860

TABLE VII
COMPARISON OF DPDE WITH STATE-OF-THE-ART DE METHODS IN TERMS OF FEASIBLE RATE AND SUCCESS RATE

Prob.	Feasible rate						Success rate					
	SaDE	MPDE	GDE	jDE-2	DSS-MDE	DPDE	SaDE	MPDE	GDE	jDE-2	DSS-MDE	DPDE
g02	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	84%	92%	72%	92%	36%	**94%**
g03	**100%**	**100%**	96%	**100%**	**100%**	**100%**	96%	84%	4%	0%	**100%**	**100%**
g05	**100%**	**100%**	96%	**100%**	**100%**	**100%**	**100%**	**100%**	92%	68%	**100%**	**100%**
g10	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	92%	**100%**
g11	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	96%	**100%**	96%	**100%**	**100%**
g13	**100%**	88%	88%	**100%**	**100%**	**100%**	**100%**	48%	40%	0%	**100%**	**100%**
g14	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	80%	**100%**	96%	**100%**	84%	**100%**
g15	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	96%	96%	**100%**	**100%**
g16	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	0%	**100%**
g17	**100%**	96%	76%	**100%**	**100%**	**100%**	4%	28%	16%	4%	**100%**	**100%**
g18	**100%**	**100%**	84%	**100%**	**100%**	**100%**	92%	**100%**	76%	**100%**	**100%**	**100%**
g19	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	88%	**100%**	68%	**100%**
g21	**100%**	**100%**	88%	**100%**	**100%**	**100%**	60%	68%	60%	**92%**	72%	**92%**
g23	**100%**	**100%**	88%	**100%**	**100%**	**100%**	88%	**100%**	40%	92%	16%	94%
Mean	**100%**	98.85%	94%	**100%**	**100%**	**100%**	86.01%	86.86%	70%	74.28%	76.28%	**98.57%**

D. Comparison With Other CODEs on Convergence Speed

In order to verify convergence speed, more experimental results are obtained in Table VI. Table VI records the number of FES needed in each run for satisfying the success condition: $f(X) - f(X^*) < 0.0001$, where X is a feasible solution and X^* is the the best known optimal solution. In the same table, feasible rate, success rate, and success performance are also reported for all of the 22 test functions in set 1. Feasible rate denotes the percentage of runs where at least one feasible solution is found in 240 000 FES. Success rate denotes the percentage of runs (out of 25) where the algorithm finds a solution that satisfies the success condition. The success performance denotes the mean number of FES for successful runs multiplied by the number of total runs and divided by the number of successful runs.

As shown in Table VI, the feasible rate of 100% has been achieved for all of the 22 test functions. Regarding the success rate, DPDE is capable of achieving a value of 100% for all of the test functions with the exception of test functions g02, g21, and g23. By making use of the indicator "success performance," we can see that g08, g12, and g24 will reach the optimum solution fairly early at FES of 1960, 5626, and 5860, respectively, showing that g08, g12, and g24 are fairly easier than the others in the set of 22 benchmark problems. On the other hand, it takes DPDE 204 450 FES to reach the optimal solution for g23, showing that g23 is much harder than the others in the set of 22 benchmark problems.

Further, DPDE is compared against five CODEs: SaDE [43], MPDE [44], GDE [45], jDE-2 [46], and DSS-MDE [52], using three performance metrics: feasible rate, success rate, and success performance. The experimental results of these five approaches are directly taken from their references and are compared with those of DPDE in Tables VII and VIII. Please note for test functions g01, g04, g06, g07, g08, g09, and g12,

TABLE VIII
COMPARISON OF DPDE WITH STATE-OF-THE-ART DE METHODS ON SUCCESS PERFORMANCE

Prob.	Success performance					Prob.	Success performance				
	MPDE	GDE	jDE-2	DSS-MDE	DPDE		MPDE	GDE	jDE-2	DSS-MDE	DPDE
g01	4.3E+04	**4.1E+04**	5.0E+04	1.2E+05	4.2E+04	g12	4.2E+03	3.1E+03	6.4E+03	**2.9E+03**	5.6E+03
g02	3.0E+05	1.5E+05	1.5E+05	2.5E+05	**9.5E+04**	g13	7.4E+05	8.7E+05	N/A	**4.4E+04**	8.2E+04
g03	**2.5E+04**	3.5E+06	N/A	1.0E+05	8.8E+04	g14	**4.3E+04**	2.3E+05	9.8E+04	3.7E+05	1.0E+05
g04	2.1E+04	**1.5E+04**	4.1E+04	4.6E+04	2.5E+04	g15	2.0E+05	7.5E+04	2.4E+05	**3.5E+04**	9.4E+04
g05	2.2E+05	1.9E+05	4.5E+05	**4.5E+04**	1.0E+05	g16	1.3E+04	**1.3E+04**	3.2E+04	N/A	1.8E+04
g06	**1.1E+04**	6.5E+03	2.9E+04	1.6E+04	1.3E+04	g17	7.3E+05	2.1E+06	1.1E+07	**5.8E+04**	1.2E+05
g07	**5.7E+04**	1.2E+05	1.3E+05	1.1E+05	9.9E+04	g18	**4.4E+04**	4.8E+05	1.0E+05	1.1E+05	8.0E+04
g08	**1.5E+03**	**1.5E+03**	3.2E+03	2.3E+03	1.9E+03	g19	**1.2E+05**	2.3E+05	2.0E+05	5.6E+05	1.6E+05
g09	**2.1E+04**	3.0E+04	5.5E+04	3.8E+04	3.1E+04	g21	2.1E+05	5.8E+05	**1.3E+05**	2.0E+05	1.6E+05
g10	**4.8E+04**	8.3E+04	1.5E+05	2.6E+05	1.4E+05	g23	2.1E+05	1.1E+06	3.6E+05	2.1E+06	**2.0E+05**
g11	2.3E+04	**8.5E+03**	5.4E+04	1.9E+04	9.0E+04	g24	4.3E+03	**3.1E+03**	1.0E+04	6.6E+03	5.8E+03
Sum	3.1E+06	9.8E+06	1.3E+7+2N/A	4.6E+6+N/A	**1.7E+06**						

since almost all of the competing algorithms can achieve the 100% feasible rate and success rate consistently, they are not included in Table VII.

It can be seen from Table VII that DPDE has similar performance with SaDE, jDE-2, and DSS-MDE and exhibits superior performance compared with MPDE and GDE, in terms of the mean feasible rate. However, SaDE requires gradient information. With respect to the mean success rate, DPDE performs significantly better than the other five CODEs. It is interesting to observe that only MPDE can achieve a 100% success rate for test function g23. Owing to its special characteristic, g23 is very difficult to optimize for most algorithms. Therefore, ameliorating the algorithm for these special problems is our future work.

Since SaDE employs the sequential quadratic programming as the local search operator, we cannot exactly calculate how many FES is spent by the local search. Because of this, the results of SaDE are not included in Table VIII. In this table, "N/A" denotes that the success performance is not available since the corresponding success rate of a method is 0%. We test the efficiency of these methods by the sum of the success performance, since the evaluation of the objective function and the degree of constraint violations may consume the main computation time at each generation, especially for some complicated COPs. It is evident from Table VIII that DPDE exhibits the highest efficiency over the chosen competitors as far as the success performance is concerned.

Through the extensive comparisons, DPDE is found fairly competitive with respect to the chosen state-of-the-art designs in the constrained optimization problems currently available.

V. CONCLUSION

DE has been widely adopted to solve constrained optimization problems. However, it is difficult for an algorithm to consider the objective function and the degree of constraint violations as a whole since one objective often conflicts with the other. To address this concerning issue, a coevolutionary dual-population DE named DPDE has been proposed for solving COPs in this paper. In DPDE, the whole population is divided into dual subpopulations based on their feasibility so that both objectives are treated separately and either subpopulation focuses on only optimizing the corresponding objective. Meanwhile, the information between the dual populations communicates through the information-sharing strategy. In other words, the information-sharing strategy and the dual-population mechanism seamlessly implement the team cooperation and the division of labor.

The proposed DPDE shows competitive results when performing extensive experiments on 35 benchmark test functions. The comparison study with the chosen state-of-the-art constrained optimization techniques indicates that DPDE is able to perform competitively in terms of commonly used performance metrics, namely, solution accuracy, feasible rate, and success rate. As a future work, the proposed framework for single-objective optimization will be extended into a multiobjective DE to exploit its robust performance under the complex environment [65]–[69]. Additionally, applications of DPDE to medical diagnosis and structural control will be exploited.

ACKNOWLEDGMENT

The authors would like to thank Dr. Y. Wang for making the source code of HCOEA and ATMES available online. They would also like to thank Dr. P. N. Suganthan for providing the source code of ECHT-EP.

REFERENCES

[1] Z. Michalewicz and M. Schoenauer, "Evolutionary algorithms for constrained parameter optimization problems," *Evol. Comput.*, vol. 4, no. 1, pp. 1–32, 1996.

[2] Z. Michalewicz, "A survey of constraint handling techniques in evolutionary computation methods," in *Proc. 4th Evol. Program.*, 1995, pp. 135–155.

[3] Z. Michalewicz and D. B. Fogel, *How to Solve It: Modern Heuristics*. New York, NY, USA: Springer-Verlag, 2004.

[4] S. Venkatraman and G. G. Yen, "A generic framework for constrained optimization using genetic algorithms," *IEEE Trans. Evol. Comput.*, vol. 9, no. 4, pp. 424–435, Aug. 2005.

[5] J. Knowles, "Closed-loop evolutionary multiobjective optimization," *IEEE Comput. Intell. Mag.*, vol. 4, no. 3, pp. 77–91, Aug. 2009.

[6] R. Farmani and J. Wrigth, "Self-adaptive fitness formulation for constrained optimization," *IEEE Trans. Evol. Comput.*, vol. 7, no. 5, pp. 445–455, Oct. 2003.

[7] H. Muehlenbein, M. Gorges-Schleuter, and O. Kraemer, "Evolution algorithms in combinatorial optimization," *Parallel Comput.*, vol. 7, no. 1, pp. 65–85, 1988.

[8] B. Tessema and G. G. Yen, "An adaptive penalty formulation for constrained evolutionary optimization," *IEEE Trans. Syst., Man, Cybern., A, Syst. Humans*, vol. 39, no. 3, pp. 565–578, Apr. 2009.

[9] M. Daneshyari and G. G. Yen, "Constrained multiple-swarm particle swarm optimization within a cultural framework," *IEEE Trans. Syst., Man, Cybern., A, Syst. Humans*, vol. 42, no. 2, pp. 475–490, Mar. 2012.

[10] K. Deb, "An efficient constraint handling method for genetic algorithms," *Comput. Methods Appl. Mech. Eng.*, vol. 186, nos. 2–4, pp. 311–338, 2000.

[11] M. A. Potter and K. A. De Jong, "A cooperative coevolutionary approach to function optimization," in *Proc. Int. Conf. Parallel Problem Solving Nature*, Jerusalem, Israel, 1994, pp. 249–257.

[12] F. van den Bergh and A. P. Engelbrecht, "A cooperative approach to particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 225–239, Jun. 2004.

[13] X. D. Li and X. Yao, "Cooperatively coevolving particle swarms for large scale optimization," *IEEE Trans. Evol. Comput.*, vol. 16, no. 2, pp. 210–224, Apr. 2012.

[14] Z. Yang, K. Tang, and X. Yao, "Multilevel cooperative coevolution for large scale optimization," in *Proc. IEEE Congr. Evol. Comput.*, Hong Kong, 2008, pp. 1663–1670.

[15] R. G. Le Riche, C. Knopf-Lenoir, and R. T. Haftka, "A segregated genetic algorithm for constrained structural optimization," in *Proc. 6th Int. Conf. Genet. Algorithms*, 1995, pp. 558–565.

[16] P. Hajela and J. Lee, "Constrained genetic search via schema adaptation. An immune network solution," in *Proc. 1st World Congr. Struct. Multidiscipl. Optim.*, 1995, pp. 915–920.

[17] W. J. Yu and J. Zhang, "Multi-population differential evolution with adaptive parameter control for global optimization," in *Proc. 13th Annu. Conf. Genet. Evol. Comput.*, 2011, pp. 1093–1098.

[18] T. Park and K. R. Ryu, "A dual-population genetic algorithm for adaptive diversity control," *IEEE Trans. Evol. Comput.*, vol. 14, no. 6, pp. 865–884, Dec. 2010.

[19] T. Park and K. R. Ryu, "A dual population genetic algorithm with evolving diversity," in *Proc. IEEE Congr. Evol. Comput.*, Singapore, 2007, pp. 3516–3522.

[20] T. S. Yan, G. Q. Guo, and W. Li, "Research on a novel genetic algorithm based on adaptive evolution in dual population," in *Proc. Consumer Electron. Commun. Netw.*, XianNing, China, 2011, pp. 594–597.

[21] J. H. Zhong *et al.*, "A differential evolution algorithm with dual populations for solving periodic railway timetable scheduling problem," *IEEE Trans. Evol. Comput.*, vol. 17, no. 4, pp. 512–527, Aug. 2013.

[22] J. P. Chiou and F. Wang, "A hybrid method of differential evolution with application to optimal control problems of a bioprocess system," in *Proc. IEEE World Congr. Comput. Intell.*, Anchorage, AK, USA, 1998, pp. 627–632.

[23] J. Lampinen and I. Zelinka, "Mixed integer-discrete-continuous optimization by differential evolution," in *Proc. 5th Int. Conf. Soft Comput.*, 1999, pp. 77–81.

[24] H. A. Abbass and K. Deb, "Searching under multi-evolutionary pressures," in *Evolutionary Multi-Criterion Optimization* (Lecture Notes in Computer Science), vol. 2632. Berlin, Germany: Springer, 2003, pp. 391–404.

[25] M. T. Jensen, "Guiding single-objective optimization using multi-objective methods," in *Applications of Evolutionary Computing* (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2003, pp. 268–279.

[26] C. Segura, C. A. C. Coello, E. Segredo, G. Miranda, and C. León, "Improving the diversity preservation of multi-objective approaches used for single-objective optimization," in *Proc. IEEE Congr. Evol. Comput.*, Cancun, Mexico, 2013, pp. 3198–3205.

[27] C. Segura, C. A. C. Coello, E. Segredo, G. Miranda, and C. León, "Using multi-objective evolutionary algorithms for single-objective optimization," *4OR*, vol. 11, no. 3, pp. 201–228, 2013.

[28] H. A. Abbass, R. Sarker, and C. Newton, "PDE: A Pareto-frontier differential evolution approach for multi-objective optimization problems," in *Proc. IEEE Congr. Evol. Comput.*, Seoul, Korea, 2001, pp. 971–978.

[29] B. V. Babu and M. M. L. Jehan, "Differential evolution for multi-objective optimization," in *Proc. IEEE Congr. Evol. Comput.*, 2003, pp. 2696–2703.

[30] F. Xue, A. C. Sanderson, and R. J. Graves, "Pareto-based multi-objective differential evolution," in *Proc. IEEE Congr. Evol. Comput.*, 2003, pp. 862–869.

[31] E. Mezura-Montes, C. A. C. Coello, and E. I. Tun-Morales, "Simple feasibility rules and differential evolution for constrained optimization," in *Proc. Congr. Adv. Artif. Intell.*, Mexico City, Mexico, 2004, pp. 707–716.

[32] A. W. Iorio and X. Li, "Solving rotated multi-objective optimization problems using differential evolution," in *Proc. Congr. Adv. Art. Intell.*, Cairns, QLD, Australia, 2005, pp. 861–872.

[33] M. J. Wade, "The co-evolutionary genetics of ecological communities," *Nat. Rev. Genet.*, vol. 8, no. 3, pp. 185–195, Mar. 2007.

[34] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, 1997.

[35] R. Storn, "System design by constraint adaptation and differential evolution," *IEEE Trans. Evol. Comput.*, vol. 3, no. 1, pp. 22–34, Apr. 1999.

[36] Y. C. Lin, K. S. Hwang, and F. S. Wang, "Hybrid differential evolution with multiplier updating method for nonlinear constrained optimization problems," in *Proc. IEEE Congr. Evol. Comput.*, Honolulu, HI, USA, 2002, pp. 872–877.

[37] J. Lampinen, "A constraint handling approach for the differential evolution algorithm," in *Proc. IEEE Congr. Evol. Comput.*, Honolulu, HI, USA, 2002, pp. 1468–1473.

[38] T. P. Runarsson and X. Yao, "Search biases in constrained evolutionary optimization," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 35, no. 2, pp. 233–243, May 2005.

[39] T. P. Runarsson and X. Yao, "Stochastic ranking for constrained evolutionary optimization," *IEEE Trans. Evol. Comput.*, vol. 4, no. 3, pp. 284–294, Sep. 2000.

[40] E. Mezura-Montes, J. Velázquez-Reyes, and C. A. C. Coello, "Promising infeasibility and multiple offspring incorporated to differential evolution for constrained optimization," in *Proc. Genet. Evol. Comput. Conf.*, 2005, pp. 225–232.

[41] R. L. Becerra and C. A. C. Coello, "Culturizing differential evolution for constrained optimization," in *Proc. Mexican Int. Conf. Comput. Sci.*, Colima, Mexico, 2004, pp. 304–311.

[42] T. Takahama and S. Sakai, "Constrained optimization by the ε constrained differential evolution with gradient-based mutation and feasible elites," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2006, pp. 1–8.

[43] V. L. Huang, A. K. Qin, and P. N. Suganthan, "Self-adaptive differential evolution algorithm for constrained real-parameter optimization," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2006, pp. 17–24.

[44] M. F. Tasgetiren and P. N. Suganthan, "A multi-populated differential evolution algorithm for solving constrained optimization problem," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2006, pp. 33–40.

[45] S. Kukkonen and J. Lampinen, "Constrained real-parameter optimization with generalized differential evolution," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2006, pp. 207–214.

[46] J. Brest, V. Zumer, and M. S. Maucec, "Self-adaptive differential evolution algorithm in constrained real-parameter optimization," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2006, pp. 215–222.

[47] E. Mezura-Montes, J. Velázquez-Reyes, and C. A. C. Coello, "Modified differential evolution for constrained optimization," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2006, pp. 332–339.

[48] K. Zielinski and R. Laur, "Constrained single-objective optimization using differential evolution," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2006, pp. 927–934.

[49] E. Mezura-Montes and B. Cecilia-López-Ramírez, "Comparing bioinspired algorithms in constrained optimization problems," in *Proc. IEEE Congr. Evol. Comput.*, Singapore, 2007, pp. 662–669.

[50] W. Gong and Z. Cai, "A multiobjective differential evolution algorithm for constrained optimization," in *Proc. IEEE Congr. Evol. Comput.*, Hong Kong, 2008, pp. 181–188.

[51] T. Takahama and S. Sakai, "Constrained optimization by ε constrained differential evolution with dynamic ε-level control," in *Advances in Differential Evolution*, U.K. Chakraborty, Ed. Berlin, Germany: Springer, 2008, pp. 139–154.

[52] M. Zhang, W. Luo, and X. Wang, "Differential evolution with dynamic stochastic selection for constrained optimization," *Inf. Sci.*, vol. 178, no. 15, pp. 3043–3074, 2008.

[53] K. Zielinski and R. Laur, "Stopping criteria for differential evolution in constrained single-objective optimization," in *Advances in Differential Evolution*, U.K. Chakraborty, Ed. Berlin, Germany: Springer, 2008, pp. 111–138.

[54] X. Yuan *et al.*, "A chaotic hybrid cultural algorithm for constrained optimization," in *Proc. IEEE Int. Conf. Genet. Evol. Comput.*, Hubei, China, 2008, pp. 307–310.

[55] Y. Wang, Z. Cai, G. Guo, and Y. Zhou, "Multiobjective optimization and hybrid evolutionary algorithm to solve constrained optimization problems," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 37, no. 3, pp. 560–575, Jun. 2007.

[56] Y. Wang, Z. Cai, G. Guo, and Y. Zhou, "A dynamic hybrid framework for constrained evolutionary optimization," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 1, pp. 203–217, Feb. 2012.

[57] R. Mallipeddi and P. N. Suganthan, "Ensemble of constraint handling techniques," *IEEE Trans. Evol. Comput.*, vol. 14, no. 4, pp. 561–579, Aug. 2010.

[58] J. J. Liang *et al.*, "Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization," School Elect. Electron. Eng., Nanyang Technol. Univ., Singapore, Tech. Rep., Sep. 2006.

[59] Y. Wang, Z. Cai, Y. Zhou, and W. Zeng, "An adaptive tradeoff model for constrained evolutionary optimization," *IEEE Trans. Evol. Comput.*, vol. 12, no. 1, pp. 80–92, Feb. 2008.

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

[61] G. Jia, Y. Wang, Z. Cai, and Y. Jin, "An improved (μ+λ)-constrained differential evolution for constrained optimization," *Inform. Sci.*, vol. 222, no. 10, pp. 302–322, Feb. 2013.

[62] B. Liu, H. N. Ma, X. J. Zhang, B. Liu, and Y. Zho, "A memetic co-evolutionary differential evolution algorithm for constrained optimization," in *Proc. IEEE Congr. Evol. Comput.*, Singapore, 2007, pp. 2996–3002.

[63] K. Tagawa and K. Nakajima, "Island-based differential evolution with panmictic migration for multi-core CPUs," in *Proc. IEEE Congr. Evol. Comput.*, Cancun, Mexico, 2013, pp. 852–859.

[64] E. Mezura-Montes and C. A. C. Coello, "A simple multimembered evolution strategy to solve constrained optimization problems," *IEEE Trans. Evol. Comput.*, vol. 9, no. 1, pp. 1–17, Feb. 2005.

[65] Y. G. Woldesenbet and G. G. Yen, "Dynamic evolutionary algorithm with variable relocation," *IEEE Trans. Evol. Comput.*, vol. 13, no. 3, pp. 500–513, Jun. 2009.

[66] Y. G. Woldesenbet, G. G. Yen, and B. G. Tessema, "Constraint handling in multi-objective evolutionary optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 3, pp. 514–525, Jun. 2009.

[67] G. G. Yen and W. F. Leong, "Dynamic multiple swarms in multiobjective particle swarm optimization," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 39, no. 4, pp. 890–911, Jul. 2009.

[68] K. M. Woldemariam and G. G. Yen, "Vaccine enhanced artificial immune system for multimodal function optimization," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 40, no. 1, pp. 218–228, Feb. 2010.

[69] W. W. Zhang, G. G. Yen, and Z. S. He, "Constrained optimization via artificial immune system," *IEEE Trans. Cybern.*, vol. 44, no. 2, pp. 185–198, Feb. 2014.

Wei-feng Gao received the Ph.D. degree in applied mathematics from the School of Science of Xidian University, Xi'an, China, in 2013.

He is currently a Lecturer with the China University of Petroleum, Qingdao, China. His current research interests include evolutionary algorithms and their applications in the real world. He has published several papers in the IEEE TRANSACTIONS ON CYBERNETICS and *Information Sciences*.

Gary G. Yen (S'87–M'88–SM'97–F'09) received the Ph.D. degree in electrical and computer engineering from the University of Notre Dame, Notre Dame, IN, USA, in 1992.

He is currently a Professor with the School of Electrical and Computer Engineering, Oklahoma State University, Stillwater, OK, USA. His current research interests include intelligent control, computational intelligence, conditional health monitoring, signal processing, and their industrial/defense applications.

Prof. Yen was an Associate Editor of the IEEE CONTROL SYSTEMS MAGAZINE, the IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY, the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART A: SYSTEMS AND HUMANS, the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART B: CYBERNETICS, the IEEE TRANSACTIONS ON NEURAL NETWORKS, *Automatica*, and *Mechantronics*. He is currently serving as an Associate Editor for the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and *International Journal of Swarm Intelligence Research*. He also served as a General Chair for the 2003 IEEE International Symposium on Intelligent Control held in Houston, TX, USA, and the 2006 IEEE World Congress on Computational Intelligence held in Vancouver, BC, Canada. He served as the Vice President for the Technical Activities in 2005–2006, and the President in 2010–2011, of the IEEE Computational intelligence Society, and was the Founding Editor-in-Chief of the IEEE COMPUTATIONAL INTELLIGENCE MAGAZINE in 2006–2009. In 2011, he received the Andrew P. Sage Best Transactions Paper Award from the IEEE Systems, Man, and Cybernetics Society. In 2013, he received Meritorious Service Award from the IEEE Computational Intelligence Society. He is a fellow of the IET.

San-yang Liu received the Ph.D. degree in computational mathematics from Xi'an Jiaotong University, Xi'an, China, in 1989.

He is currently a Professor and the Dean of the School of Science of Xidian University, Xi'an. His current research interests include nonlinear optimization, combinatorial optimization, network optimization, and system reliability.