# A Multiobjective Optimization-Based Evolutionary Algorithm for Constrained Optimization

Zixing Cai, *Senior Member, IEEE*, and Yong Wang

***Abstract*—A considerable number of constrained optimization evolutionary algorithms (COEAs) have been proposed due to increasing interest in solving constrained optimization problems (COPs) by evolutionary algorithms (EAs). In this paper, we first review existing COEAs. Then, a novel EA for constrained optimization is presented. In the process of population evolution, our algorithm is based on multiobjective optimization techniques, i.e., an individual in the parent population may be replaced if it is dominated by a nondominated individual in the offspring population. In addition, three models of a population-based algorithm-generator and an infeasible solution archiving and replacement mechanism are introduced. Furthermore, the simplex crossover is used as a recombination operator to enrich the exploration and exploitation abilities of the approach proposed. The new approach is tested on 13 well-known benchmark functions, and the empirical evidence suggests that it is robust, efficient, and generic when handling linear/nonlinear equality/inequality constraints. Compared with some other state-of-the-art algorithms, our algorithm remarkably outperforms them in terms of the best, mean, and worst objective function values and the standard deviations. It is noteworthy that our algorithm does not require the transformation of equality constraints into inequality constraints.**

***Index Terms*—Constrained optimization, evolutionary algorithm (EA), multiobjective optimization, nondominated individuals.**

## I. Introduction

IN MANY science and engineering disciplines, it is not uncommon to face a large number of constrained optimization problems (COPs). Without loss of generality, the general nonlinear programming (NLP) problem that we are interested in can be formulated as

$$\text{Find } \vec{x}(\vec{x}=(x_1,x_2,\ldots,x_n)\in\Re^n) \quad \text{such that } f(\vec{x}) \text{ is optimized}$$

where $\vec{x}\in\Omega\subseteq S$, and $S$ is an $n$-dimensional rectangle space in $\Re^n$ defined by the parametric constraints

$$l(i)\le x_i\le u(i),\quad 1\le i\le n.$$

Manuscript received April 18, 2005; revised September 13, 2005, November 12, 2005, and January 11, 2006. This work was supported in part by the National Natural Science Foundation of China under Grant 60234030 and Grant 60404021, in part by the National Basic Scientific Research Founds under Grant A1420060159, and in part by Hunan S&T Founds under Grant 05IJY3035.

The authors are with the College of Information Science and Engineering, Central South University, Changsha, Hunan 410083, China (e-mail: zxcai@ieee.org; wangyong1226@yahoo.com).

Digital Object Identifier 10.1109/TEVC.2006.872344

The feasible region $\Omega\subseteq S$ is defined by a set of $m$ additional linear or nonlinear constraints $(m\ge 0)$

$$g_j(\vec{x})\le 0,\quad j=1,\ldots,q \quad\text{and}\quad h_j(\vec{x})=0,\quad j=q+1,\ldots,m$$

where $q$ is the number of inequality constraints and $m-q$ is the number of equality constraints.

For an inequality constraint that satisfies $g_j(\vec{x})=0(j\in\{1,\ldots,q\})$ at any point $\vec{x}\in\Omega$, we say it is *active* at $\vec{x}$. All equality constraints $h_j(\vec{x})(j=q+1,\ldots,m)$ are considered active at all points of $\Omega$.

Recently, as a search and optimization technique inspired by nature, evolutionary algorithms (EAs) have been broadly applied to solve COPs ([1]–[3]). Although many methods have been proposed to handle constraints by EAs for parameter optimization problems, experimental results actually reveal that most of them do not have general capability in handling various constrained optimization problems, e.g., optimization problems with equality constraints, inequality constraints, or mixed constraints.

There is no formal guarantee of an algorithm's general effectiveness if insufficient knowledge of the problem characteristics is incorporated into the algorithm domain according to the no-free-lunch theorems ([4], [5]). However, optimization problems in nature have certain pertinence with each other rather than total independence. Thus, we are faced with the difficulty of how to explore knowledge in order to design effective and efficient algorithms in a specific domain.

The current study proposes a novel EA based on multiobjective optimization techniques for COPs. In the new approach, a given COP is converted into a biobjective optimization problem. Meanwhile, the nondominated individuals replacement scheme is devised in terms of the converted fitness. Three models of a population-based algorithm-generator are introduced on the basis of different replacement and comparison rules. The aim of these models is to guide the search toward the global optima of COPs. To accelerate the convergence speed, an infeasible solution archiving and replacement mechanism (ISARM), which helps the population to approach or land in the feasible region of the search space from different directions rapidly is added. Simplex crossover is used to improve the exploration and exploitation capabilities of our algorithm. With these combined elements, we show that the proposed algorithm has advantages over other algorithms compared in many indicators of the experimental results. The main contribution of our approach is that it can reach the global feasible optima of all test cases without the transformation of equality constraints into inequality constraints.

The remainder of this paper is organized as follows. Section II reviews the related work of handling COPs via EAs. Section III presents a detailed description of the proposed algorithm. In Section IV, our algorithm with six state-of-the-art algorithms is used to produce experimental results for the chosen test functions. Several performance metrics are examined to compare the performance of our algorithm against the chosen algorithms. Finally, Section V provides some concluding remarks.

## II. Previous Work

Michalewicz and Schoenauer [1] and Coello [2] provided a comprehensive survey of the most popular constraint-handling techniques currently used with EAs and grouped them into four and five categories, respectively. As stated in [2], the constraint-handling techniques can be divided into: 1) penalty functions; 2) special representations and operators; 3) repair algorithms; 4) separate objective and constraints; and 5) hybrid methods.

Penalty function methods are among the most common methods for solving COPs. The principal idea of this kind of methods is to reformulate a constrained optimization problem as an unconstrained one by introducing a penalty term into the original objective function to penalize constraint violations. In general, a penalty term is based on the distance of an individual from the boundaries of the feasible set. The distance of an individual $\vec{x}$ from the $j$th constraint can be constructed as

$$G_j(\vec{x}) = \begin{cases} \max\{0, g_j(\vec{x})\}, & 1 \le j \le q \\ |h_j(\vec{x})|, & q+1 \le j \le m \end{cases}.$$

Let $G(\vec{x}) = \sum_{j=1}^{m} G_j(\vec{x})$ denote the distance of the individual $\vec{x}$ from the boundaries of the feasible set, which also reflects the degree of its constraint violation.

In this kind of methods, the static penalty refers to the penalty term that increases with the degree of constraint violation, and the dynamic penalty is the penalty term that increases with both the degree of constraint violation and generation number. In addition, annealing penalties and death penalties are also used in some methods to deal with COPs.

Le Riche *et al.* [6] proposed a segregated genetic algorithm. Two different penalized fitness functions $f_1$ and $f_2$ are designed with two static penalty terms $p_1$ and $p_2$, one is smaller and the other is larger. Individuals will be more likely to lie within the infeasible region when selected on the basis of $f_1$. On the contrary, the ones selected on the basis of $f_2$ will more likely stay in the feasible region. During the search process, the feasible optimum will be reached from both sides of the feasible region boundaries.

Adaptive constraint-handling methods ([7], [8]) are very promising for constrained optimization, since they can make use of information obtained during the search to adjust their own parameters.

Farmani and Wright [9] presented a self-adaptive fitness formulation method, an enhanced version of the algorithm in [10], where the penalty is divided into two stages. The improved approach eliminates the fixed weight for the second penalty stage proposed in [10], by assigning the penalized objective function value of the worst individual to be equal to that of the individual with maximum objective function value in the current population. This makes the method self-adaptive and more dynamic due to the fact that the individual with maximum objective function value may vary from one generation to another. However, to find the optimum or the approximate optimum, considerable computational effort would be required.

A remarkable limitation of the penalty function methods is that most of them require a careful fine-tuning of parameters to obtain competitive results. A too small penalty parameter results in underpenalization, and consequently, the population will experience difficulty in landing within the feasible region and may converge to an infeasible solution. Instead, a too large penalty coefficient will result in the loss of some valuable information provided by infeasible individuals. Even the most dynamic setting methods, which start with a low parameter value and end up with a high one, are unlikely to work well for problems where the unconstrained global optimum is far away from the constrained one [11]. Further, these methods lack generality and are usually only fit for optimization problems with certain constraint types.

The homomorphous mapping approach [12] converts COPs into unconstrained optimization problems by using a mapping between an $n$-dimensional cube and the feasible space of the given problem. This approach has many advantages over other constraint-handling techniques. For example, it does not require any special operators to maintain feasible solutions and it does not need to evaluate infeasible solutions. However, it also exhibits a number of drawbacks. First, the implementation of this method is very difficult especially for nonconvex feasible search spaces. Second, it requires initial feasible solutions.

In recent years, emphasis has been increasingly placed on COEAs based on multiobjective concepts ([13]–[28]). The main idea of these algorithms is to treat constraints as one or more objectives. In this paper, we classify the methods based on multiobjective concepts into two categories: 1) methods based on biasing feasible over infeasible solutions; and 2) methods based on multiobjective optimization techniques. We next discuss them in turn.

The first category usually redefines a single objective optimization problem in such a manner that two objectives are considered: the first is to optimize the original objective function $f(\vec{x})$, and the second is to minimize the degree of constraint violation $G(\vec{x})$.

Inspired by Powell and Skolnick [13], Deb [14] proposed an alternative method that requires no penalty parameters. This method uses a tournament selection operator, where pair-wise solutions are compared at a time using the following criteria: 1) any feasible solution is preferred to any infeasible solution; 2) between two feasible solutions, the one with better objective function value is preferred; and 3) between two infeasible solutions, the one with smaller degree of constraint violation is preferred. The main drawback of this scheme is that it is hard to maintain a reasonable proportion of infeasible and feasible solutions in the population. These or similar criteria are also adopted to compare individuals in [15] and [16].

After systematically analyzing the pitfalls of the penalty function methods, Runarsson and Yao [17] introduced a stochastic ranking method to balance the aforementioned two objectives.

A probability parameter $p_f$ is involved to compare individuals as follows: given pairwise adjacent individuals, 1) if both individuals are feasible, the one having better objective function value wins, else 2) if a uniformly generated random number $u$ between 0 and 1 is less than $p_f$, the one with better objective function value wins, otherwise, the one with smaller degree of constraint violation wins. This approach significantly improves the search performance without any specialized operators. But empirical evidence shows that it is sensitive to the parameter $p_f$. Recently, a further exploration has been done by the same authors [18].

As for the second category of the multiobjective concepts based methods, i.e., the methods based on multiobjective optimization techniques, the main idea is to convert COPs into multiobjective optimization problems (MOPs) that have $m+1$ objectives, where $m$ is the number of constraints. Then, we can apply any multiobjective optimization techniques to new vector $\vec{v} = (f(\vec{x}), f_1(\vec{x}), \ldots, f_m(\vec{x}))$, where $f_1(\vec{x}), \ldots, f_m(\vec{x})$ are constraints of the given problem. Several representative approaches in this area are reviewed next.

Coello [19] used a population-based multiobjective technique such as Vector Evaluated Genetic Algorithm (VEGA) [20] to treat each constraint as an objective. In each generation, the population is split into $m+1$ subpopulations with equal size, where $m$ refers to the number of constraints (they add one to consider also the objective function). In this approach, some individuals of the population are selected using the objective function as their fitness, and the remaining individuals are selected by applying the corresponding constraints as their fitness. For the subpopulation guided by the objective function, the evaluation of such objective function for a given vector $\vec{x}$ is used directly as the fitness function. For all the other subpopulations, the algorithm used in [19] was the following: if $g_j(\vec{x}) < 0$, then fitness = $-g_j(\vec{x})$; else if $v \neq 0$, then $\text{fitness} = -v$; else $\text{fitness} = -f(\vec{x})$, where $g_j(\vec{x})$ refers to the constraint corresponding to the $j+1$ subpopulation and $v$ refers to the number of constraints that are violated. However, since the size of each subpopulation should not be fixed, how to determine the appropriate size of each subpopulation remains an open issue.

Coello and Mezura [21] implemented a version of the Niched-Pareto Genetic Algorithm (NPGA) [22]. Unlike NPGA, this approach does not require niching. Instead, it uses an additional parameter called $S_r$ to control the diversity of the population. The selection ratio $S_r$ denotes the minimum number of individuals that are selected through dominance-based tournament selection. The remaining $(1-S_r)$ individuals are selected by a purely probabilistic approach. During the dominance-based tournament selection, four comparison rules are involved as follows: 1) when both individuals are feasible, the individual with better fitness value is preferred; 2) any feasible individual is preferred to any infeasible individual; 3) when both are infeasible, the nondominated individual is selected, only if the other candidate is dominated; and 4) when both are infeasible and both are either nondominated or dominated, the individual with the lower amount of constraint violation wins. However, this selection procedure also faces the defects of Deb's tournament selection [14].

Besides, the method [23] based on Fonseca and Fleming's Pareto ranking process [24], and the method [25] based on the combination of VEGA and Pareto ranking are also developed to handle COPs.

Aguirre *et al.* [26] proposed an alternative approach, which is an extension of the Pareto Archived Evolutionary Strategy (PAES) [27] and uses Pareto dominance as the selection criterion. In this method, two crucial features are introduced, i.e., inverted "ownership" and shrinking the objective space. In the process of evolution, the search space is shrunk continuously through exploring the information of the individuals surrounding the feasible region. Thus, the size of the search space will be very small and the solutions obtained will be competitive in the end. However, the implementation of this method is very complex. Moreover, once the decrease of the search space is directed toward false direction, the algorithm might be trapped in a local optimum.

In order to evaluate the capabilities of a set of constraint-handling methods based on multiobjective optimization techniques, Mezura and Coello [28] presented a comprehensive experimental study, in particular, focusing on four of this kind of methods. The experimental results indicate that the selection criterion of Pareto dominance gives better results than both Pareto ranking and population-based approach. On the other hand, an important conclusion in [28] is that additional mechanisms have to be used to improve the effectiveness of these approaches.

## III. Proposed Approach

As multiobjective evolutionary algorithms (MOEAs) have two goals (convergence to the true Pareto optimal set, and maintenance of a uniform distribution of the Pareto front), COEAs also have two definite objectives: 1) landing in or approaching the feasible domain promptly, and 2) reaching the global optimal solution in the end. However, these ultimate goals are far from being accomplished by the existing COEAs according to empirical evidence. One of the main deficiencies of the existing COEAs lies in designing a suitable scheme to compare and select individuals, which is vital for the achievement of the second objective. Again, unlike unconstrained optimization problems, the search space of COPs is composed of the feasible and infeasible regions. As a result, how to effectively utilize infeasible solutions becomes very important, which also has a notable impact on the realization of the first objective, especially when the optimum is located on the boundaries of the feasible region or the ratio of the feasible region is very small compared with the entire search space.

Usually, in order to find feasible solutions, an equality constraint is replaced by pairwise inequality constraints: $h(\vec{x}) - \delta \leq 0$ and $-h(\vec{x}) - \delta \leq 0$, where $\delta$ is a small positive value. Note that this conversion changes the topology structure of the feasible set. Obviously, if the structure of the feasible set changes, then the optimum maybe changes accordingly and, consequently, this conversion will directly affect the capability of the approach, as well as the accuracy of the solution found. Furthermore, such effects may arise provided this conversion exists, although a dynamic tolerance value has been used in some literature ([7], [16]).

Motivated by these considerations, we present a novel approach based on multiobjective optimization techniques. The new approach can find the true feasible optima of 13 well-known

benchmark functions chosen from [17] and relaxes the need to transform equality constraints. It recasts COPs as biobjective optimization problems to minimize the original objective function $f(\vec{x})$ and the degree of constraint violation $G(\vec{x})$ simultaneously. For the sake of clarity, let $\mathbf{f}(\vec{x}) = (f(\vec{x}), G(\vec{x}))$. Without loss of generality, minimization problems are assumed in this paper.

### A. Difference Between $\mathbf{f}(\vec{x})$ and the General MOPs

In fact, the main purpose of changing a COP into a MOP is to avoid using the penalty function. Despite the fact that a COP can be converted into a biobjective optimization problem $\mathbf{f}(\vec{x})$, still an essential difference remains with the general MOPs. That is, the philosophy of the general MOPs is obtaining a final population with a diversity of nondominated individuals, whereas $\mathbf{f}(\vec{x})$ would retrogress into a single objective optimization problem $f(\vec{x})$ within the feasible region (because in this case $G(\vec{x}) = 0$), thus it still has only one global optimum and, consequently, there is no need to care about the uniform distribution of the resulting solutions which is one of the goals of MOEAs, when optimizing $\mathbf{f}(\vec{x})$. So the solution of $\mathbf{f}(\vec{x})$ is not equivalent to that of the general MOPs.

*Definition 1 (Global Minimum):* $\vec{x}^* \in S$ is called global minimum if and only if $G(\vec{x}^*) = 0$ and $\neg\exists \vec{x} \in S$ such that $G(\vec{x}) = 0$ and $f(\vec{x}) \leq f(\vec{x}^*)$.

### B. Nondominated Individuals Replacement Scheme

Unlike the existing COEAs, the fundamental idea of this paper aims at: 1) taking advantage of multiobjective optimization techniques to extract the main information contained in the current population and 2) directing the population to evolve according to that valuable information. We next introduce four essential definitions regarding multiobjective optimization in the context of our approach.

*Definition 2 (Pareto Dominance):* A vector $\vec{u} = (u_1, \ldots, u_k)$ is said to Pareto dominate another vector $\vec{v} = (v_1, \ldots, v_k)$, denoted as $\vec{u} \prec \vec{v}$, if

$$\forall i \in \{1, \ldots k\}, u_i \leq v_i \quad \text{and} \quad \exists j \in \{1, \ldots, k\}, u_j < v_j.$$

*Definition 3 (Pareto Optimality):* $\vec{x}_u \in S$ is said to be Pareto optimal (in $S$) if and only if $\neg\exists \vec{x}_v \in S, \vec{v} \prec \vec{u}$, where $\vec{v} = \mathbf{f}(\vec{x}_v) = (v_1, v_2), \vec{u} = \mathbf{f}(\vec{x}_u) = (u_1, u_2)$.

*Definition 4 (Pareto Optimal Set):* The Pareto optimal set, denoted as $\rho^*$, is defined as

$$\rho^* = \{\vec{x}_u \in S | \neg\exists \vec{x}_v \in S, \vec{v} \prec \vec{u}\}.$$

The vectors included in the Pareto optimal set are called nondominated individuals.

*Definition 5 (Pareto Front):* According to the Pareto optimal set, the Pareto front, denoted as $\rho f^*$, is defined as

$$\rho f^* = \{\vec{u} = \mathbf{f}(\vec{x}_u) | \vec{x}_u \in \rho^*\}.$$

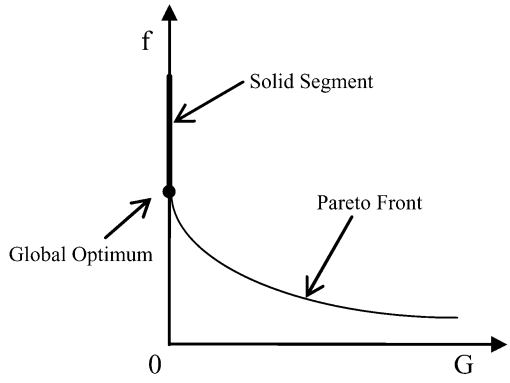



Fig. 1. Graph representation for $\mathbf{f}(x)$. The Pareto optimal set $\rho^*$ is mapped to the Pareto front $\rho f^*$. The feasible region $\Omega$ is mapped to the solid segment. The global optimum $x^*$ is mapped to the intersection of the Pareto front and the solid segment. The search space $S$ is mapped to points on and above the Pareto front.

Clearly, the Pareto front is the image of the Pareto optimal set in the objective space.

Since COPs are transformed to MOPs, the definition of Pareto optimality is considered with respect to the decision variable space $S$ instead of the feasible region $\Omega$. Based on the above definitions, a more profound illustration of $\mathbf{f}(\vec{x})$ is given in Fig. 1 [29].

In MOEAs, each individual in the population is associated with a rank. Moreover, all nondominated individuals in the population have the same rank value. For instance, the rank value of nondominated individuals is assigned to 1 in [24] and [30], but to 0 in [31]. In contrast to MOEAs, since nondominated individuals represent the most important feature of the population they belong to, our concern in this paper is only the nondominated individuals. Thus, we can avoid assigning a rank value to each individual in the population, which makes the new algorithm more efficient. The complexity of identifying all nondominated individuals in a population is $O(N\bar{N})$, where $N$ is the population size, $\bar{N}$ is the number of nondominated individuals in the population.[1]

Fig. 2(a) shows an example for illustrating the importance of nondominated individuals. For example, there are three nondominated individuals denoted as "e," "f," and "g" in a population. It can be found that individual "e" denotes the best feasible solution, individual "f" denotes the infeasible solution with the lowest degree of constraint violation, and individual "g" denotes the infeasible solution with the minimum objective function value, therefore the most important information of a population is clearly represented by nondominated individuals.

As for nondominated individuals, their characteristics can be summarized as follows.

*Theorem 1:* There exists at most one feasible solution in nondominated individuals in a population.

*Proof:* Assume that there are $k(k > 1)$ feasible solutions in nondominated individuals in a population. Then, the feasible

[1]Although Deb [30] described a fast nondominated sorting approach which requires $O(MN^2)$ comparisons, the storage requirement has increased to $O(MN^2)$. In this paper, the aim here is to identify the first nondominated front, therefore $O(N)$ storage is required.

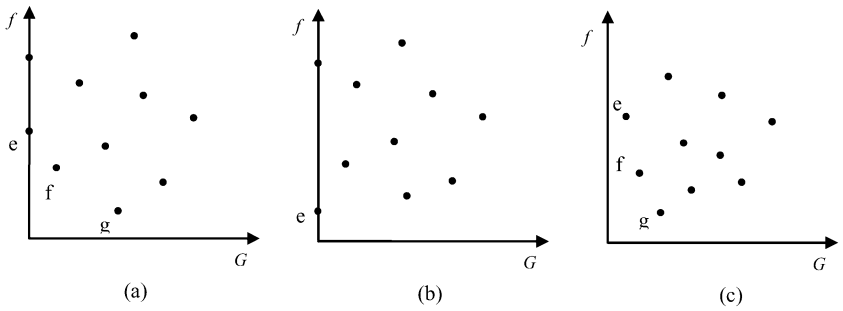



Fig. 2. (a) Nondominated individuals in a population consist of a combination of feasible and infeasible solutions. (b) Nondominated individuals in a population consist of one feasible solution. (c) Nondominated individuals in a population consist of infeasible solutions.

solution with the minimum $f(\vec{x})$ in these $k$ individuals must dominate other feasible solutions in these $k$ individuals based on Pareto dominance. This is a contradiction.

Theorem 1 implies that in terms of the whole search space, the feasible individual in nondominated individuals is just the global optimum (see Fig. 1).

*Property 1:* Nondominated individuals in a population may consist of one feasible solution [Fig. 2(b)], infeasible solutions [Fig. 2(c)], or a combination of feasible and infeasible solutions [Fig. 2(a)].

*Property 2:* Feasible solutions of nondominated individuals in the offspring population can dominate either feasible or infeasible solutions in the parent population. However, infeasible solutions of nondominated individuals in the offspring population can only dominate infeasible solutions in the parent population.

Property 2 can be obtained from the definition of Pareto dominance.

After all nondominated individuals are selected from the offspring population; they are then applied to replace individuals in the parent population dominated by them.

### C. Models of a Population-Based Algorithm-Generator

As a class of EAs, evolutionary strategy (ES) has been attached more importance to solve COPs ([7], [16], [17]). Two main reasons for this are: 1) there is a theoretical background that supports the ES convergence and 2) ES's original self- adaptation mechanism helps itself to deal with constrained search spaces ([32], [33]). On the whole, experimental results reveal that with the same constraint-handling technique the overall capability of ES is better than GA [34].

In order to test the effectiveness of ES, Mezura and Coello [32] implemented different types of ES with three simple comparison criteria and demonstrated that ES is not effective with respect to the test functions with high dimension, large feasible regions, or many nonlinear equality constraints. Meanwhile, an essential limitation of ES is that ES usually neglects the crossover operator of the decision variables, whereas the crossover operator is very important for constrained optimization, especially when it occurs between feasible and infeasible individuals close to the border of the feasible region. Although discrete and intermediate crossover operators can be employed [16], their abilities are limited.

Additionally, real-coded genetic algorithm (RCGA) has been proposed to overcome the drawbacks of the binary string representation in traditional GA. In RCGA, crossover is the principal operator for search, which utilizes the information on several parents to yield new offspring adaptively according to the distribution of the parents. Kita [35], by comparing the self-adaptive ES with the RCGA with unimodal normal distribution crossover (UNDX) operator, pointed out that the RCGA is more suitable to optimize multimodal and high dimension functions.

**Step 1** Choose $\mu$ solutions (the set $Q$) from $B$ using a *selection plan* (SP).
**Step 2** Create $\lambda$ solutions (the set $C$) from $Q$ using a *generation plan* (GP).
**Step 3** Choose $\gamma$ solutions (the set $R$) from $B$ for replacement using a *replacement plan* (RP).
**Step 4** Update these $\gamma$ solutions by $\gamma$ solutions chosen from a comparison set of $R$, $Q$ and $C$, using an *update plan* (UP).

Fig. 3. Population-based algorithm-generator for real-parameter optimization.

In RCGA, besides crossover operator, population evolution model also has a significant influence on the capability of optimization. Recently, Deb [36] proposed a population-based algorithm-generator for real-parameter optimization, which divides the task of searching the optimum into four independent plans: 1) selection plan; 2) generation plan; 3) replacement plan; and 4) update plan. The primary advantage of this algorithm-generator is the functional decomposition of these four important plans. It is described in Fig. 3. Note that the signs (i.e., $\mu, \lambda, \gamma, R, Q$, and $C$) in Fig. 3 do not have any special meaning.

Inspired by Deb's algorithm-generator, we propose two computation models (i.e., Models 1 and 2) on the basis of the nondominated individuals replacement scheme described in Section III-B. The difference between these two models is that Model 1 uses all information provided by nondominated individuals, while Model 2 only uses partial information provided by nondominated individuals. The pseudocode of Model 1 is shown in Fig. 4. Model 2 is different from Model 1 only in steps 3 and 4; the steps 3 and 4 of Model 2 are depicted in Fig. 5.

There are two differences between Deb's algorithm-generator and the two proposed computation models. First, in Deb's algorithm-generator, the aim of step 3 is to choose some individuals to be replaced by the individuals produced by step 4. On the contrary, the purpose of step 3 in our computation models is to choose some individuals to replace the individuals produced by step 4. Second, in Deb's algorithm-generator the individuals to

**Step 1** Choose $\mu$ solutions (the set $Q$) from the set $B$ randomly, thereafter $B = B - Q$

**Step 2** Create $\lambda$ offspring solutions (the set $C$) from the set $Q$ using simplex crossover (SPX, see Section III-F for details).

**Step 3** Choose the nondominated individuals (the set $R$) from the set $C$. Assume there exist $m'$ nondominated individuals, denoted as $\vec{x}_1, \cdots, \vec{x}_{m'}$.

**Step 4** Suppose that there are $n'$ individuals in the set $Q$ dominated by $\vec{x}_1$.

**if** $n' = 0$ **then**
no replacement will happen;
**else if** $n' = 1$ **then**
the corresponding dominated individual will be replaced by $\vec{x}_1$;
**else** /*it means $n' > 1$ */
**if** the dominated individuals are feasible **then**
the individuals with the maximum original objective function value will be replaced by $\vec{x}_1$;
**else**
one of the dominated individuals chosen at random, will be replaced by $\vec{x}_1$;
**end if**
**end if**
Thereafter, the remaining nondominated individuals of the set $R$ will continue the same process for the updated set $Q$ in turn.

**Step 5** $B = B \bigcup Q$.

Fig. 4. Pseudocode of Model 1.

**Step 3** Randomly choose one individual from the set $R$ that contains all nondominated individuals in the set $C$, denoted as $\vec{x}_1$.

**Step 4** Suppose that there are $n'$ individuals of the set $Q$ dominated by $\vec{x}_1$.

**if** $n' = 0$ **then**
no replacement will happen;
**else if** $n' = 1$ **then**
the corresponding dominated individual will be replaced by $\vec{x}_1$;
**else** /*it means $n' > 1$ */
**if** the dominated individuals are feasible **then**
the individual with the maximum original objective function value will be replaced by $\vec{x}_1$;
**else**
one of the dominated individuals chosen at random, will be replaced by $\vec{x}_1$;
**end if**
**end if**

Fig. 5. Steps 3 and 4 of Model 2.

be replaced are chosen from the set $B$, however, from the set $Q$ in our computation models. Indeed, Models 1 and 2 are similar to $(\mu+(m',\lambda))$ ES and $(\mu+(1,\lambda))$ ES, respectively. Note that, $n'$ and $m'$ are dynamic in the process of population evolution. An explanation of the implementation process of Models 1 and 2 is shown in Fig. 6.

Both models have the ability to coordinate exploration and exploitation.

1) The variance in the population provides good information about the extent of the potential search region. At the early stage of the search, through randomly selecting several individuals in the population, the variance among individuals is expected to be large. Therefore, the creation of several individuals from the chosen individuals results in exploring a wider region. On the other hand, the variance in the population is expected to be small, thereby ensuring a focused search near the optimum, which occurs at the later stage when the individuals in the population have converged near the optimum.
2) Nondominated individuals in the offspring population are chosen and replace dominated individuals of the parent population dynamically (the parent population corresponds to the set $Q$ and the offspring population corresponds to the set $C$). This ensures the exploitation of the global information during the early stage and of the local information during the later stage.

### *D. Infeasible Solution Archiving and Replacement Mechanism (ISARM)*

A nontrivial difference between unconstrained optimization problems and COPs is that the latter's search space comprises of the feasible and infeasible parts. Thereby, the proportion of these two parts, the types of constraints (linear/nonlinear, or equality/inequality), and the location of the optimum (on the boundaries of the feasible region or within the feasible region) concurrently add difficulties to the search of the global optimum.

Researchers have gradually realized the effects of infeasible solutions on finding the global optimum in the feasible region. Farmani and Wright [9] formulated a method to ensure that slight infeasible solutions with a low objective function value remain fit. Mezura and Coello [16] and Coello and Mezura [21] tended to bring into play the function of infeasible solutions by a diversity mechanism.

Nevertheless, unlike the above approaches, a new proposal, called ISARM, is introduced here to attack the first objective of COEAs, namely, efficiently guiding the population toward feasibility. The main principle of our mechanism is that, provided the current offspring population is composed of only infeasible individuals, the "best" of infeasible individuals in the current offspring population, who has the lowest degree of constraint violation is stored into a predefined archive $A$. Then, after a fixed interval of generations, some randomly selected individuals of the main population $P$ (the population $P$ corresponds to the set $B$ in Section III-C) are replaced by the same number of randomly selected infeasible individuals in the archive $A$. This process is described in Fig. 7.

With respect to COPs with a large proportion of the feasible region, the proposed mechanism seems to play a minor role in finding the optimal solution. Since the initial population contains a large number of feasible solutions in this case, the archiving and replacement of infeasible individuals do not happen unless the offspring population is composed of only infeasible individuals, but the likelihood of such case occurring is very small.

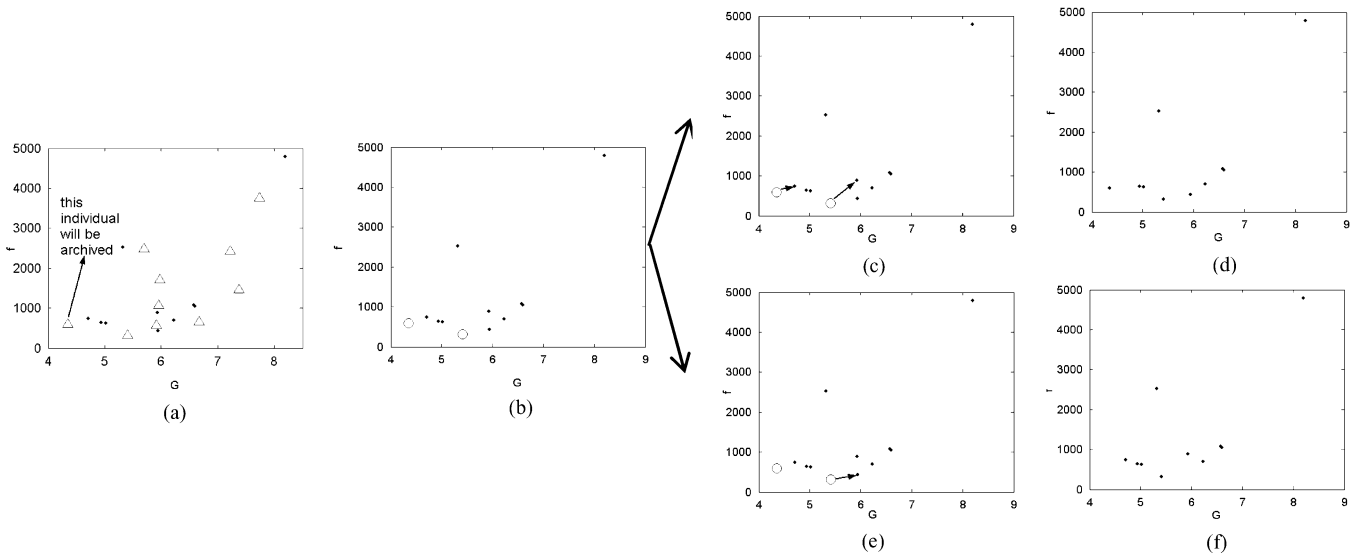



Fig. 6. An implementation process of Models 1 and 2 when nondominated individuals consist entirely of infeasible solutions. " . " denotes the parent, "∧" denotes the offspring, and "o" denotes the nondominated individual in the offspring population. (a)–(d) describe the implementation process of Model 1: (a) create ten offspring from nine parents using simplex crossover (step 2 in Fig. 4); (b) choose nondominated individuals from the ten offspring (step 3 in Fig. 4); (c) use the nondominated individuals to replace the dominated individuals in the parent population (step 4 in Fig. 4); and (d) get a new population (step 5 in Fig. 4). Meanwhile, (a), (b), (e), and (f) describe the implementation process of Model 2: (e) use one of the nondominated individuals to replace the dominated individual in the parent population (step 4 in Fig. 5) and (f) get a new population.

$\hat{x}$ : "best" of the infeasible individuals, which refers to the infeasible individual with the lowest degree of constraint violation in the current offspring population.
$A$ : archive, which is used to store the "best" of the infeasible individuals
$m''$ and $n''$ : user-defined parameters
*Condition 1*: the absolute difference of the original objective function values (i.e., $f(\vec{x})$ ) between the best and worst feasible solutions in the population $P$ is less than $\theta_1$ , where $\theta_1$ is a user-defined parameter with a very small positive value.
/*Condition 1 means the distinction among feasible solutions in a population is very slight.*/

**if** condition 1 does not hold **then**
  **if** $\neg\exists$ feasible individual in the current offspring population **then** /* It implies that the parent population is still far from the feasible region or a majority of individuals in the parent population are still far from the feasible region */
    $A := A \cup \hat{x}$ ; /*Preserve the "best" of the infeasible individuals. */
  **end if** /*See Fig. 6 (a) for illustration*/
  **if** mod $(gen, m'')=0$ **then** /**"gen" refers to the current generation*/
    randomly choose at most $n''$ individuals from the archive $A$ to replace the same number individuals randomly selected from the population $P$ ;
    $A := \phi$ ;
  **end if**
**end if**

Fig. 7. Pseudocode of the ISARM. The procedure above also embodies a self-adaptive process, because the number of infeasible individuals in archiving and replacing changes dynamically.

However, the proposed mechanism is very useful for COPs with a small proportion of the feasible region. Because in this case, the population $P$ and the offspring population are always composed of only infeasible solutions at the early stage, the proposed mechanism will drive the population approaching or landing in the feasible region constantly from the fact that the archiving and replacement of infeasible individuals occur frequently. Thereafter, in the general cases: 1) if the global optimum is located on the boundaries of the feasible region, then the search will synchronously occur from both sides of the boundaries of the feasible region based on the nondominated individuals replacement scheme and 2) if the global optimum lies within the feasible region, then feasible individuals will rapidly accumulate approximately from the middle stage, and finally strive for meeting the global optimum according to the nondominated individuals replacement scheme.

In addition, based on the Property 2 of nondominated individuals, feasible solutions will gradually increase as the iteration proceeds. So, this mechanism can also serve as a diversity operator to adjust the balance between feasible and infeasible individuals in the population because of the randomicity of the replacement of infeasible individuals.

On the other hand, when the population has converged or been very close to the optimum at the later stage, one may assume the optimal solution is located on the boundaries of the feasible region and the region surrounding the optimum consists of a very big scale of the infeasible region yet a very small scale of the feasible region (Fig. 8). In this case, if the procedure still does not terminate, the offspring population created from the crossover operator may be always composed of infeasible solutions, which means a sampling error has occurred. If the current offspring population consists of only infeasible solutions, according to ISARM, then the "best" of the infeasible solutions is to be archived. Hence, if the sampling error occurs with a very high frequency, it is very likely that the feasible individuals in

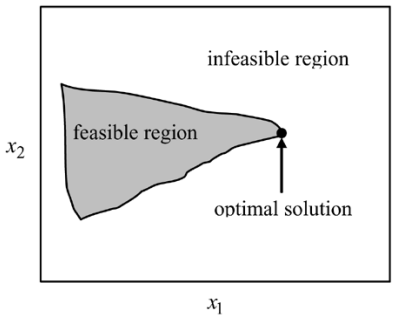



Fig. 8. A structure of the search space.

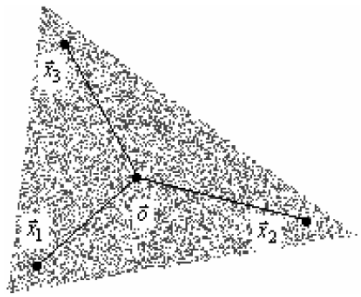



Fig. 9. Density of the offspring produced with three-parent SPX.

the population $P$ are unreasonably replaced by the infeasible solutions in the archive $A$. The worst scenario is that the feasibility proportion of the final population could possibly decrease to zero in the end because of the lack of sufficient knowledge about when the sampling error arises and the time the procedure should halt. To overcome this weakness, a simple yet effective measure is introduced in this paper as follows: once condition 1 is satisfied, the "best" of the infeasible solutions are not preserved in the archive $A$ later on.

### *E. Two Extreme Cases*

In this section, we will proceed to discuss a very difficult kind of COPs, i.e., COPs with equality constraints, in which the feasibility proportion of the entire search space is very small (almost equal to 0). First, the reason why feasible solutions cannot be found in the final population is analyzed. Then, one possible way to overcome this issue is presented, which relaxes the need for transforming equality constraints into inequality constraints adopted by most previous approaches.

With respect to the COPs with equality constraints, since the offspring population contains no feasible individual at the early stage, population $P$ will continuously approach the feasible set during the evolution on the basis of ISARM.

At the later stage, population $P$ will cluster in a small part of the search space. Assume that population $P$ remains infeasible and is not trapped in a local optimum. Two extreme cases then may occur: 1) if nondominated individuals in the offspring population include feasible solution, and if such feasible solution is nondominated with infeasible individuals in the parent population, then it cannot be accepted to the next population; and 2) few feasible solutions can enter the population, but due to ISARM, such stored feasible solutions are eliminated by infeasible solutions subsequently and, therefore, the population still does not contain any feasible solution in the end. The occurrence of these two extreme cases is the essential reason why feasible solutions cannot be found until the evolution halts, without the transformation of equality constraints.

*Condition 2*: Population $P$ consists entirely of infeasible individuals and $|\max\{\mathrm{f}(\vec{x}_i)|1 \le i \le N\} - \min\{\mathrm{f}(\vec{x}_j)|1 \le j \le N\}| < \vartheta_2$, where $\theta_2$ is a user-defined parameter with a very small positive value, $N$ is the population size.

Condition 2 indicates that the distinction among infeasible individuals in an infeasible population is extremely tiny.

To avoid the first extreme case, we introduce additional individual comparison criteria as follows: if condition 2 holds, then 1) any feasible solution is preferred to any infeasible solution; 2) between two feasible solutions, the one with better objective function value is preferred; and 3) between two infeasible solutions, the one with smaller constraint violation is preferred. According to the above individual comparison criteria, we can obtain Model 3. Model 3 is the same as Model 1, with the exception of step 4 in which we eliminate the former criterion of individual comparison in Models 1 and 2 (Pareto dominance), and only use the above criteria to judge which individual in the parent population is worse than a nondominated individual in the offspring population. Meanwhile, the way to address the second extreme case is that ISARM should not be applied if condition 2 holds.

By overcoming these two extreme cases, we can hold feasible solutions in the final population and relax the need to modify equality constraints.

### *F. Recombination*

In RCGA, the common recombination operators involve simulated binary crossover (SBX), unimodal normal distribution crossover (UNDX), simplex crossover (SPX) [37], etc. In this paper, our algorithm adopts SPX, which generates offspring based on uniform probability distribution and does not need any fitness information, as the recombination operator. The reason for using SPX is that it is very simple and the computational complexity for creating an offspring is only $O(N)$. Note that, no mutation operator is used in this paper.

In $\Re^n$, $\mu$ mutually independent parent vectors $(\vec{x}_i, i = 1, \ldots, \mu)$ form a simplex. The production of an offspring consists in: 1) employing a certain ratio to expand the original simplex in each direction $(\vec{x}_i - \vec{o}$, where $\vec{o}$ is the center of $\mu$ vectors, i.e., $\vec{o} = (1)/(n+1)\sum_{i=1}^{\mu}\vec{x}_i)$ and forming a new simplex; and 2) choosing one point from the new simplex as an offspring. For simplicity, we consider this process with a three-parent SPX in two-dimensional space, where $\vec{x}_1, \vec{x}_2$, and $\vec{x}_3$ indicate three vectors of the parents. Then, these vectors form a simplex. We expand this simplex in each direction by a factor of $(1+\varepsilon)$ ($\varepsilon \ge 0$ is the expanding rate). Let $\vec{o} = (1/3)\sum_{i=1}^{3}\vec{x}_i$ and $\vec{y}_i = (1+\varepsilon)(\vec{x}_i - \vec{o})$, thus $\vec{y}_1, \vec{y}_2$, and $\vec{y}_3$ form a new simplex. We then randomly choose a point $\vec{z}$ from the new simplex, i.e., $\vec{z} = k_1\vec{y}_1 + k_2\vec{y}_2 + k_3\vec{y}_3 + \vec{o}$, where, $k_1, k_2$, and $k_3$ are randomly selected within the range [0, 1] and satisfy the condition $k_1 + k_2 + k_3 = 1$. Fig. 9 illustrates the density of the offspring produced with three-parent SPX.

**Begin**
Select appropriate parameters and generate the initial main population $P$ randomly;
**Repeat**
**If** condition 2 does not hold **then**
Execute one model of a population-based algorithm-generator (i.e., Model 1 or Model 2);
**Else**
Execute Model 3;
**End if** (operator 1)
**If** condition 2 does not hold **then**
Implement the infeasible solutions archiving and replacement mechanism; (operator 2)
**End if**
**Until** either an acceptable solution or a predetermined number of fitness function evaluations (FFEs) is reached
**End.**

Fig. 10. Framework of the proposed algorithm.

### G. Framework

We incorporate the individual components described in detail above and present the framework of our algorithm in Fig. 10.

It is necessary to note that the operators 1 and 2 in the procedure are detached but interactive. Moreover, in the process of evolution, the number of nondominated individuals in the offspring population, the number of individuals in the parent population replaced by nondominated individuals (only in Model 1), and the number of infeasible individuals chosen from the archive $A$ to replace the same number of individuals in the population $P$ are dynamic, respectively.

We now explain how the two objectives of COEAs are addressed by our algorithm. For the first objective, the operator 2 in the procedure is designed to motivate the population approaching or landing in the feasible region efficiently. For the second objective, the algorithm provides operator 1 including three computation models in the procedure to guide the search toward the optimum. Meanwhile, the effects of operator 2 will become weaker and weaker with the gradual landing of individuals in the feasible region. A remarkable difference between our algorithm and other methods is that, in our approach guiding the population toward feasibility and refining the fitness of individuals are pursued simultaneously by combining the operators 1 and 2, while some other methods (such as [16], [38], and [39]) only focus on reaching the feasible region of the search space before improving the objective performance of individuals.

### H. Computational Time Complexity

From all of the psuedocodes listed above, we can see that the algorithm proposed here is an efficient approach. The identifying of all nondominated individuals in the offspring population requires $2\lambda\bar{\lambda}$ comparisons among individuals, where $\bar{\lambda}$ is the number of nondominated individuals in the offspring population. The operation of adopting nondominated individuals to replace the dominated individuals needs $2\mu\bar{\lambda}$ comparisons for Models 1 and 3, and $2\mu$ comparisons for Model 2. The number of comparisons can be negligible for ISARM. Therefore, the overall comparisons will be $2(\mu+\lambda)\bar{\lambda}$ or $2(\lambda\bar{\lambda}+\mu)$ when using Model 1 or 2, respectively.

TABLE I
SUMMARY OF 13 BENCHMARK FUNCTIONS

| Fcn | n | Type of $f$ | $\rho$ | LI | NE | NI | $a$ |
|---|---|---|---|---|---|---|---|
| g01 | 13 | quadratic | 0.0003% | 9 | 0 | 0 | 6 |
| g02 | 20 | nonlinear | 99.9965% | 1 | 0 | 1 | 1 |
| g03 | 10 | nonlinear | 0.0000% | 0 | 1 | 0 | 1 |
| g04 | 5 | quadratic | 26.9356% | 0 | 0 | 6 | 2 |
| g05 | 4 | nonlinear | 0.0000% | 2 | 3 | 0 | 3 |
| g06 | 2 | nonlinear | 0.0064% | 0 | 0 | 2 | 2 |
| g07 | 10 | quadratic | 0.0003% | 3 | 0 | 5 | 6 |
| g08 | 2 | nonlinear | 0.8640% | 0 | 0 | 2 | 0 |
| g09 | 7 | nonlinear | 0.5256% | 0 | 0 | 4 | 2 |
| g10 | 8 | linear | 0.0005% | 3 | 0 | 3 | 3 |
| g11 | 2 | quadratic | 0.0000% | 0 | 1 | 0 | 1 |
| g12 | 3 | quadratic | 0.0197% | 0 | 0 | $9^3$ | 0 |
| g13 | 5 | nonlinear | 0.0000% | 0 | 3 | 0 | 3 |

## IV. EXPERIMENTAL STUDY

In this section, we evaluate the performance of our algorithm on thirteen well-known benchmark functions. All test functions are taken from [17]. These test cases include various types (linear, nonlinear and quadratic) of objective functions with different number of decision variables $(n)$ and a range of types [linear inequalities (LI), nonlinear equalities (NE), and nonlinear inequalities (NI)] and number of constraints. The main characteristics of the test cases are reported in Table I, where $a$ is the number of constraints active at the optimal solution. In addition, $\rho$ is the ratio between the size of the feasible search space and that of the entire search space, i.e.,

$$\rho = |\Omega|/|S|$$

where $|S|$ is the number of solutions randomly generated from $S$, $|\Omega|$ is the number of feasible solutions out of these $|S|$ solutions. In our experiment setup, $|S| = 1\,000\,000$.

For each test case, 50 independent trials are executed in MATLAB (the source code may be obtained from the authors upon request). We use $\mu = n+1, \lambda = 10, m'' = 10, n'' = 2$ and $\theta_1 = 1E-10$. Also, $\theta_2$ is set to $10^{\theta_3^+ \log_{10}^{abs(current\_\min\_objective)}}$, where $\theta_3 = -12$, and "current_min_objective" denotes the minimum original objective function value [i.e., $f(\vec{x})$] in the current population $P$. These parameters were kept in all the experiments.

Indeed, one can conclude that there are about $(n+1)m''$ individuals taking part in the crossover operation before ISARM is implemented every time. To ensure that each individual in the population has enough lifespan to contribute its valuable information, we hope that each individual can carry out the crossover operation about once before being replaced. According to the above analysis, for the sake of convenience, the population size of each test case is set as follows:

$$N = \begin{cases} 50, & 0 < n < 5 \\ 100, & 5 \le n \le 15 \\ 150, & 15 < n \le 20 \end{cases}.$$

The number of FFEs is 350 000 for all the test cases. Note that, ten FFEs are performed at each generation.

TABLE II
EXPERIMENT RESULTS WITH 50 INDEPENDENT RUNS ON 13 BENCHMARK FUNCTIONS USING MODEL 1 OR MODEL 2 (THE RESULTS WITH MODEL 1 ARE BETTER THAN THE CORRESPONDING ONES WITH MODEL 2 ARE SHOWN IN "**BOLDFACE**")

| Fcn/ optimal | model | best | median | mean | st. dev | worst | FFEs |
|---|---|---|---|---|---|---|---|
| g01/ | Model 1 | -15.000000 | -15.000000 | -15.000000 | 1.109E-08 | -15.000000 | 350000 |
| -15.000000 | Model 2 | -15.000000 | -15.000000 | -15.000000 | 1.332E-14 | -15.000000 | 350000 |
| g02/ | Model 1 | -0.803619 | -0.803619 | -0.803169 | 2.201E-03 | -0.792608 | 350000 |
| -0.803619 | Model 2 | -0.803619 | -0.803619 | -0.803220 | 1.987E-03 | -0.792608 | 350000 |
| g03/ | Model 1 | -1.000000 | -1.000000 | -1.000000 | 8.027E-15 | -1.000000 | 350000 |
| -1.000000 | Model 2 | -1.000000 | -1.000000 | -1.000000 | 2.757E-16 | -1.000000 | 350000 |
| g04/ | Model 1 | -30665.539 | -30665.539 | -30665.538 | 4.262E-03 | -30665.508 | 350000 |
| -30665.539 | Model 2 | -30665.539 | -30665.539 | -30665.539 | 7.990E-12 | -30665.539 | 350000 |
| g05/ | Model 1 | 5126.4981 | 5126.4981 | 5126.4981 | 1.837E-12 | 5126.4981 | 350000 |
| 5126.4981 | Model 2 | 5126.4981 | 5126.4981 | 5126.4981 | 1.513E-12 | 5126.4981 | 350000 |
| g06/ | Model 1 | -6961.81388 | -6961.81388 | -6961.81388 | 1.837E-12 | -6961.81388 | 350000 |
| -6961.81388 | Model 2 | -6961.81388 | -6961.81388 | -6961.81388 | 1.837E-12 | -6961.81388 | 350000 |
| g07/ | Model 1 | 24.3062091 | 24.3062091 | 24.3062091 | **2.807E-12** | 24.3062091 | 350000 |
| 24.3062091 | Model 2 | 24.3062091 | 24.3062091 | 24.3062091 | 5.652E-12 | 24.3062091 | 350000 |
| g08/ | Model 1 | -0.095825 | -0.095825 | -0.095825 | 5.414E-17 | -0.095825 | 350000 |
| -0.095825 | Model 2 | -0.095825 | -0.095825 | -0.095825 | 3.241E-17 | -0.095825 | 350000 |
| g09/ | Model 1 | 680.6300573 | 680.6300573 | 680.6300573 | 5.742E-13 | 680.6300573 | 350000 |
| 680.6300573 | Model 2 | 680.6300573 | 680.6300573 | 680.6300573 | 4.661E-13 | 680.6300573 | 350000 |
| g10/ | Model 1 | 7049.248021 | 7049.248021 | 7049.637057 | 1.753E+00 | 7060.096482 | 350000 |
| 7049.248 | Model 2 | 7049.248021 | 7049.248021 | 7049.248021 | 4.013E-09 | 7049.248021 | 350000 |
| g11/ | Model 1 | 0.750000 | 0.750000 | 0.750000 | 0.000E+00 | 0.750000 | 350000 |
| 0.750000 | Model 2 | 0.750000 | 0.750000 | 0.750000 | 0.000E+00 | 0.750000 | 350000 |
| g12/ | Model 1 | -1.000000 | -1.000000 | -1.000000 | 0.000E+00 | -1.000000 | 350000 |
| -1.000000 | Model 2 | -1.000000 | -1.000000 | -1.000000 | 0.000E+00 | -1.000000 | 350000 |
| g13/ | Model 1 | 0.0539498 | 0.0539498 | 0.0539498 | **5.607E-17** | 0.0539498 | 350000 |
| 0.0539498 | Model 2 | 0.0539498 | 0.0539498 | 0.0539498 | 6.539E-17 | 0.0539498 | 350000 |

### A. General Performance of the Proposed Algorithm

We summarize the experimental results using the above parameters with Model 1 or 2 in Table II. For each test case, we list the "known" optimal solution and the best, median, mean and worst objective function values, and the standard deviations after 50 independent runs by our algorithm.

When Model 2 is adopted, our algorithm performs pretty well in that it consistently finds the global optima of all test cases for all 50 runs, except for problem g02. Moreover, with respect to problem g10, a typical solution found by our algorithm is: $\vec{x} = (579.30663371658\ 1359.97068227017\ 5109.97070454191\ 182.01769534597\ 295.60117181832\ 217.98230465403\ 286.1652352765\ 395.60117181832)$ with $f(\vec{x}) = 7049.248020528672$, which is the "best" result reported so far. For problem g02, a typical solution found by our algorithm is: $\vec{x} = (3.16246065061821\ 3.12833143125259\ 3.09479210698838\ 3.06145063097946\ 3.02792912724552\ 2.99382607867366\ 2.95866874241085\ 2.92184233357046\ 0.49482516067620\ 0.48835710726434\ 0.48231645861281\ 0.47664473053803\ 0.47129550866925\ 0.46623098401343\ 0.46142001695710\ 0.45683664685665\ 0.45245869011821\ 0.44826764426227\ 0.44424698415295\ 0.44038291462275)$ with $f(\vec{x}) = -0.80361910412559$, which is also the "best" result reported so far. Furthermore, for problem g02, the resulting objective function values are less than $-0.803619$ for 48 out of the 50 runs. Fig. 11 describes the distribution of the solution quality achieved over 50 trials for problem g02. In addition, it is apparent that the standard deviations provided in Table II are very small, which reflects that our approach is capable of performing a robust and stable search.

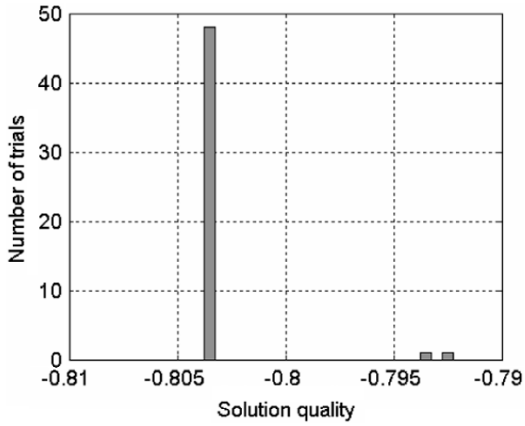



Fig. 11. Plot shows the number of trials versus the solution quality achieved. Results are derived from 50 independent trials on problem g02.

Compared with the execution of Model 2, the algorithm with Model 1 is relatively easy to get stuck in a local optimum due to a higher selection pressure. For example, premature convergence presents for problems g04 and g10 in addition to problem g02. Model 1 finds a "better" standard deviation in two problems (g07 and g13), whereas Model 2 finds "better" standard deviations in the remaining problems. In general, the overall performance of Model 1 appears appreciably worse than that of Model 2.

The results in Table II also reveal that our algorithm has substantial potential in coping with various COPs without any complex operators. Hereafter, we will only discuss the results provided by Model 2 unless otherwise specified.

TABLE III
AVERAGE PERCENTAGE OF FEASIBLE SOLUTIONS IN THE FINAL POPULATION WITH 50 INDEPENDENT RUNS

| Fcn | g01 | g02 | g03 | g04 | g05 | g06 | g07 | g08 | g09 | g10 | g11 | g12 | g13 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| average percentage | 100 | 99 | 91 | 86 | 100 | 35 | 88 | 71 | 66 | 94 | 42 | 100 | 9 |

TABLE IV
AVERAGE DISTANCE FROM THE BEST INDIVIDUAL OF THE POPULATION TO THE BOUNDARIES OF THE FEASIBLE REGION AT EVERY 5000 GENERATIONS FOR THE 50 TRIALS

| Fcn | Iteration number | | | | | | |
|---|---|---|---|---|---|---|---|
| | 5000 | 10000 | 15000 | 20000 | 25000 | 30000 | 35000 |
| g01 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| g02 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| g03 | 9.54E-09 | 6.57E-10 | 4.12E-12 | 6.66E-16 | 0 | 0 | 0 |
| g04 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| g05 | 3.27E-05 | 0 | 0 | 0 | 0 | 0 | 0 |
| g06 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| g07 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| g08 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| g09 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| g10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| g11 | 3.78E-12 | 0 | 0 | 0 | 0 | 0 | 0 |
| g12 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| g13 | 3.62E-03 | 3.16E-05 | 1.94E-09 | 5.67E-12 | 0.00E-14 | 0 | 0 |

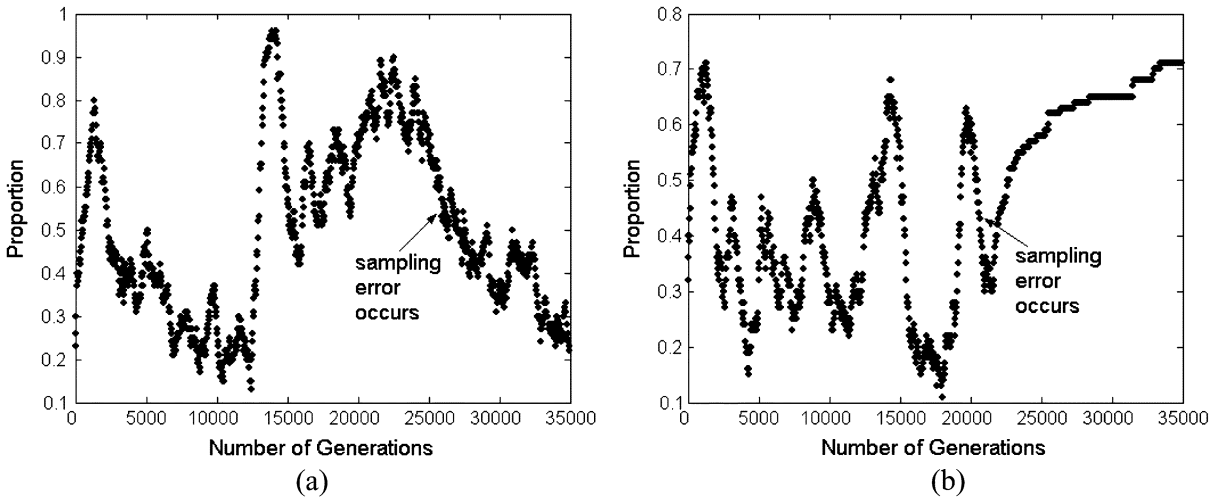



Fig. 12. Proportion of feasible solutions during the evolution with two different trials on test function g04. (a) Without using the parameter $\theta_1$ (i.e., $\theta_1 = 0$). (b) Using the parameter $\theta_1$ (i.e., $\theta_1 = 1\mathrm{E}-10$).

### *B. Feasibility Proportion of the Final Population*

For all test cases, feasible solutions are consistently found for all 50 trials. Table III depicts the average feasibility percentage of the final population.

In order to ascertain the rate at which the algorithm is able to approach or enter the feasible region, we monitor the average distance from the best individual of the population to the boundaries of the feasible region at every 5000 generations for the 50 trials. The results are presented in Table IV. As can be seen, for the test problems without equality constraints (g01, g02, g04, g06, g07, g08, g09, g10, and g12), the algorithm can enter the feasible region within 5000 generations (i.e., 50 000 FFEs); for problems g05 and g11, the algorithm can enter the feasible region within 10 000 generations (i.e., 100 000 FFEs). Although for problems g03 and g13, the algorithm can enter the feasible region within 25 000 and 30 000 generations (i.e., 250 000 and 300 000 FFEs), after 10 000 generations, the best individual of the population has had very little distance to the boundaries of the feasible region. The above observation verifies that the algorithm has the capability in approaching or entering the feasible region quickly.

Now, we will use test function g04 whose optimum is located on the boundaries of the feasible region, to explain the effects of the sampling error of the crossover operator on population feasibility. We design two contrasting trials, i.e., implement our algorithm with or without using the parameter $\theta_1$, and the feasibility proportion of the population is recorded at every 30 generations. From Fig. 12(a), we can easily observe that when the population has been very close or converged to the optimum at the later stage, the sampling error induces a rapid decrease of the feasibility proportion of the population. Fig. 12(b) verifies the effectiveness of the use of the parameter $\theta_1$.

Figs. 13 and 14 account for the effects of the two extreme cases described in Section III-E on finding feasible solutions for COPs with equality constraints. Also, the feasibility proportion of the population is recorded at every 30 generations. In test

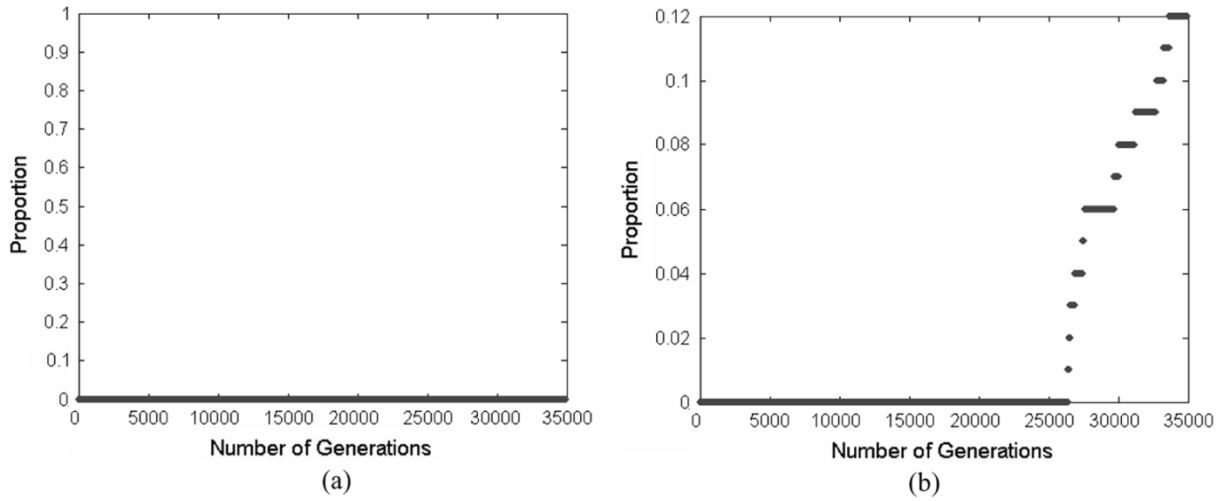



Fig. 13. Proportion of feasible solutions during the evolution with two different trials on test function g13. (a) The first extreme case happens. (b) Overcome the first extreme case by applying the additional individual comparison criteria when condition 2 holds.

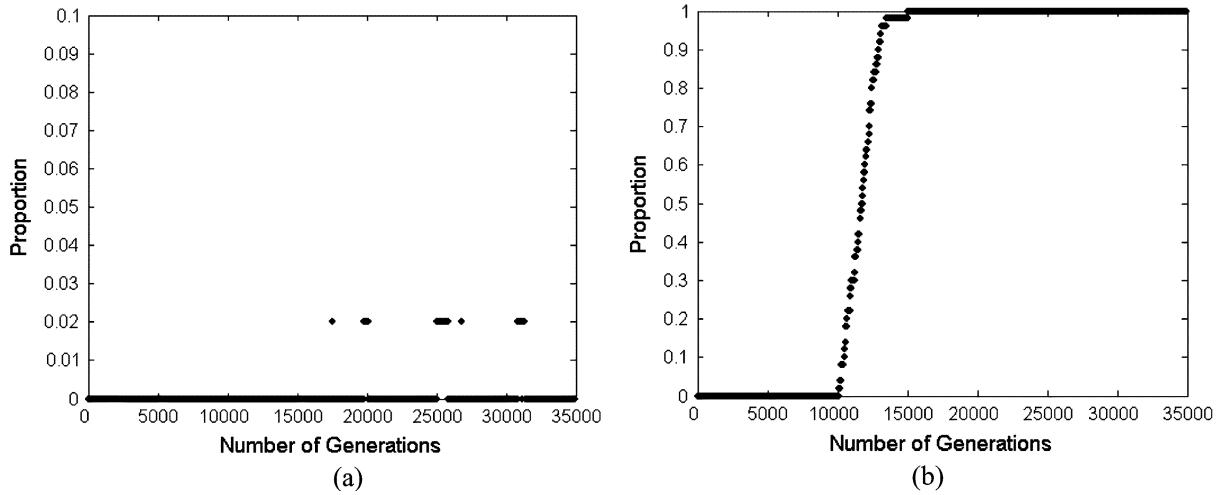



Fig. 14. Proportion of feasible solutions during the evolution with two different trials on test function g05. (a) The second extreme case happens. (b) Overcome the second extreme case by stopping performing ISARM when condition 2 is satisfied.

case g13, some feasible solutions can be found in the offspring population during the search process. However, these feasible solutions cannot survive in the next generation, because they are nondominated with infeasible solutions in the parent population and the population $P$ only involves infeasible solutions (note that this is the first extreme case). By applying the additional individual comparison criteria when condition 2 holds, this phenomenon can be addressed to some extent [Fig. 13(b)]. For problem g05, although some feasible solutions can enter the population $P$, they are replaced by infeasible solutions in the archive $A$ subsequently due to ISARM, thereby the population $P$ does not contain any feasible solution in the end [Fig. 14(a), note that this is the second extreme case]. It is clearly shown in Fig. 14(b) that stopping performing ISARM when condition 2 is satisfied can help the population to land within the feasible set under this condition.

### C. Comparison of Different Constrained Optimization Approaches

We compare our algorithm against some other representative approaches using two metrics suggested in [40], i.e., solution qualities and computation effort. The other approaches in the comparison include:

1) Self-Adaptive Fitness Formulation (SAFF) method [9];
2) Simple Multimembered Evolution Strategy (SMES) method [16];
3) Stochastic Ranking (RY) method [17];
4) Inverted-Shrinkable PAES (IS-PAES) method [26];
5) Derivative-Free Filter Simulated Annealing (FSA) method [40];
6) Improved Stochastic Ranking (IRY) method [18].

Among the six algorithms above, algorithm 5 is a point-to-point method and the rest are population-based methods. The experiment results are listed in Table V.

One can conclude that our algorithm can consistently produce the best performance for all the test functions, as opposed to methods 1–5, which only manage to achieve best performance for certain test functions. Besides, our algorithm consistently converges to the optimum for all 50 runs, except for problem g02 where premature convergence occurs for 2 out of 50 runs. While methods 1–5 seem to have a great tendency to converge to a local optimum, especially for those complicated functions, e.g., g02, g05, g07, g09, g10, and g13. For the large feasible problem g02,

TABLE V
COMPARISON OF OUR ALGORITHM (INDICATED BY CW) WITH RESPECT TO SAFF [9], SMES [16], RY [17], IS-PAES [26], FSA [40], AND IRY [18] ON 13 BENCHMARK FUNCTIONS. NA = NOT AVAILABLE. THE RESULTS OBTAINED BY ALGORITHMS 1–6 AND BETTER THAN THE CORRESPONDING ONES RESULTED FROM OUR ALGORITHM ARE SHOWN IN "**BOLDFACE**"

| Fcn/ optimal | status | methods | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | SAFF[9] | SMES[16] | RY[17] | IS-PAES[26] | FSA[39] | IRY[18] | CW |
| g01/ -15.000 | best | -15.000 | -15.000 | -15.000 | -15.000 | -14.999 | -15.000 | -15.000 |
| | mean | -15.000 | -15.000 | -15.000 | -14.494 | -14.993 | -15.000 | -15.000 |
| | worst | -15.000 | -15.000 | -15.000 | -12.446 | -14.980 | -15.000 | -15.000 |
| | st. dev | 0 | 0 | **0.0E+00** | 9.3E-01 | 4.8E-03 | 1.3E-13 | 1.3E-14 |
| g02/ -0.803619 | best | -0.80297 | -0.803601 | -0.803515 | -0.803376 | -0.754913 | -0.803619 | -0.803619 |
| | mean | -0.79010 | -0.785238 | -0.781975 | -0.793281 | -0.371708 | -0.772078 | -0.803220 |
| | worst | -0.76043 | -0.751322 | -0.726288 | -0.768291 | -0.271311 | -0.683055 | -0.792608 |
| | st. dev | 1.2E-02 | 1.7E-02 | 2.0E-02 | 9.0E-03 | 9.8E-02 | 2.6E-02 | 2.0E-03 |
| g03/ -1.000 | best | -1.000 | -1.000 | -1.000 | -1.000 | -1.000 | -1.001 | -1.000 |
| | mean | -1.000 | -1.000 | -1.000 | -1.000 | -0.999 | -1.001 | -1.000 |
| | worst | -1.000 | -1.000 | -1.000 | -1.000 | -0.992 | -1.001 | -1.000 |
| | st. dev | 7.5E-05 | 2.1E-04 | 1.9E-04 | 9.7E-05 | 1.7E-03 | 6.0E-09 | 2.8E-16 |
| g04/ -30665.539 | best | -30665.50 | -30665.539 | -30665.539 | -30665.539 | -30665.538 | -30665.539 | -30665.539 |
| | mean | -30665.20 | -30665.539 | -30665.539 | -30665.539 | -30665.467 | -30665.539 | -30665.539 |
| | worst | -30663.30 | -30665.539 | -30665.539 | -30665.539 | -30664.688 | -30665.539 | -30665.539 |
| | st. dev | 4.9E-01 | 0 | 2.0E-05 | 0 | 1.7E-01 | 2.2E-11 | 8.0E-12 |
| g05/ 5126.498 | best | 5126.989 | 5126.599 | 5126.497 | NA | 5126.4981 | 5126.4981 | 5126.4981 |
| | mean | 5432.080 | 5174.492 | 5128.881 | NA | 5126.4981 | 5126.4981 | 5126.4981 |
| | worst | 6089.430 | 5304.167 | 5142.472 | NA | 5126.4981 | 5126.4981 | 5126.4981 |
| | st. dev | 3.9E+03 | 5.0E+01 | 3.5E+00 | NA | 0 | 6.2E-12 | 1.513E-12 |
| g06/ -6961.814 | best | -6961.800 | -6961.814 | -6961.814 | -6961.814 | -6961.814 | -6961.814 | -6961.814 |
| | mean | -6961.800 | -6961.284 | -6875.940 | -6961.813 | -6961.814 | -6961.814 | -6961.814 |
| | worst | -6961.800 | -6952.482 | -6350.262 | -6961.810 | -6961.814 | -6961.814 | -6961.814 |
| | st. dev | 0 | 1.9E+00 | 1.6E+02 | 8.5E-05 | 0 | 6.4E-12 | 1.8E-12 |
| g07/ 24.306 | best | 24.48 | 24.327 | 24.307 | 24.338 | 24.311 | 24.306 | 24.306 |
| | mean | 26.58 | 24.475 | 24.374 | 24.527 | 24.380 | 24.306 | 24.306 |
| | worst | 28.40 | 24.843 | 24.642 | 24.995 | 24.644 | 24.308 | 24.306 |
| | st. dev | 1.1E+00 | 1.3E-01 | 6.6E-02 | 1.7E-01 | 7.2E-02 | 2.7E-04 | 5.7E-12 |
| g08/ -0.095825 | best | -0.095825 | -0.095825 | -0.095825 | -0.095825 | -0.095825 | -0.095825 | -0.095825 |
| | mean | -0.095825 | -0.095825 | -0.095825 | -0.095825 | -0.095825 | -0.095825 | -0.095825 |
| | worst | -0.095825 | -0.095825 | -0.095825 | -0.095825 | -0.095825 | -0.095825 | -0.095825 |
| | st. dev | 0 | 0 | 2.6E-17 | 0 | 0 | 4.2E-17 | 3.2E-17 |
| g09/ 680.630 | best | 680.64 | 680.632 | 680.630 | 680.630 | 680.630 | 680.630 | 680.630 |
| | mean | 680.72 | 680.643 | 680.656 | 680.631 | 680.636 | 680.630 | 680.630 |
| | worst | 680.87 | 680.719 | 680.763 | 680.634 | 680.698 | 680.630 | 680.630 |
| | st. dev | 5.9E-02 | 1.6E-02 | 3.4E-02 | 8.1E-04 | 1.5E-02 | **4.6E-13** | 4.7E-13 |
| g10/ 7049.248 | best | 7061.34 | 7051.903 | 7054.316 | 7062.019 | 7059.864 | 7049.248 | 7049.248 |
| | mean | 7627.89 | 7253.047 | 7559.192 | 7342.944 | 7509.321 | 7049.249 | 7049.248 |
| | worst | 8288.79 | 7638.366 | 8835.655 | 7588.054 | 9398.649 | 7049.296 | 7049.248 |
| | st. dev | 3.7E+02 | 1.4E+02 | 5.3E+02 | 1.4E+02 | 5.4E+02 | 4.9E-03 | 4.0E-09 |
| g11/ 0.75 | best | 0.750 | 0.75 | 0.750 | 0.750 | 0.750 | 0.750 | 0.750 |
| | mean | 0.750 | 0.75 | 0.750 | 0.750 | 0.750 | 0.750 | 0.750 |
| | worst | 0.750 | 0.75 | 0.750 | 0.751 | 0.750 | 0.750 | 0.750 |
| | st. dev | 0 | 1.5E-04 | 8.0E-05 | 2.6E-04 | 0 | 1.8E-15 | 0.0E+00 |
| g12/ -1.000 | best | -1.000 | -1.000 | -1.000 | NA | -1.000 | -1.000 | -1.000 |
| | mean | -1.000 | -1.000 | -1.000 | NA | -1.000 | -1.000 | -1.000 |
| | worst | -1.000 | -1.000 | -1.000 | NA | -1.000 | -1.000 | -1.000 |
| | st. dev | 0 | 0 | 0.0E+00 | NA | 0 | 9.6E-10 | 0.0E+00 |
| g13/ 0.0539498 | best | NA | 0.053986 | 0.053957 | 0.05517 | 0.0539498 | 0.053942 | 0.0539498 |
| | mean | NA | 0.166385 | 0.067543 | 0.28184 | 0.2977204 | 0.096276 | 0.0539498 |
| | worst | NA | 0.468294 | 0.216915 | 0.5471 | 0.4388511 | 0.438803 | 0.0539498 |
| | st. dev | NA | 1.8E-01 | 3.1E-02 | 1.8E-01 | 1.9E-01 | 1.2E-01 | 6.5E-17 |

methods 1–5 are unable to reach the true optimum. Although the "best" objective function value of $-0.803601$ found by method 2 is very close to the optimum, the resulting objective function values of our algorithm are less than that for 48 out of 50 trials. In terms of the highly constrained problem g10, the "best" objective function values of these five methods are still far from the true optimum. Moreover, for problems g10, methods 1 and 3 have trouble in finding feasible solutions. More specifically, method 1 finds 17 feasible solutions from 20 runs, and method 3 finds only 6 from 30 runs. However, our algorithm consistently finds the feasible optimum for this problem.

With respect to method 6, which is the most competitive approach known to date, most of its results match the solutions derived from our algorithm. Method 6 obtains a "better" standard deviation in problem g09, while our algorithm reaches "better" standard deviations in the remaining 12. Also, method 6 cannot consistently converge to the global optima for problems g02, g07, g10, and g13.

TABLE VI
EXPERIMENT RESULTS WITH 50 INDEPENDENT RUNS ON 13 BENCHMARK FUNCTIONS WITHOUT USING ISARM. "*" MEANS NO FEASIBLE SOLUTIONS WERE FOUND. "MEAN DISTANCE" DENOTES THE AVERAGE DISTANCE FROM THE BEST INDIVIDUAL IN THE FINAL POPULATION TO THE BOUNDARIES OF THE FEASIBLE REGION

| Fcn | optimal | best | median | worst | mean distance |
|---|---|---|---|---|---|
| g01 | -15.000 | -57.897* | -36.199* | -14.178* | 9.7E+01 |
| g02 | -0.803619 | -0.803619 | -0.803619 | -0.792608 | 0 |
| g03 | -1.000 | -2.301* | -0.927* | -0.151* | 5.7E-03 |
| g04 | -30665.539 | -30665.539 | -30665.538 | -30665.515 | 0 |
| g05 | 5126.498 | 2979.760* | 4248.671* | 5076.819* | 1.8E+02 |
| g06 | -6961.814 | -6961.814 | -6961.788 | -6955.513 | 0 |
| g07 | 24.306 | 24.306 | 24.306 | 24.798 | 0 |
| g08 | -0.095825 | -0.095825 | -0.095825 | -0.095825 | 0 |
| g09 | 680.630 | 680.630 | 680.630 | 680.630 | 0 |
| g10 | 7049.248 | 5107.408* | 7588.978* | 10681.686 | 1.6E-01 |
| g11 | 0.75 | 0.75* | 0.75* | 0.985* | 2.7E-05 |
| g12 | -1.000 | -1.000 | -1.0000 | -1.000 | 0 |
| g13 | 0.0539498 | 0.0465034* | 0.0570823* | 0.0911286* | 1.0E-01 |

On the other hand, because of the transformation from an equality constraint into pairwise inequality constraints and the use of parameter $\delta$, some results provided by algorithms 1–6 are better than the "known" optimum. For instance, for problems g03, g04, g06, and g11, the "best" results provided by method 2 [41]; for problem g05, the "best" result provided by method 3; for problem g11, the "best" result provided by method 5; and for problems g03 and g13, the "best" results provided by method 6 are better than the "known" optima, respectively. But the above behaviors do not mean that the "new" optima are really found by them.

As far as the computational cost (the number of FFEs) is concerned, algorithms 2 and 5 seem to have the minimum computational cost for most test functions, while algorithm 1 has considerable computational cost for all test functions (1 400 000 FFEs). In general, the cost required by our algorithm (350 000 FFEs) is moderate among the constrained optimization approaches in comparison.

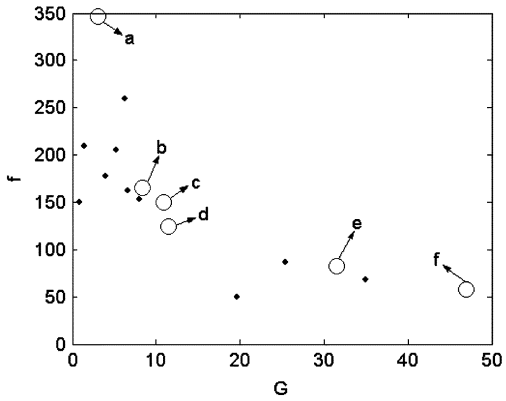



Fig. 15. Some nondominated individuals (e.g., individuals "a," "b," "e," and "f") in the offspring population are dominated by the individuals in the parent population. In addition, some nondominated individuals (e.g., individual "c" and "d") are nondominated with the individuals in the parent population.

### D. Validation of ISARM

An important factor of our algorithm is the promotion of the population feasibility by means of ISARM. In this section, we attempt to examine whether ISARM works as expected, i.e., we will remove ISARM from the algorithm proposed and monitor whether there exist some changes of the capability. We have used Table VI in order to accomplish this investigation. This time, 50 independent runs are also performed. The following is a summary of the results obtained.

*A) Test Functions g01, g03, g05, g10, g11, and g13:* In these cases, feasible solutions cannot be found during the evolution. This occurs because nondominated individuals in the offspring population are always dominated by or nondominated with the individuals in the parent population (see Fig. 15), thus, new individuals produced cannot enter the population and the population cannot proceed to approach the feasible region under this condition. The use of ISARM clearly improved the results by directing the population toward feasibility from different directions.

*B) Test Functions g04, g06, and g07:* In these cases, although the population can find feasible solutions, the global optimum cannot be attained. Again, the use of ISARM produced a significant improvement for our algorithm. However, this time ISARM serves as a diversity mechanism as stated in Section III-D.

*C) Test Function g02:* In this case, the version without ISARM provided a similar capability. As already analyzed, regarding some COPs with large ratio of the feasible set, similar to problem g02, the role of the ISARM is not very apparent, but empirical results indicate that the models of a population-based algorithm-generator have powerful abilities in finding the global optima.

*D) Test Functions g08, g09, and g12:* In these cases, the global optimum can be found consistently in all experiments. Once more, the effect of ISARM is not very evident. We attribute this behavior to the fact that these test functions have a search space considerably easier to explore than the others employed in this paper.

*Conclusions from Experiment:* Based on the analysis performed in this experiment, we conclude that the use of ISARM turns out to be quite beneficial in most cases with small proportion of the feasible region, but the impact of ISARM is not evident when the feasibility proportion of the problem is very large or the problem itself is relatively simple to solve. However, since the replacement behavior of ISARM happens randomly and the likelihood of replacement becomes less and less with the gradual increase of feasible solutions in the population, the overall orientation of advance for the population is primarily

TABLE VII
EXPERIMENTAL RESULTS ON FOUR PROBLEMS WITH EQUALITY CONSTRAINTS WITH VARYING $\theta_3$; 50 INDEPENDENT RUNS WERE PERFORMED; MPF MEANS THE MEAN PERCENTAGE OF FEASIBLE SOLUTIONS IN THE FINAL POPULATION; (#) DENOTES THE NUMBER OF TRIALS THAT FEASIBLE SOLUTIONS ARE NOT FOUND

| $\theta_3$ | g03 | | | g05 | | | g11 | | | g13 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Mean | st. dev | MPF | Mean | st. dev | MPF | Mean | st. dev | MPF | Mean | st. dev | MPF |
| -9 | | (20) | | 5126.4981 | 2.9E-07 | 100 | 0.75 | 0.0E+00 | 98 | | (1) | |
| -10 | -1.000 | 9.6E-12 | 7 | 5126.4981 | 5.1E-09 | 100 | 0.75 | 0.0E+00 | 93 | 0.0539498 | 2.1E-14 | 8 |
| -11 | -1.000 | 5.1E-13 | 56 | 5126.4981 | 7.5E-10 | 100 | 0.75 | 0.0E+00 | 72 | 0.0539498 | 3.7E-14 | 8 |
| -12 | -1.000 | 2.8E-16 | 91 | 5126.4981 | 1.5E-12 | 100 | 0.75 | 0.0E+00 | 42 | 0.0539498 | 6.5E-17 | 9 |
| -13 | -1.000 | 3.2E-15 | 90 | 5126.4981 | 1.8E-12 | 100 | 0.75 | 0.0E+00 | 22 | 0.0539498 | 5.6E-17 | 8 |
| -14 | -1.000 | 1.4E-14 | 86 | 5126.4981 | 1.8E-12 | 96 | | (1) | | | (48) | |

controlled by the models of a population-based algorithm-generator and assisted by ISARM during the evolution. Thus, the global optimum can always be found by coupling these two main ingredients of our algorithm.

### E. Influence of the Parameter $\theta_2$

The parameter $\theta_2$ is effective for finding feasible solutions when tackling COPs with equality constraints. As discussed, this parameter reflects the distinction among infeasible individuals in an infeasible population. To get a tradeoff between the quality of the resulting solutions and the number of feasible solutions, this parameter should be small. However, a too small value may also induce the final population being infeasible. Therefore, suitable value of this parameter must be selected. Table VII summarizes the mean of the objective function values, the standard deviations of the objective function values, and the mean percentage of feasible solutions in the final population in the case of the parameter $\theta_3$ alone being changed to $-9, -10, -11, -12, -13$, and $-14$. In the case of $\theta_3 = -9$, feasible solutions for problems g03 and g13 cannot be consistently found, and the standard deviation for problem g05 is bigger than the standard deviation when $\theta_3$ is greater. Also, in the case of $\theta_3 = -14$, feasible solutions for problems g11 and g13 cannot be consistently found. These results show that a value of the parameter $\theta_3$ between $-10$ and $-13$ is an appropriate setting for the problems with equality constraints.

In addition, the proposed method is very robust with respect to the used setting of the parameter $\theta_1$. The related results about this parameter are not listed here for limited pages.

### F. Sensitivity in Relation to the Parameters $m''$ and $n''$

In order to illustrate the effect of the parameters $m''$ and $n''$ used by ISARM on performance, a set of experiments has been performed. We use the exact setting of values for the remaining parameters used in our previous experiments, and we also maintain the number of FFEs to have a fair comparison. We only modify the aforementioned two parameters in the following combinations: 1) $m'' = 5$ and $n'' = 2$; 2) $m'' = 15$ and $n'' = 2$; 3) $m'' = 10$ and $n'' = 1$; and 4) $m'' = 10$ and $n'' = 3$. For each combination, we perform 50 runs for per test function. The experimental results are presented in Table VIII. The results that are approximately the same as those shown in Table II are omitted. The following is a summary of the results obtained.

1) There is a negative effect when increasing the parameter $m''$ or decreasing the parameter $n''$. This phenomenon is reasonable, because with higher value of $m''$ or with lower value of $n''$, the replacement of ISARM will occur with a relatively lower frequency and ISARM cannot exert its strength very well. Thus, in one respect, only a fraction of the final populations can find feasible solutions for some test cases, such as problems g03, g11, and g13. On the other hand, although the final population can consistently find feasible solutions for some test cases, it is unable to converge to the global optimum, such as problems g01, g04, g07, and g10.
2) We can argue that there is no significant negative effect when decreasing the parameter $m''$ or increasing the parameter $n''$. Nevertheless, a lower value of $m''$ and higher value of $n''$ also indicate that the replacement of ISARM will occur with a relatively higher frequency, so that some fresh individuals in the population may be replaced unreasonably before they contribute their valuable information as analyzed and, therefore, the degradation of performance tends to take place not only with respect to problems g05, but to problem g13 where a part of the final populations cannot find infeasible solutions.

### G. Influence of the Parameter $\lambda$

Finally, one may be interested in the sensitivity of the parameter $\lambda$. Again, four new runs are performed with different parameter values (5, 8, 12, and 15), each using the same remaining parameters and the number of FFEs adopted in our previous experiments. The results are given in Table IX. From this table, it can be seen that smaller values for $\lambda$ (5 and 8) do not influence the capability of our algorithm. However, smaller values of $\lambda$ also mean slower convergence. Indeed, the convergence reliability of our algorithm when using $\lambda = 10$ is appreciably better than that of $\lambda = 5$ and 8.

However, the degradation of capability arises when using the bigger values of $\lambda$ (12 and 15) that mean a lower number of iterations. On the one hand, a lower number of iterations may result in an incomplete convergence of the population, see for example functions g01, g04, g05, g07, and g10. On the other hand, ISARM has difficulty in guiding the population toward feasibility due to a lower number of iterations, see, for example, functions g03, g11, and g13.

### H. Limitation and Discussion

The limitation of our algorithm is the problem-dependence of the crossover operator, i.e., the expanding factor $\varepsilon$ should be altered based on specific problems. Though $\varepsilon = \sqrt{n+2}$ is a theoretical guideline in using SPX, and it is shown that the performance of this value works well on various test functions

TABLE VIII
STATISTICS RESULTS OBTAINED BY CW WITH DIFFERENT VALUES FOR THE $m''$ AND $n''$ PARAMETERS. N/E MEANS NO EVIDENT EFFECT FOUND BY THIS PARAMETER COMBINATION COMPARED WITH THE RESULTS WHEN USING $m'' = 10$ AND $n'' = 2$. (#) DENOTES THE NUMBER OF TRIALS THAT FEASIBLE SOLUTIONS ARE NOT FOUND IN 50 TRIALS. [#] DENOTES THE AVERAGE DISTANCE OF THE POPULATION FROM THE BOUNDARIES OF THE FEASIBLE REGION

| Fcn | status | different parameter values | | | |
|---|---|---|---|---|---|
| | | $m''=5$ $n''=2$ | $m''=15$ $n''=2$ | $m''=10$ $n''=1$ | $m''=10$ $n''=3$ |
| g01 | best | N/E | -15.000 | -15.000 | N/E |
| | mean | N/E | -15.000 | -15.000 | N/E |
| | worst | N/E | -15.000 | -15.000 | N/E |
| | st. dev | N/E | 6.7E-10 | 2.6E-07 | N/E |
| g02 | best | N/E | N/E | N/E | N/E |
| | mean | N/E | N/E | N/E | N/E |
| | worst | N/E | N/E | N/E | N/E |
| | st. dev | N/E | N/E | N/E | N/E |
| g03 | best | N/E | N/E | | N/E |
| | mean | N/E | N/E | (27) | N/E |
| | worst | N/E | N/E | [6.4E-15] | N/E |
| | st. dev | N/E | N/E | | N/E |
| g04 | best | N/E | -30665.54 | -30665.54 | N/E |
| | mean | N/E | -30665.54 | -30665.54 | N/E |
| | worst | N/E | -30665.54 | -30665.54 | N/E |
| | st. dev | N/E | 3.5E-09 | 9.0E-07 | N/E |
| g05 | best | 5126.498 | N/E | N/E | 5126.498 |
| | mean | 5126.513 | N/E | N/E | 5126.499 |
| | worst | 5126.747 | N/E | N/E | 5126.550 |
| | st. dev | 5.5E-02 | N/E | N/E | 7.3E-03 |
| g06 | best | N/E | N/E | N/E | N/E |
| | mean | N/E | N/E | N/E | N/E |
| | worst | N/E | N/E | N/E | N/E |
| | st. dev | N/E | N/E | N/E | N/E |
| g07 | best | N/E | 24.306 | 24.306 | N/E |
| | mean | N/E | 24.306 | 24.306 | N/E |
| | worst | N/E | 24.306 | 24.306 | N/E |
| | st. dev | N/E | 7.6E-10 | 1.0E-08 | N/E |
| g08 | best | N/E | N/E | N/E | N/E |
| | mean | N/E | N/E | N/E | N/E |
| | worst | N/E | N/E | N/E | N/E |
| | st. dev | N/E | N/E | N/E | N/E |
| g09 | best | N/E | N/E | N/E | N/E |
| | mean | N/E | N/E | N/E | N/E |
| | worst | N/E | N/E | N/E | N/E |
| | st. dev | N/E | N/E | N/E | N/E |
| g10 | best | N/E | 7049.248 | 7049.248 | N/E |
| | mean | N/E | 7049.248 | 7049.489 | N/E |
| | worst | N/E | 7049.254 | 7060.733 | N/E |
| | st. dev | N/E | 9.8E-04 | 1.6E+00 | N/E |
| g11 | best | N/E | | | N/E |
| | mean | N/E | (14) | (19) | N/E |
| | worst | N/E | [4.0E-06] | [1.3E-05] | N/E |
| | st. dev | N/E | | | N/E |
| g12 | best | N/E | N/E | N/E | N/E |
| | mean | N/E | N/E | N/E | N/E |
| | worst | N/E | N/E | N/E | N/E |
| | st. dev | N/E | N/E | N/E | N/E |
| g13 | best | | | | |
| | mean | (19) | (39) | (50) | (11) |
| | worst | [5.2E-16] | [2.8E-13] | [1.2E-10] | [2.0E-15] |
| | st. dev | | | | |

TABLE IX
STATISTICS RESULTS OBTAINED BY CW WITH DIFFERENT VALUES FOR THE $\lambda$ PARAMETER. N/E MEANS NO EVIDENT EFFECT FOUND BY THIS PARAMETER COMPARED WITH THE RESULTS WHEN USING $\lambda = 10$. (#) DENOTES THE NUMBER OF TRIALS THAT FEASIBLE SOLUTIONS ARE NOT FOUND IN 50 TRIALS. [#] DENOTES THE AVERAGE DISTANCE OF THE POPULATION FROM THE BOUNDARIES OF THE FEASIBLE REGION

| Fcn | status | different parameter values | | | |
|---|---|---|---|---|---|
| | | $\lambda=5$ | $\lambda=8$ | $\lambda=12$ | $\lambda=15$ |
| g01 | best | N/E | N/E | -15.000 | -15.000 |
| | mean | N/E | N/E | -15.000 | -15.000 |
| | worst | N/E | N/E | -15.000 | -14.999 |
| | st. dev | N/E | N/E | 1.4E-06 | 1.3E-04 |
| g02 | best | N/E | N/E | N/E | N/E |
| | mean | N/E | N/E | N/E | N/E |
| | worst | N/E | N/E | N/E | N/E |
| | st. dev | N/E | N/E | N/E | N/E |
| g03 | best | N/E | N/E | N/E | |
| | mean | N/E | N/E | N/E | (23) |
| | worst | N/E | N/E | N/E | [2.0E-16] |
| | st. dev | N/E | N/E | N/E | |
| g04 | best | N/E | N/E | N/E | -30665.54 |
| | mean | N/E | N/E | N/E | -30665.54 |
| | worst | N/E | N/E | N/E | -30665.54 |
| | st. dev | N/E | N/E | N/E | 1.3E-07 |
| g05 | best | N/E | N/E | N/E | 5126.498 |
| | mean | N/E | N/E | N/E | 5126.502 |
| | worst | N/E | N/E | N/E | 5126.740 |
| | st. dev | N/E | N/E | N/E | 3.4E-02 |
| g06 | best | N/E | N/E | N/E | N/E |
| | mean | N/E | N/E | N/E | N/E |
| | worst | N/E | N/E | N/E | N/E |
| | st. dev | N/E | N/E | N/E | N/E |
| g07 | best | N/E | N/E | 24.306 | 24.306 |
| | mean | N/E | N/E | 24.306 | 24.306 |
| | worst | N/E | N/E | 24.306 | 24.306 |
| | st. dev | N/E | N/E | 5.6E-08 | 1.4E-06 |
| g08 | best | N/E | N/E | N/E | N/E |
| | mean | N/E | N/E | N/E | N/E |
| | worst | N/E | N/E | N/E | N/E |
| | st. dev | N/E | N/E | N/E | N/E |
| g09 | best | N/E | N/E | N/E | N/E |
| | mean | N/E | N/E | N/E | N/E |
| | worst | N/E | N/E | N/E | N/E |
| | st. dev | N/E | N/E | N/E | N/E |
| g10 | best | N/E | N/E | 7049.248 | 7049.248 |
| | mean | N/E | N/E | 7049.255 | 7049.679 |
| | worst | N/E | N/E | 7049.612 | 7070.711 |
| | st. dev | N/E | N/E | 5.1E-02 | 3.0E+00 |
| g11 | best | N/E | N/E | | |
| | mean | N/E | N/E | (6) | (16) |
| | worst | N/E | N/E | [3.7E-06] | [2.7E-06] |
| | st. dev | N/E | N/E | | |
| g12 | best | N/E | N/E | N/E | N/E |
| | mean | N/E | N/E | N/E | N/E |
| | worst | N/E | N/E | N/E | N/E |
| | st. dev | N/E | N/E | N/E | N/E |
| g13 | best | N/E | N/E | N/E | |
| | mean | N/E | N/E | N/E | (30) |
| | worst | N/E | N/E | N/E | [3.5E-13] |
| | st. dev | N/E | N/E | N/E | |

[42], this setting does not seem to be absolutely effective for COPs according to the experimental study.

In fact, the setting of the parameter $\varepsilon$ is very simple, and there is a large range of setting for this parameter to obtain better performance for a given COP. It is inconclusive to recommend that, $\varepsilon$ can be an integer between 3 and 6 if $2 \leq n \leq 10$, and $\varepsilon$ can be an integer between 8 and 11 if $10 < n \leq 20$. When we decide this parameter for a given COP, first, we can test each value among the above range based on the dimension of the decision variables by running once, and then we ascertain

which setting has the best performance and use this setting as the unique selection for the parameter $\varepsilon$. According to the above method, in this paper, the parameter $\varepsilon$ is fixed to 8, 11, 6, 3, 4, 5, 6, 4, 5, 6, 3, 3, and 5 for the 13 test functions, respectively.

In essence, SPX resembles Nelder and Mead's method [43] (a recent paper [44] adopts Nelder and Mead's method for constrained optimization). Nelder and Mead's method is known to degrade its performance when adopting a large number of decision variables (e.g., 50 or more), thus, this sort of limitation in Nelder and Mead's method might impact the performance of SPX when tackling COPs with large scale. A possible way to overcome this limitation is to adopt or design a more effective parameter-independent crossover operator, a research direction which can serve as a basis for future work.

In addition, the core procedures of our method are the two computation models, as analyzed, Models 1 and 2 are similar to $(\mu + (m', \lambda))$ ES and $(\mu + (1, \lambda))$ ES, respectively. Thus, our method can be used to to solve unconstrained problems directly, since in this case we only need to set $G(\vec{x}) = 0$ and the comparison of individuals is only based on $f(\vec{x})$.

## V. Conclusion and Future Work

Based on multiobjective optimization techniques, a novel COEA is presented in this paper. In our algorithm, an individual in the parent population may be replaced if it is dominated by a nondominated individual in the offspring population. In addition, two detached but interactive operators are introduced, one is the set of models of a population-based algorithm-generator, and the other is the ISARM. COEAs have two definite goals that can be achieved by the above two operators as expected.

From the experimental results, it is evident that the approach presented in this paper has substantial potential in handling various COPs, and its performance remarkably outperforms other algorithms in many respects, though the results of other algorithms are statistically competitive. Meanwhile, it can meet the feasible optima of all test cases when there is no transformation of equality constraints. The standard deviations also reveal the strong robustness of our algorithm.

Since the test functions used in this paper are still far from embodying a complete COP test suite, a more profound study in designing a more representative test function set in this field is absolutely necessary in future work. Another direction of future work is to overcome the limitation of our algorithm discussed in Section IV-H. As discussed, the main motivation to use SPX is its simplicity and efficiency. However, there also exist many other excellent crossover operators for real- parameter optimization, such as UNDX and PCX [45]. In our future work, we may do a thorough study comparing SPX with other crossover operators. We are also considering the possibility of extending this algorithm so that it can deal with other types of COPs, such as constrained combinational optimization problems and MOPs with constraints.

## Acknowledgment

The authors sincerely thank the anonymous reviewers, the associate editor and Prof. X. Yao for their valuable and constructive comments and suggestions. They also gratefully acknowledge Dr. J. Cai and Dr. Q. Cai for improving the presentation of this paper.

## References

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

[2] C. A. C. Coello, "Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art," *Comput. Methods Appl. Mech. Eng.*, vol. 191, no. 11–12, pp. 1245–1287, Jan. 2002.

[3] Z. Michalewicz, K. Deb, M. Schmidt, and T. Stidsen, "Test-case generator for constrained parameter optimization techniques," *IEEE Trans. Evol. Comput.*, vol. 4, no. 3, pp. 197–215, Aug. 2000.

[4] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 2, pp. 62–87, Apr. 1997.

[5] H. Lu and G. G. Yen, "Rank-density-based multiobjective genetic algorithm and benchmark test function study," *IEEE Trans. Evol. Comput.*, vol. 7, no. 4, pp. 325–343, Aug. 2003.

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

[7] S. B. Hamida and M. Schoenauer, "ASCHEA: New results using adaptive segregational constraint handling," in *Proc. Congr. Evol. Comput.*, May 2002, vol. 1, pp. 884–889.

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

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

[10] J. A. Wright and R. Farmani, "Genetic algorithm: A fitness formulation for constrained minimization," in *Proc. Genetic Evol. Comput. Conf.*, San Francisco, CA, 2001, pp. 725–732.

[11] J. X. Yu, X. Yao, C. Choi, and G. Gou, "Materialized view selection as constrained evolutionary optimization," *IEEE Trans. Syst., Man, Cybern. C*, vol. 33, no. 4, pp. 458–467, Aug. 2003.

[12] S. Koziel and Z. Michalewicz, "Evolutionary algorithm, homomorphous mappings, and constrained parameter optimization," *Evol. Comput.*, vol. 7, no. 1, pp. 19–44, 1999.

[13] D. Powell and M. M. Skolnick, "Using genetic algorithm in engineering design optimization with nonlinear constraint," in *Proc. 5th Int. Conf. Genetic Algorithms*, S. Forrest, Ed., San Mateo, CA, Jul. 1993, pp. 424–431.

[14] K. Deb, "An efficient constraint handling method for genetic algorithms," *Comput. Methods Appl. Mech. Eng.*, vol. 186, no. 2/4, pp. 311–338, 2000.

[15] F. Jiménez and J. L. Verdegay, "Evolutionary techniques for constrained optimization problems," in *Proc. 7th Eur. Congr. Intell. Tech. Soft Comput.*, 1999.

[16] 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.

[17] 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.

[18] ——, "Search biases in constrained evolutionary optimization," *IEEE Trans. Syst., Man, Cybern. C*, vol. 35, no. 2, pp. 233–243, May 2005.

[19] C. A. C. Coello, "Treating constraints as objectives for single-objective evolutionary optimization," *Eng. Opt.*, vol. 32, no. 3, pp. 275–308, 2000.

[20] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," in *Proc. 1st Int. Conf. Genetic Algorithms and Their Applications*, J. J. Grefenstette, Ed., Hillsdale, NJ, 1985, pp. 93–100.

[21] C. A. Coello Coello and E. Mezura-Montes, "Constraint-handling in genetic algorithms through the use of dominance-based tournament Selection," *Advanced Eng. Inform.*, vol. 16, no. 3, pp. 193–203, 2002.

[22] J. Horn, N. Nafpliotis, and D. Goldberg, "A niched Pareto genetic algorithm for multiobjective optimization," in *Proc. 1st IEEE Conf. Evol. Comput., World Congr. Comput. Intell.*, June 1994, vol. 1, pp. 82–87.

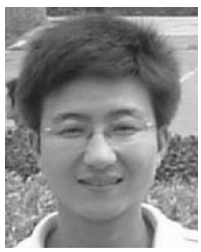
[23] C. A. C. Coello, "Constraint handling using an evolutionary multiobjective optimization technique," *Civil Eng. Environ. Syst.*, vol. 17, pp. 319–346, 2000.

[24] C. M. Fonseca and P. J. Fleming, "Multiobjective optimization and multiple constraint handling with evolutionary algorithms—Part I: A unified formulation," *IEEE Trans. Syst., Man, Cybern.*, vol. 28, no. 1, pp. 26–37, Jan. 1999.
[25] P. D. Surry and N. J. Radcliffe, "The COMOGA method: Constrained optimization by multiobjective genetic algorithm," *Control Cybern.*, vol. 26, no. 3, pp. 391–412, 1997.
[26] A. H. Aguirre, S. B. Rionda, C. A. C. Coello, G. L. Lizáraga, and E. Mezura-Montes, "Handling constraints using multiobjective optimization concepts," *Int. J. Numerical Methods in Eng.*, vol. 59, no. 15, pp. 1989–2017, Apr. 2004.
[27] J. D. Knowles and D. W. Corne, "Approximating the nondominated front using the Pareto archived evolutionary strategy," *Evol. Comput.*, vol. 8, no. 2, pp. 149–172, 2000.
[28] E. Mezura-Montes and C. A. C. Coello, A numerical comparison of some multiobjective-based techniques to handle constraints in genetic algorithms, Evol. Comput. Group, CINVESTAV, Sección de Computación, Dept. de Ingeniería Eléctrica, CINVESTAV-IPN, México D.F., México, Tech. Rep. EVOCINV-01-2003.
[29] K. Deb, *Multiobjective Optimization Using Evolutionary Algorithms*. Chichester, U.K.: Wiley, 2001.
[30] K. Deb, A. Pratab, S. Agrawal, and T. Meyarivan, "A fast and elitist nondominated sorting genetic algorithm for multi-objective optimization: NSGA II," *IEEE Trans. Evol. Comput.*, vol. 6, pp. 182–197, Apr. 2002.
[31] E. Zitzler, M. Laumannns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization," in *Proc. EUROGEN 2001-Evolutionary Methods for Design. Optimization and Control with Applications to Industrial Problem*, K. C. Giannakoglou, Ed., 2001, pp. 95–100.
[32] E. Mezura-Montes and C. A. C. Coello, On the usefulness of the evolution strategies self-adaptation mechanism to handle constraints in global optimization, Evol. Comput. Group, CINVESTAV, Sección de Computación, Dept. de Ingeniería Eléctrica, CINVESTAV-IPN, México D.F., México, Tech. Rep. EVOCINV-01-2003. [Online]. Available: http://www.cs.cinvestav.mx/constraint/
[33] T. Asselmeyer, W. Ebeling, and H. Rośe, "Evolutionary strategies of optimization," *Phys. Rev. E*, vol. 56, no. 1, pp. 1171–1180, Jul. 1997.
[34] E. Mezura-Montes, "Alternative techniques to handle constraints in evolutionary optimization," Ph.D. dissertation, Dept. Comput. Sci., CINVESTAV-IPN, México, 2004.
[35] H. Kita, "A comparison study of self-adaptation in evolution strategies and real-coded genetic algorithms," *Evol. Comput.*, vol. 9, no. 2, pp. 223–241, 2001.
[36] K. Deb, A population-based algorithm-generator for real-parameter optimization, KanGAL Rep. 2003003.
[37] S. Tsutsui, M. Yamamure, and T. Higuchi, "Multi-parent recombination with simplex crossover in real coded genetic algorithms," in *Proc. Genetic and Evol. Comput. Conf.*, 1999, pp. 657–664.
[38] T. Ray and K. M. Liew, "Society and civilization: An optimization algorithm based on the simulation of social behavior," *IEEE Trans. Evol. Comput.*, vol. 7, no. 4, pp. 386–396, Aug. 2003.
[39] 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.
[40] A. Hedar and M. Fukushima, Derivative-free filter simulated annealing method for constrained continuous global optimization, Dept. Appl. Math. Physics, Kyoto Univ., Kyoto, Japan, Tech. Rep. 2004-007.
[41] E. Mezura-Montes and C. A. Coello Coello, A simple multimembered evolution strategy to solve constrained optimization problems, Evol. Comput. Group, CINVESTAV, Sección de Computación, Dept. de Ingeniería Eléctrica, CINVESTAV-IPN, México D.F., México, Tech. Rep. EVOCINV-04-2003. [Online]. Available: http://www.cs.cinvestav.mx/constraint/
[42] T. Higuchi, S. Tsutsui, and M. Yamamura, "Theoretical analysis of simplex crossover for real-coded genetic algorithms," *Parallel Problem Solving from Nature (PPSN-6)*, pp. 365–374, 2000.
[43] J. A. Nelder and R. Mead, "A simplex method for function minimization," *Comput. J.*, vol. 7, no. 4, pp. 308–313, 1965.
[44] T. Takahama and S. Sakai, "Constrained optimization by applying the $\alpha$ constrained method to the nonlinear simplex method with mutations," *IEEE Trans. Evol. Comput.*, vol. 9, no. 5, pp. 437–451, Oct. 2005.
[45] K. Deb, A. Anand, and D. Joshi, "A computationally efficient evolutionary algorithm for real-parameter optimization," *Evol. Comput.*, vol. 10, no. 4, pp. 371–395, 2002.

**Zixing Cai** (SM'98) received the Diploma degree from the Department of Electrical Engineering, Jiao Tong University, Xi'an, China, in 1962.

He has been teaching and doing research in the College of Information Science and Engineering, Central South University (CSU), Changsha, Hunan, China, since 1962. From May 1983 to December 1983, he visited the Center of Robotics, Department of Electrical Engineering and Computer, Science, University of Nevada, Reno. Then, he visited the Advanced Automation Research Laboratory, School of Electrical Engineering, Purdue University, West Lafayette, IN, from December 1983 to June 1985. From October 1988 to September 1990, he was a Senior Research Scientist in the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Science and the National Laboratory of Machine Perception, Center of Information, Beijing University. From September 1992 to March 1993, he visited the Center for Intelligent Robotic Systems for Space Exploration, Department of Electrical, Computer and System Engineering, Rensselaer Polytechnic Institute, Troy, NY, as a visiting Professor. From April 2004 to July of 2004, he visited the Institute of Informatics and Automation, Russia Academy of Sciences. Since February 1989, he has become an Expert of United Nations granted by UNIDO. His research interests include intelligent system, artificial intelligence, intelligent computation, and robotics. Over 500 papers and 26 books/textbooks have been published.

Dr. Cai received over 30 State, Province, and University awards in science, technology and teaching. One of the newest prizes is the State Eminent Professor Prize of China.



**Yong Wang** was born in Hubei, China, on December 26, 1980. He is working towards the Ph.D. degree in the College of Information Science and Engineering, Central South University, Changsha, Hunan, China.

He has published over ten academic papers since 2001. His current research interests include evolutionary computation, constrained optimization, multiobjective optimizaiton and their applications in multiple mobile robots.