Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2014, Article ID 606019, 15 pages
http://dx.doi.org/10.1155/2014/606019

Research Article

A Memetic Differential Evolution Algorithm Based on Dynamic Preference for Constrained Optimization Problems

Ning Dong[1] **and Yuping Wang**[2]

[1] *School of Mathematics and Statistics, Xidian University, Xi'an 710071, China*
[2] *School of Computer Science and Technology, Xidian University, Xi'an 710071, China*

Correspondence should be addressed to Ning Dong; dongning@snnu.edu.cn

Received 9 February 2014; Revised 8 May 2014; Accepted 14 May 2014; Published 4 June 2014

Academic Editor: Hak-Keung Lam

The constrained optimization problem (COP) is converted into a biobjective optimization problem first, and then a new memetic differential evolution algorithm with dynamic preference is proposed for solving the converted problem. In the memetic algorithm, the global search, which uses differential evolution (DE) as the search scheme, is guided by a novel fitness function based on achievement scalarizing function (ASF). The novel fitness function constructed by a reference point and a weighting vector adjusts preference dynamically towards different objectives during evolution, in which the reference point and weighting vector are determined adapting to the current population. In the local search procedure, simplex crossover (SPX) is used as the search engine, which concentrates on the neighborhood embraced by both the best feasible and infeasible individuals and guides the search approaching the optimal solution from both sides of the boundary of the feasible region. As a result, the search can efficiently explore and exploit the search space. Numerical experiments on 22 well-known benchmark functions are executed, and comparisons with five state-of-the-art algorithms are made. The results illustrate that the proposed algorithm is competitive with and in some cases superior to the compared ones in terms of the quality, efficiency, and the robustness of the obtained results.

1. Introduction

In many science and engineering fields, there often occurs a kind of optimization problems which are subject to different types of constraints, and they are called constrained optimization problems (COPs). Due to the presence of constraints, COPs are well known as a challenging task [1] in optimization fields. Evolutionary algorithms (EAs), inspired by nature, are a class of stochastic and population-based optimization techniques, which have been widely applied to solve global optimization problems. In the last decades, researchers applied EAs for COP and proposed a large number of constrained optimization EAs (COEAs) [2–4].

As is well known, the goal of COEAs is to find the optimal solution which satisfies all constraints. So constraint satisfaction is the primary requirement and the technique of constraints handling will greatly affect the performance of COEAs. The most common methods to handle constraints are penalty function based methods [4, 5], which transform a COP into an unconstrained one by adding a penalty term into the original objective function in order to penalize the infeasible solutions. The main drawback of the methods based on penalty function is to preset an appropriate penalty factor, which greatly influences the performance of these methods. However, as indicated in [2], deciding an optimal penalty factor is a very difficult problem. To avoid the use of penalty factor, Powell and Skolnick [6] suggested a fitness function, which maps feasible solutions into the interval $(-\infty, 1)$ and infeasible solutions into the interval $(1, +\infty)$. The method considered that the feasible solutions are always superior to infeasible ones. Inspired by Powell and Skolnick [6], Deb [3] proposed three feasibility rules for binary tournaments selection without introducing new parameters. The binary tournament selection based on the three feasibility rules is as follows: (1) any feasible solution is preferred to any infeasible one; (2) between two feasible solutions, the one with better objective value is preferred; (3) between two infeasible solutions, the one with smaller constraint violation is preferred. However, the feasibility rules always prefer feasible solutions to infeasible ones, which makes the promising infeasible solutions which carry more important information than its feasible counterparts have little opportunity to go into next population. Therefore, the diversity of the population degrades, which makes the algorithm prone to trapping into local optima.

In recent years, some researchers suggested transforming a COP into a biobjective optimization problem based on the penalty function [7–9]. For the converted biobjective optimization problem, if the Pareto dominance is used as the unique strategy for individuals comparison, then the objective function and constraint violation function are seen of the same significance. This will result in the situation that some solutions being far away from the feasible region but with small objective function value will be retained in the evolution process. However, these solutions have little help in searching the optimal solution of the original problem, especially for problems whose optimal solutions are inside the feasible region.

From the above analysis, it can be concluded that not only the methods which consistently prefer feasible solutions to infeasible ones are arguable, but also the methods which treat the objective function as equally important as the constraint satisfaction are ineffective. Instead, the methods that can balance the objective function and the constraint satisfaction will be effective. Nevertheless, how to balance both objectives during the evolution needs to be further considered carefully.

Runarsson and Yao [2] proposed the stochastic ranking (SR) method which used a comparison probability P_f to balance the objective function and constraint violation. The SR based comparison prefers some good infeasible solutions with the probability $1-P_f$, which not only makes the promising infeasible solutions have the opportunity to survive into the next population, but also enhances the diversity of the population. Wang et al. [9] proposed an adaptive tradeoff model (ATM) with evolution strategy to solve COP, which can adaptively adjust the proportion of infeasible solutions survived into the next generation. Takahama and Sakai [10] transformed a COP into an unconstrained numerical optimization problem with novel constraint-handling techniques, so called ε-constrained method. The method relaxes the limit to consider a solution as feasible, based on its sum of constraint violation, with the aim of using its objective function value as a comparison criterion whereas the extent of the relaxation is dynamic in the evolution.

As is indicated in [11], besides the constraints handling technique, the performance of COEAs depends on another crucial factor: the search mechanism of EAs. The current popular search algorithms involve evolution strategy (ES) [2, 9, 12], particle swarm optimization (PSO) [13], differential evolution (DE) [14–17], electromagnetism-like mechanism algorithm [18], and so forth. Among these search algorithms, DE is a more recent and simple search engine. In competition on constrained real-parameter optimization at IEEE CEC 2006 Special session, among the top five best algorithms, there are three algorithms with DE as a search engine, which shows the efficiency of DE and much attention has been attracted by DE in various practical cases. Wang and Cai [19] combined multiobjective optimization with DE to solve COP, in which DE was used as a search mechanism, and Pareto dominance was used to update the evolution population. Zhang et al. [20] proposed a dynamic stochastic selection scheme based on SR [2] and combined it with the multimember DE [21]. Jia et al. [22] presented an improved $(\mu + \lambda)$ DE for COPs, in which a multimember DE [21] is also used as the search engine, and an improved ATM [9] version is applied as the comparison criterion between individuals. In recent years, the combination of different DE variants and the adaptive parameters setting is popular in the algorithm designing [1, 23, 24]. In [1], the modified basic mutation strategy is combined with a new directed mutation scheme, which is based on the weighting difference vector between the best and the worst individual at a particular generation. Moreover, the scaling factor and crossover rate are also dynamically set to balance the global search and local search. Huang et al. [23] also proposed a self-adaptive DE for COPs. In the methods, the mutation strategies and the parameters settings are self-adapted during the evolution. Elsayed et al. [24] proposed an evolutionary framework that utilizes existing knowledge to make logical changes for better performance. In the proposed algorithm, 8 mutation schemes are tested on a suit of benchmark instances, and the performance of each scheme is analyzed. Then a self-adaptive multistrategy differential evolution algorithm, using the best several variants, is exhibited. The proposed algorithm divides the population into a number of subpopulations, where each subpopulation evolves with its own mutation and crossover, and also the population size, scaling factor, and crossover rate are adaptive to the search progresses. Furthermore, the SQP based local search is used to speed up the convergence of the algorithm. Moreover, DE or its mutation operator is also combined with other algorithms to improve the performance of methods [11, 25]. In [11], Runarsson and Yao improved SR [2] by combining ES with the mutation operator adopted in DE, which results in great promotion in the quality of solutions. In [25], the authors proposed biogeography-based optimization (BBO) approaches for COP, in which one version is combined with DE mutation operators, instead of the conventional BBO-based operators. The experiment results show that the version combined with DE mutation operators outperforms the other variants of BBO, which confirms the efficiency of DE operators.

In this paper, we proposed a memetic differential evolution algorithm with dynamic preference (MDEDP) to solve COP. In the proposed method, DE is used as the global search engine, and the simplex crossover (SPX) is incorporated to encourage the local search in the promising region in the search space. In the constraint-handling technique, COP is first transformed into a biobjective optimization problem with original objective function and constraint violation function as two objectives, in which the constraint violation function is constructed by constraints. Then, for the converted biobjective problem, an achievement sacralizing function (ASF) based fitness function is presented to balance the objective function and constraints violation in the evolution. The reference point and the weighting vector used in constituting the ASF are dynamically adopted to achieve a good balance between objective and constraint violation function.

The rest of this paper is organized as follows. COP transformation and some related concepts are briefly introduced in Section 2. In Section 3, DE algorithm and simplex crossover operator are briefly presented. The novel memetic DE with dynamic preference for COP is proposed in Section 4. Section 5 gives experimental results and comparisons with other state-of-the-art algorithms for 22 standard test problems. Finally, we summarize the paper in Section 6.

2. COPs and Some Related Concepts

2.1. Constrained Optimization Problem. In general, a constrained optimization problem can be stated as follows:

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & g_i(x) \le 0, \quad i = 1, 2, \ldots, p \\ & h_j(x) = 0, \quad j = p+1, \ldots, q, \end{aligned} \tag{1}$$

where $x = (x_1, x_2, \ldots, x_n) \in S$ is decision variable and S is an n-dimension rectangular search space in R^n defined by $S = \{x \mid l_i \le x_i \le u_i, i = 1, 2, \ldots, n\}$. $f : S \to R$ is an objective function, $g_i(x) \le 0$ $(i = 1, 2, \ldots, p)$ are inequality constraints, and $h_j(x) = 0$ $(j = p+1, \ldots, q)$ are equality constraints. The feasible region $\Omega \subset S$ is defined as $\Omega = \{x \in S \mid g_j(x) \le 0,\ j = 1, \ldots, p,\ h_j(x) = 0,\ j = p+1, \ldots, q\}$. Solutions in Ω are feasible solutions.

2.2. Problem Transformation and Related Concepts. Penalty function methods are the common constraints handling technique for solving COPs; however, the methods are sensitive to the penalty factor, and the tuning of penalty parameters is very difficult [2]. Reformulating a constrained optimization problem as a biobjective problem is a recently effective constraints handling method [7–9]. In general, the degree of constraint violation of x on the ith constraint is defined as

$$G_i(x) = \begin{cases} \max\{0, g_i(x)\}, & i = 1, 2, \ldots, p \\ \max\{0, |h_i(x)| - \delta\}, & i = p+1, \ldots, q, \end{cases} \tag{2}$$

where δ is a small positive tolerance value for equality constraints. Then $G(x) = \sum_{i=1}^{q} G_i(x)$ reflects the degree of violation of all constraints at x and also denotes the distance of x to a feasible region. Obviously, $G(x) \ge 0$.

Based on the constraint violation function $G(x)$, COP is converted into a biobjective optimization problem, which minimizes the original objective function $f(x)$ and the constraint violation function $G(x)$ simultaneously. The converted biobjective problem is as follows:

$$\begin{aligned} \min \quad & (f(x), G(x)) \\ \text{s.t.} \quad & x \in S. \end{aligned} \tag{3}$$

For simplicity, the two objectives of (3) are denoted by $f_1(x) = f(x)$, $f_2(x) = G(x)$.

Though COP is converted into a biobjective optimization problem (3), they are different essentially (see Figure 1).

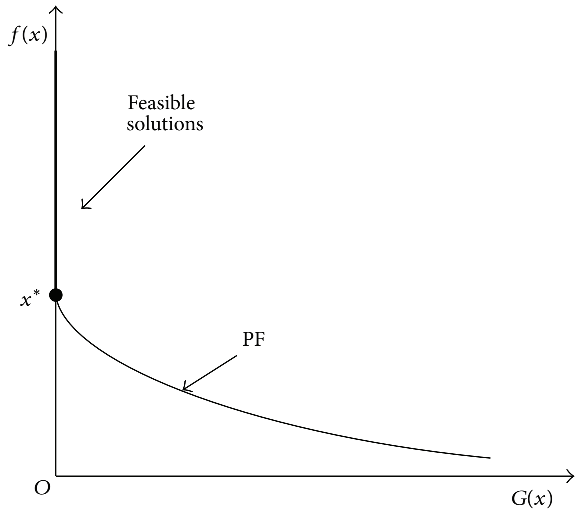

Figure 1: Relationship of solutions between COPs (1) and biobjective problem (3).

Biobjective problem (3) intends to find a set of representation solutions uniformly distributed on Pareto front (PF), while COP (1) is to find a solution which satisfies all constraints ($G(x) = 0$) and also minimizes $f(x)$, that is, to find the point in the intersection of PF and the feasible region.

3. Differential Evolution and Simplex Crossover Operator

3.1. Differential Evolution [26]. Differential evolution, proposed by Storn and Price [26], is a parallel direction search method and also follows the general procedure of an EA. For its simplicity and efficiency, DE has been widely employed to solve constraints optimization problem [1, 14–16, 19–21]. In the DE process, initial population P with NP individuals is produced randomly in decision space. For each target vector $x_i \in P$ $(i = 1, 2, \ldots, \text{NP})$ at each generation, DE then employs the three operations below in turn (there are more than 10 variants of DE schemes in the related references; in this paper, the most often used DE scheme "DE/rand/1/bin" is employed in the search process and its details are as follows: where "rand" denotes that the vector to be mutated is randomly selected in the current population, "1" specifies the number of difference vectors, and "bin" denotes the binomial crossover scheme).

Mutation. A trial vector $v_i = (v_{i,1}, v_{i,2}, \ldots, v_{i,n})$ for each target vector $x_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,n})$ is generated based on the current parent population via the "rand/1" mutation strategy:

$$v_i = x_{r1} + F \cdot (x_{r2} - x_{r3}), \tag{4}$$

where $r1$, $r2$, and $r3$ are mutually different integers randomly selected from $\{1, 2, \ldots, \text{NP}\} \setminus \{i\}$ and $F \in [0, 2]$ is a scaling factor which controls the amplification of the differential variation $(x_{r2} - x_{r3})$.

Crossover. To increase the diversity, the offspring vector $u_i = (u_{i,1}, u_{i,2}, \dots, u_{i,n})$ is then generated by the binomial crossover between target vector and trial vector, in which

$$u_{i,j} = \begin{cases} v_{i,j}, & \text{if rand}(0,1) < \text{Cr or } j = j_{\text{rand}} \\ x_{i,j}, & \text{else} \end{cases} \tag{5}$$

for $j = 1, 2, \dots, n$, where $\text{Cr} \in [0, 1]$ is crossover probability and $j_{\text{rand}} \in \{1, 2, \dots, n\}$ is a randomly chosen index, which ensures that offspring u_i is different from its parent x_i in at least one component.

Selection. The offspring u_i is compared to its parent x_i using greedy criterion, and the one with smaller fitness will be selected into the next population.

3.2. Simplex Crossover Operator (SPX) [27]. Simplex crossover (SPX) is one of the most common used recombination operators in EAs, which generates offspring based on uniform distribution. The search region of SPX is adaptively adjusted with the evolution, which ensures that SPX has the good ability of exploration in the earlier stages and the good ability of exploitation in the later stages of evolution. Moreover, SPX is very simple and easy to realize.

In R^n, $n + 1$ mutually independent parent vectors x_i $(i = 1, 2, \dots, n+1)$ form a simplex. Offspring is generated through the following two steps: (1) expand the original simplex along each direction $x_i - o$ $(i = 1, 2, \dots, n + 1)$ by $(1 + \varepsilon)$ $(\varepsilon > 0)$ times to form a new simplex, where o is the center of original simplex, that is, $o = \sum_{i=1}^{n+1} x_i/(n + 1)$ and the vertexes of the new simplex are $y_i = (1 + \varepsilon)(x_i - o)$, $(i = 1, 2, \dots, n + 1)$ and (2) choose a point uniformly from the new simplex as the offspring, that is, offspring $z = k_1 y_1 + k_2 y_2 + \cdots + k_{n+1} y_{n+1} + o$, where k_i $(i = 1, 2, \dots, n+1)$ is a random number in $[0, 1]$ and satisfies the condition $k_1 + k_2 + \cdots + k_{n+1} = 1$.

In general, the SPX is specified as SPX-n-m-ε, where n is the dimension of search space, m is the number of parents to constitute the simplex which is selected in $[2, n + 1]$, and ε is a control parameter that defines the expanding rate. In the above procedure of producing offspring, the number of parents m is set to $n + 1$, and the crossover scheme SPX is denoted by SPX-n-$(n + 1)$-ε.

4. Memetic DE Based on Dynamic Preference for Solving COPs

4.1. Novel Fitness Function Based on Dynamic Preference. Pareto dominance is the most often used rule in multiobjective optimization to sort individuals. However, if two individuals are not dominated by each other, the better one of them cannot be determined. To solve the converted biobjective problem (3), we intend to find the solution which not only satisfies the constraints $(G(x) = 0)$, but also optimizes the original objective function $f(x)$; therefore, problem (3) is substantially a biobjective problem with preference. Among methods in solving multiobjective problem, weighted metrics method [28] scalarizes the multiobjectives into a single objective by the weighting distance from individuals to a reference point. One of the single objective functions obtained by scalarizing is called achievement scalarizing function (ASF) [28], which implies the preference to different objectives via different weighting vectors and reference points. For general biobjective optimization problems, ASF is defined as follows:

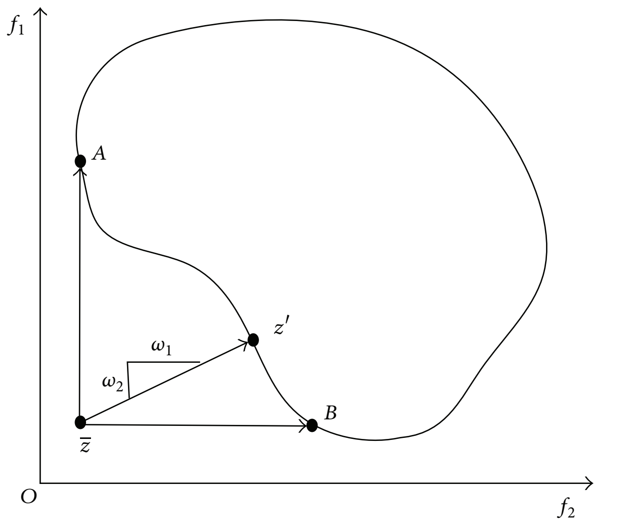

FIGURE 2: Illustration of weighted metric function ASF.

$$F(x) = \max_{i=1,2} \{\omega_i (f_i(x) - \overline{z}_i)\}, \tag{6}$$

where $\overline{z} = (\overline{z}_1, \overline{z}_2)$ is a reference point (it may be inside the objective space or not), which determines the preferred region on Pareto front (curve between A and B on PF), $\omega = (\omega_1, \omega_2)$, satisfying $\omega_1, \omega_2 \geq 0$ and $\omega_1 + \omega_2 = 1$, is a weighting vector, which points to the certain Pareto optimal point z' in the preferred region (see Figure 2). From Figure 2, $F(x)$ is the ω-weighted distance from individual x to the reference point $\overline{z}$, and the value ω_1, ω_2 realizes the extent of preference to the two objectives f_1 and f_2. $\omega_1 > \omega_2$ means that ASF (4) prefers the Pareto optimal solution with small f_1 between A and B, and conversely, it prefers the solution with small f_2. The advantage of ASF is that arbitrary (weakly) Pareto optimal solution can be obtained by moving the reference point $\overline{z}$ only. Moreover, the different (weakly) Pareto optimal solutions between A and B can be found by various weighting vectors [28].

For the converted biobjective problem, the evolution procedure usually experiences three stages. In the first stage, there is no feasible solution in the population; therefore, the individuals with small constraint violations should be selected in priority to help the population be close to the feasible region. With the population evolution, some individuals go into the feasible region. In this stage, the feasible individuals are preferred, and some nondominated infeasible individuals with small constraints violation are also maintained to increase the diversity. In the third stage, there are only feasible individuals in population, and the feasible individuals with small objective function value are preferred. According to the characteristics of ASF and the above analysis, in different stages of evolution, the proper reference

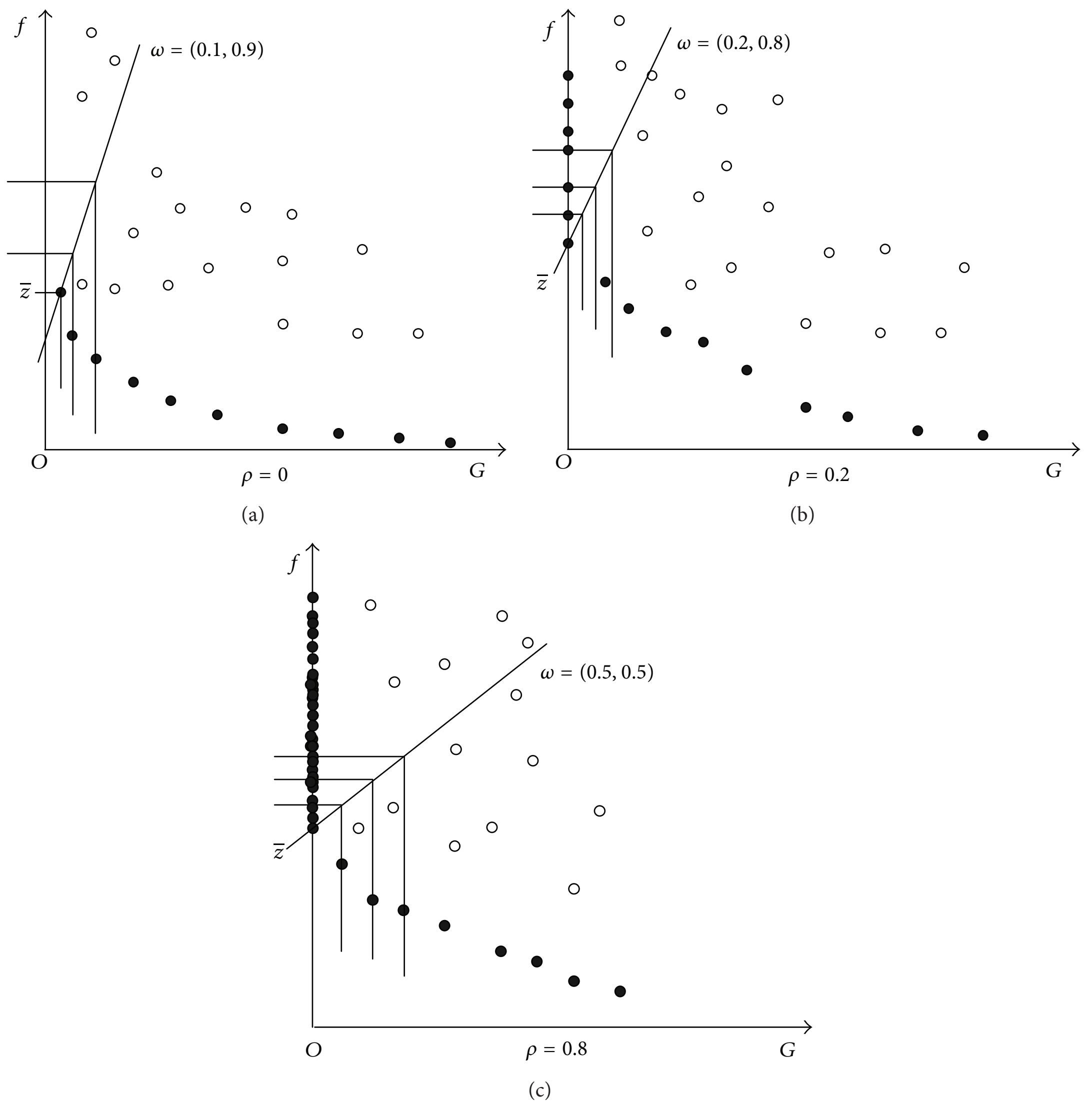

FIGURE 3: Contours of the fitness function in different stages and the selection.

point $\overline{z}$ and weighting vector ω should be chosen adaptively to construct the fitness function $F(x)$ and realize the preference to different objectives.

In the first stage, there are only infeasible solutions in the population; that is, $\rho = 0$ (where ρ is the proportion of feasible solutions in population). Therefore, constraint satisfaction is more important than objective minimization in the individuals comparison. However, if constraint satisfaction is the only condition considered, dominated individuals with small constraint violations and large objective function value will also access into the next population with priority, and these individuals have little effect on finding the optimal solution. On the other hand, the individuals with smaller objective function value and slightly bigger constraint violations would be neglected. Furthermore, some algorithms adopted Pareto ranking which put the same importance on all objectives and assigned the same best fitness value to all the Pareto optimal solutions in population first. But, for some problems (e.g., problems whose optimal solution is inside the feasible region), infeasible individuals which are far away from the boundaries of the feasible region have little effect on searching optimal solution. Thus, in this stage, the small constraint violation should be guaranteed in priority, and meanwhile the objective function value should be guaranteed not too large. In the absence of feasible solution in the population, the nondominated infeasible solution with the smallest constraint violation (it is also called the best infeasible solution) is the best solution, and it is chosen as the reference point $\overline{z}$. Moreover, in this stage, the constraint satisfaction is a key issue; thus, the weighting vector can be set to $\omega = (0.1, 0.9)$. This choice of the reference point and the weighting vector demonstrates that the smaller the weighting distance from an individual to the reference point, the better the individual. Figure 3(a) illustrates the fitness value of individuals in population.

In the second stage, there are both feasible and infeasible solutions in the population; that is, $0 < \rho < 1$. In this stage, the evolution should adjust preference according to the proportion of feasibility in current population. If the proportion is low, it is possible that the evolution is in the early time of the search, or the feasible region is very small compared to the search space and it is difficult to obtain feasible individuals in the evolution. Therefore, the feasible individuals are preferred. If the proportion is high, some nondominated infeasible individuals with small constraint violations are more useful than some feasible individuals with large objective function value, especially for the problem whose optimal solution is located exactly on or nearby the boundaries of the feasible region. From the above analysis, the reference point $\overline{z}$ is determined as the best feasible individual, and the weighting vector is determined according to the proportion ρ. If the proportion ρ is small, the evolution should prefer the individuals with small value of $G(x)$ (i.e., individuals which are feasible or with small constraint violations). Conversely, if ρ is big, the evolution should prefer the individuals with small value of $f(x)$. Because the original COP is to find the optimal solution which satisfies all constraints, then the constraint satisfaction is primary, so the preference to the objective function cannot be too large, and the maximum of the preference weighting value to the objective function is set to 0.5; that is, $\omega_1^{\max} = 0.5$. The weighting vector can be determined as $\omega_1 = \min\{\rho, \omega_1^{\max}\}$, and $\omega_2 = 1 - \omega_1$. Figures 3(b) and 3(c) illustrate the preference in this stage.

In the third stage, there are only feasible solutions in the population; that is, $\rho = 1$. It is obvious that the comparison of individuals is based on their objective function values. Actually, the criterion is the special case of formula (4), in which $\overline{z}$ is the best feasible solution and weighting vector $\omega = (1, 0)$.

In the comparison among individuals, $F(x)$ is used as the fitness function. The smaller the value of $F(x)$, the better the individual x. Fitness function $F(x)$ is the weighted distance from individual to the reference point which is the best in the population. Obviously, the fitness $F(x)$ is nonnegative and has the minimal value 0 at the best individual; therefore, the elitist strategy is taken to prevent the population from degenerating. To avoid the biases in $F(x)$ caused by the magnitude of $f_1(x)$ and $f_2(x)$, both of them are normalized firstly. For simplicity, we still use $f_1(x)$ and $f_2(x)$ to denote the normalized objective functions and

$$f_1(x) = \frac{f(x) - f_{\min}}{f_{\max} - f_{\min}}, \qquad f_2(x) = \frac{G(x) - G_{\min}}{G_{\max} - G_{\min}}, \qquad (7)$$

where $f_{\max}$, $f_{\min}$, $G_{\max}$, and $G_{\min}$ are the maximum and minimum values of $f(x)$ and $G(x)$, respectively. Besides the minimum of $G(x)$ is known as 0, and the other three values are unknown. Therefore, $f_{\max}$, $f_{\min}$, and $G_{\max}$ are updated by the maximum and minimum values of $f(x)$ and $G(x)$ in the population. Moreover, in the normalization, we do not use the minimum value of $G(x)$ in the population because if there is no feasible solution, the infeasible solution with smallest constraint violation will be normalized to feasible one.

4.2. Local Search Based on SPX. In this paper, DE/rand/1/bin is used as the global search engine, which has good ability of exploration in the early phases, and can explore the search space fully. However, its exploitation ability is weak in the early stage, which results in slow convergence. To balance the exploration and exploitation in optimizing process, local search (LS) based on simplex crossover (SPX) is employed. For many COPs in practice, especially those with equality constraints, the optimal solution is situated on the boundary of the feasible region or nearby. Thus searching from both sides of the boundary of the feasible region is effective. From Figure 1, one can conclude that the solutions in the lower-left part of the objective space, which are feasible solutions with small objective values or nondominated infeasible ones with small constraint violation, are promising. Thus, the neighborhood embraced by the promising solutions and nearby deserves more computing resources. Therefore, the LS will be efficient when focusing on the promising region. Besides the neighborhood the LS is concentrating on, other issues influencing the LS are the solutions on which the LS engine will work and the number of solutions used to participate the LS. From the conclusion in [27], SPX with a large number of parents, m, has sampling biases that reflect biases in the population distribution too much. So a medium number of parents are a good choice to overcome the oversampling biases. Furthermore, [27] concluded that SPX works well with $m = 3$ on a low dimensional function and $m = 4$ on high dimensional functions. Based on the conclusion, in the proposed SPX based local search, the number of parents, m, is set to 3. Furthermore, the three parents selected to perform SPX in LS are demonstrated in Figure 4.

In the first stage, there is no feasible solution in the population; that is, $\rho = 0$. The nondominated solutions with small constraint violation are located in the lower-left part in the objective space; therefore, the top three nondominated solutions with small constraint violation are selected as parents x_1, x_2, and x_3 to perform SPX (Figure 4(a)).

In the second stage, there are feasible and infeasible solutions simultaneously in the population; that is, $0 < \rho < 1$. If there is only one feasible solution, it is the first selected parent x_1. And then, the solutions which are nondominated with it are checked. If there is no nondominated solution, then the top two solutions with small constraint violation are selected as the second parent and the third parent x_2 and x_3 (Figure 4(b)); otherwise, the nondominated solution with the smallest constraint violation is selected as the second parent x_2, and then the infeasible solution with smallest constraint violation in the population excluding x_2 is selected as the third parent x_3 (Figure 4(c)). If there is more than one feasible solution, the top two feasible solutions with small objective value are selected as the parents x_1 and x_2. Moreover, if there is no solution which is nondominated with the best feasible solution, then the infeasible solution with the smallest constraint violation is selected as the third parent x_3 (Figure 4(d)); otherwise, the nondominated one with smallest constraint violation is the third parent x_3 (Figure 4(e)).

In the last stage, all solutions in the population are feasible, that is, $\rho = 1$. It is obvious that the top three solutions with small objective value are selected as parents x_1, x_2, and x_3.

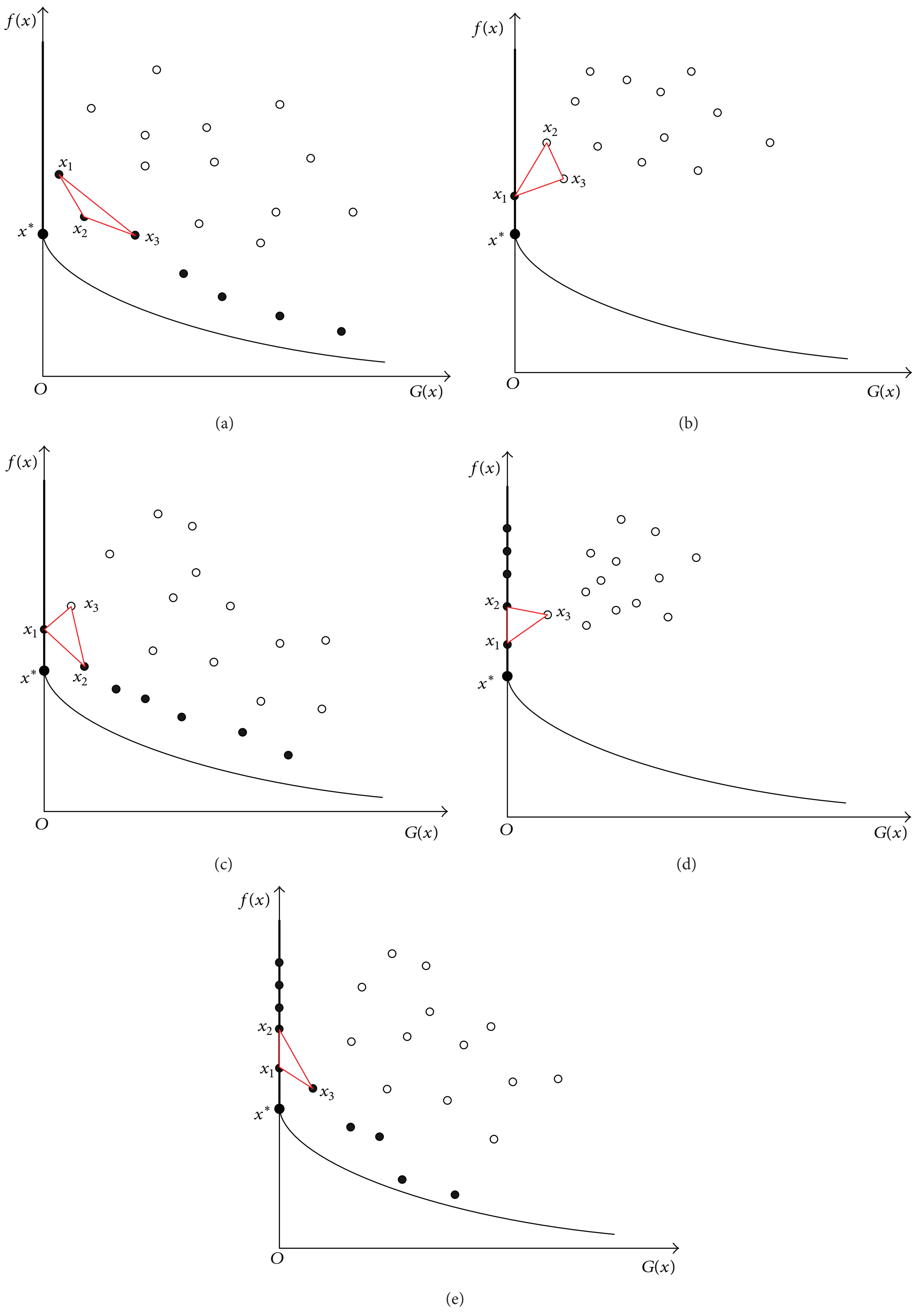

Figure 4: Solutions selected to perform SPX in different cases.

Table 1: Statistical results obtained by MDEDP over 30 independent runs and the comparison with Var2 [24], SAMSDE-2 [24], SR [2], ATMES [9], and BBO-DM [25]. The result in boldface indicates that the proposed MDEDP obtained the global optimum. The bold-face values mean they are optimal solutions. NA means not available in related reference.

fcn/opt.	Status	MDEDP	Var2 [24]	SAMSDE-2 [24]	SR [2]	ATMES [9]	BBO-DM [25]	DEDP
g01/−15.0000	Best	**−15.000**	**−15.0000**	**−15.0000**	**−15.000**	**−15.000**	**−15**	−14.9999
	Median	**−15.000**	**−15.0000**	**−15.0000**	**−15.000**	**−15.000**	NA	−14.9998
	Mean	**−15.000**	**−15.0000**	**−15.0000**	**−15.000**	**−15.000**	**−15**	−14.9999
	Worst	**−15.000**	**−15.0000**	**−15.0000**	**−15.000**	**−15.000**	**−15**	−14.9998
	Std.	0	0	0	0.0*E* + 00	1.6*E* − 14	8.2*E* − 14	3.9*E* − 05
g02/−0.803619	Best	−0.803616	−0.803616	**−0.803619**	−0.803515	−0.803388	−0.803557	−0.604182
	Median	−0.763716	−0.785259	−0.783053	−0.785800	−0.792420	NA	−0.554816
	Mean	−0.757713	−0.779414	−0.774917	−0.781975	−0.790148	−0.802774	−0.553601
	Worst	−0.616138	−0.729905	−0.729398	−0.726288	−0.756986	−0.792576	−0.511070
	Std.	1.9*E* − 02	1.9*E* − 02	2.5*E* − 02	2.0*E* − 02	1.3*E* − 02	2.7*E* − 03	2.0*E* − 02
g03/−1.0005	Best	**−1.0005**	**−1.0005**	**−1.0005**	−1.000	−1.000	−1.000	**−1.0005**
	Median	**−1.0005**	**−1.0005**	**−1.0005**	−1.000	−1.000	NA	**−1.0005**
	Mean	**−1.0005**	**−1.0005**	**−1.0005**	−1.000	−1.000	−1.000	**−1.0005**
	Worst	**−1.0005**	**−1.0005**	**−1.0005**	−1.000	−1.000	−1.000	**−1.0005**
	Std.	2.9*E* − 07	1.3*E* − 05	8.2*E* − 06	1.9*E* − 04	5.9*E* − 05	6.0*E* − 16	9.4*E* − 14
g04/ −30665.53867	Best	**−30665.53867**	**−30665.53867**	**−30665.53867**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.53867**
	Median	**−30665.53867**	**−30665.53867**	**−30665.53867**	**−30665.539**	**−30665.539**	NA	**−30665.53867**
	Mean	**−30665.53867**	−30662.69876	**−30665.53867**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.53867**
	Worst	**−30665.53867**	−30634.93399	**−30665.53867**	**−30665.539**	**−30665.539**	**−30665.539**	**−30665.53867**
	Std.	1.4*E* − 11	8.7*E* + 00	0	2.0*E* − 05	4.6*E* − 13	1.7*E* − 11	1.1*E* − 11
g05/5126.49671	Best	**5126.49671**	5126.49674	**5126.49671**	5126.497	5126.498	5126.498	**5126.49671**
	Median	**5126.49671**	5126.98662	5126.49674	5127.372	5126.776	NA	**5126.49671**
	Mean	**5126.49671**	5133.70682	5126.63543	5128.881	5127.648	5126.498	**5126.49671**
	Worst	**5126.49671**	5217.94796	5128.80623	5142.472	5135.256	5126.498	**5126.49671**
	Std.	3.0*E* − 12	2.0*E* + 01	5.0*E* − 01	3.5*E* + 00	1.8*E* + 00	2.2*E* − 04	9.3*E* − 13
g06/−6961.814	Best	**−6961.814**	**−6961.83188**	**−6961.83188**	**−6961.814**	**−6961.814**	**−6961.814**	**−6961.814**
	Median	**−6961.814**	−6961.80882	**−6961.83188**	**−6961.814**	**−6961.814**	NA	**−6961.814**
	Mean	**−6961.814**	−6937.41732	**−6961.83188**	−6875.940	**−6961.814**	**−6961.814**	**−6961.814**
	Worst	**−6961.814**	−6804.36991	**−6961.83188**	−6350.262	**−6961.814**	**−6961.814**	**−6961.814**
	Std.	1.8*E* − 12	4.8*E* + 01	0	1.6*E* + 02	4.6*E* − 12	4.6*E* − 12	1.9*E* − 02
g07/24.306	Best	**24.306**	24.3069	24.3069	24.307	**24.306**	24.326	24.330
	Median	**24.306**	24.3205	24.3287	24.357	24.313	NA	24.340
	Mean	**24.306**	24.3289	24.3330	24.374	24.316	24.345	24.342
	Worst	**24.306**	24.4085	24.3881	24.642	24.359	24.378	24.360
	Std.	4.6*E* − 06	2.5*E* − 02	2.5*E* − 02	6.6*E* − 02	1.1*E* − 02	1.3*E* − 02	8.7*E* − 03
g08/−0.095825	Best	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**
	Median	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	**−0.095825**	NA	**−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**
	Std.	8.4*E* − 17	0	0	2.6*E* − 17	2.8*E* − 17	2.8*E* − 17	2.8*E* − 17
g09/680.630	Best	**680.630**	**680.630**	**680.630**	**680.630**	**680.630**	**680.630**	**680.630**
	Median	**680.630**	680.633	680.631	680.641	680.633	NA	**680.630**
	Mean	**680.630**	680.634	680.631	680.656	680.639	**680.630**	**680.630**
	Worst	**680.630**	680.646	680.632	680.763	680.673	**680.630**	**680.630**
	Std.	3.6*E* − 10	4.0*E* − 03	6.8*E* − 04	3.4*E* − 02	1.0*E* − 02	4.3*E* − 13	1.5*E* − 11

Table 1: Continued.

fcn/opt.	Status	MDEDP	Var2 [24]	SAMSDE-2 [24]	SR [2]	ATMES [9]	BBO-DM [25]	DEDP
g10/7049.248	Best	7049.249	7049.570	7049.378	7054.316	7052.253	7059.802	7052.473
	Median	7049.257	7071.060	7105.992	7372.613	7215.357	NA	7054.639
	Mean	7049.258	7105.844	7112.012	7559.192	7250.437	7075.832	7054.623
	Worst	7049.311	7347.264	7191.295	8835.655	7560.224	7098.254	7058.081
	Std.	$1.1E-02$	$7.3E+01$	$7.0E+01$	$5.3E+02$	$1.2E+02$	$8.5E+00$	$1.5E+00$
g11/0.7499	Best	**0.7499**	**0.7499**	**0.7499**	**0.750**	**0.75**	**0.750**	**0.7499**
	Median	**0.7499**	**0.7499**	**0.7499**	**0.750**	**0.75**	NA	**0.7499**
	Mean	**0.7499**	**0.7499**	**0.7499**	**0.750**	**0.75**	**0.750**	**0.7499**
	Worst	**0.7499**	**0.7499**	**0.7499**	**0.750**	**0.75**	**0.750**	**0.7499**
	Std.	$3.4E-16$	0	0	$8.0E-05$	$3.4E-04$	0	$1.0E-16$
g12/−1.000	Best	**−1**	**−1.0000**	**−1.0000**	**−1.000000**	**−1.000**	**−1.000000**	**−1**
	Median	**−1**	**−1.0000**	**−1.0000**	**−1.000000**	**−1.000**	NA	**−1**
	Mean	**−1**	**−1.0000**	**−1.0000**	**−1.000000**	**−1.000**	**−1.000000**	**−1**
	Worst	**−1**	**−1.0000**	**−1.0000**	**−1.000000**	−0.994	**−1.000000**	**−1**
	Std.	0	0	0	$0.0E+00$	$1.0E-03$	0	0
g13/0.0539415	Best	**0.0539415**	0.0539452	**0.0539415**	0.053957	0.053950	NA	**0.0539415**
	Median	**0.0539415**	0.0546893	**0.0539415**	0.057006	0.053952	NA	0.0588605
	Mean	**0.0539415**	0.0602843	0.0539417	0.067543	0.053959	NA	0.1586557
	Worst	**0.0539415**	0.0828200	0.0539428	0.216915	0.053999	NA	0.4461180
	Std.	$1.3E-12$	$9.9E-03$	$3.0E-07$	$3.1E-02$	$1.3E-05$	NA	$1.6E-01$

In the above LS procedure, the best feasible and infeasible solutions are selected to form the simplex which crosses the feasible and infeasible region in the search space. Furthermore, the neighborhood that the LS engine works on is dynamically adjusted with the evolution and the search will focus on the promising region in search space, and the strategy encourages exploitation adaptively from both sides of the boundary of the feasible region.

4.3. Memetic DE Based on Dynamic Preference for COPs.

For biobjective problem (3) with preference, the proposed EA, denoted by MDEDP, is described as follows.

Algorithm 1. Consider the following.

Step 1 (initialization). Randomly generate the initial population $P(0)$ with size NP in the search space S, and set $k = 0$.

Step 2 (global search). Apply DE/rand/1/bin as the global search engine to evolve the population $P(k)$, and the offspring set is denoted by O_1.

Step 3 (local search). Perform the local search on the promising region according to the details in Section 3.2. The LS based on SPX generates μ offspring, which are randomly selected from the enlarged simplex. The offspring constitute set O_2.

Step 4 (selection). Select the next population from $P(k)\bigcup O_1\bigcup O_2$ according to the fitness function value $F(x)$ in Section 3.1.

Step 5 (stopping criterion). If stopping criterion is met (usually the maximum number of function evaluations is used), stop; else let $k = k + 1$, and go to Step 2.

5. Simulation Results

5.1. Test Functions and Parameters Setting.

In this section, we apply the proposed MDEDP to 22 standard test functions from [1, 2, 24]. The parameters used in simulations are given below. Population size, NP $= 200$, the scale factor F in DE mutation, and the probability of crossover CR are uniformly random number in $[0.8, 0.9]$ and $[0.9, 0.95]$, respectively. In the SPX based LS, the expanding ratio is $\varepsilon = 3$, and the number of offspring is $\mu = 10$. The maximum number of function evaluations (FEs) is set to 240000. In the constraint handling technique, the equality constraints are converted to inequality constrains by a tolerance value δ. In the experiments, the tolerance value δ is dynamically set as proposed in [29] and used in the [9, 12]. The tolerance value δ is

$$\delta(t+1) = \begin{cases} \dfrac{\delta(t)}{1.0168}, & \text{if } \delta > 10^{-4}, \\ 10^{-4}, & \text{else.} \end{cases} \tag{8}$$

The initial $\delta(0)$ is set to 3. The initial value and the proportional decreasing factor δ are the same as those in [9]. For each test function, 30 independent trials are executed, and the statistical results of 30 trials are recorded. The statistical results include the “best,” “median,” “mean,” and “worst” of the obtained optimal solutions and the standard deviations (Std.).

5.2. Experimental Results and Comparison. The experimental results are compared with the state-of-the-art algorithms: Var2 [24] and SAMSDE-2 [24], SR [2], ATMES [9], and CBBO-DM [25]. Among the comparison algorithms, Var2 [24] used DE/rand/3/bin as search engine and used feasibility rules [3] as comparison criteria, ASMSDE-2 combines adaptively two DE mutation schemes as search engine and also used feasibility rules [3] as comparison criteria, SR employed ES as the search engine and designed stochastic ranking to make comparison, ATMES used ES as the search engine and proposed an adaptive tradeoff fitness as selection operator, and CBBO-DM combines the BBO and DE mutation operators together. It is mentioned here that our algorithms MDEDP, ATMES, Var2 [24], and SAMSDE-2 [24] used 240000 FEs, while SR and CBBO-DM used 350000 FEs. The tolerance of equality constraints δ for Var2 [24] and SAMSDE-2 [24] is $1.0E-04$, while it was set to $1.0E-03$ for CBBO-DM, and for MDEDP and ATMES, both tolerance values are dynamically decreased. For MDEDP, it is decreased from initial value 3 to $1.0E-04$ eventually, while for ATMES, it is decreased from initial value of 3 to $5.0E-06$ in the end.

Table 1 gives the results of six algorithms, our MDEDP, Var2 [24], SAMSDE-2 [24], SR [2], ATMES [9], and CBBO-DM [25] for the first 13 test instances since SR, ATMES, and CBBO-DM solved only the first 13 problems. The results and comparisons with Var2 [24] and SAMSDE-2 [24] for the remaining nine test instances are given in Table 2.

In order to illustrate the significant difference, the approximate two-sample t-tests between the comparison algorithms and our MDEDP have been done according to [18]:

$$t_0 = \frac{\overline{y}_1 - \overline{y}_2}{\sqrt{S_1^2/n_1 + S_2^2/n_2}}, \tag{9}$$

where $\overline{y}_1$ and $\overline{y}_2$ denote the mean values, S_1 and S_2 are the standard deviations of the results obtained by the two algorithms, and n_1 and n_2 are the independent runs of two algorithms, respectively. The value of degrees of freedom is calculated as follows:

$$f = \left\lfloor \frac{1}{k^2/n_1 + (1-k)^2/n_2} \right\rfloor, \quad \text{where } k = \frac{S_1^2/n_1}{S_1^2/n_1 + S_2^2/n_2}. \tag{10}$$

For the first 13 test instances, the t-tests are done between our MDEDP and three algorithms, SR, ATMES, and CBBO-DM, on 6 instances (g02, g05, g07, g10, and g13). The t-test is not done on other instances since the optimal solutions are found by at least three algorithms in all runs. The data used for t-test is from the related references and the results of t-tests are calculated and showed in Table 3. In the table, "NA" means no available data in the related references, "–" means both approaches have obtained the optimal solutions in all runs on the given functions, and "NF" means no feasible solutions found in the algorithm. Superscript "a" means the difference between MDEDP and the compared algorithm with the corresponding degrees of freedom is significant at $\alpha = 0.05$ and MDEDP is better than the compared algorithm, while superscript "b" means that the MDEDP is worse than the compared algorithm.

As Table 3 shows, most of the differences are significant except the differences between ATMES and MDEDP on

Table 2: Statistical results obtained by MDEDP over 30 independent runs and the comparison with Var2 [24] and SAMSDE-2 [24] for the remaining 9 test instances. "NF" means no feasible solution is found.

fcn/opt.	Status	MDEDP	Var2 [24]	SAMSDE-2 [24]
g14/ −47.76489	Best	**−47.76489**	−47.76206	−46.66291
	Median	**−47.76489**	−47.52454	−46.63655
	Mean	−47.76487	−47.46548	−46.43386
	Worst	−47.76478	−46.53409	−46.00211
	Std.	$2.9E-05$	$3.0E-01$	$3.7E-01$
g15/ 961.715022	Best	**961.715022**	**961.715022**	**961.715022**
	Median	**961.715022**	**961.715022**	**961.715022**
	Mean	**961.715022**	961.716603	**961.715022**
	Worst	**961.715022**	961.757371	**961.715022**
	Std.	$6.9E-13$	$7.9E-03$	$4.1E-08$
g16/ −1.9051553	Best	**−1.9051553**	**−1.9051553**	**−1.9051553**
	Median	**−1.9051553**	**−1.9051553**	**−1.9051553**
	Mean	**−1.9051553**	**−1.9051553**	**−1.9051553**
	Worst	**−1.9051553**	**−1.9051553**	**−1.9051553**
	Std.	$6.6E-16$	$3.2E-10$	0
g17/ 8853.5339	Best	**8853.5339**	8853.5398	8853.5397
	Median	**8853.5339**	8949.0383	8927.5977
	Mean	8858.4711	8937.0179	8914.0841
	Worst	8927.5917	8956.7443	8941.2845
	Std.	$1.9E+01$	$2.7E+01$	$3.1E+01$
g18/ −0.8660254	Best	**−0.8660254**	**−0.866025**	**−0.866025**
	Median	**−0.8660254**	−0.866024	−0.866024
	Mean	−0.8405529	−0.866020	−0.866018
	Worst	−0.6749814	−0.865983	−0.865959
	Std.	$6.6E-02$	$9.8E-06$	$1.4E-05$
g19/ 32.65559	Best	**32.65559**	32.69440	32.67473
	Median	**32.65559**	32.85972	32.88680
	Mean	**32.65559**	32.93035	32.92857
	Worst	32.65567	33.46296	33.37313
	Std.	$1.6E-05$	$2.1E-01$	$2.0E-01$
g21/ 193.72451	Best	**193.72451**	NF	308.90899
	Median	193.72451	NF	324.71234
	Mean	220.35875	NF	324.10440
	Worst	330.18271	NF	329.54888
	Std.	$5.4E+01$	NF	$6.7E+01$
g23/ −400.05510	Best	**−400.05510**	−382.522771	−374.201398
	Median	**−400.05510**	−191.134573	−262.684823
	Mean	−400.05509	−199.903075	−265.241202
	Worst	−400.05505	−6.556980	−176.606485
	Std.	$1.3E-05$	$9.7E+01$	$5.3E+01$
g24/ −5.5080133	Best	**−5.5080133**	**−5.5080133**	**−5.5080133**
	Median	**−5.5080133**	**−5.5080133**	**−5.5080133**
	Mean	**−5.5080133**	**−5.5080133**	**−5.5080133**
	Worst	**−5.5080133**	**−5.5080133**	**−5.5080133**
	Std.	$1.8E-15$	0	0

Table 3: Results of the approximate two-sample t-test between the other three algorithms and our MDEDP for six instances.

Algorithms	t-test	g02	g05	g07	g09	g10	g13
SR-MDEDP	t_0	**−3.09**[b]	**3.73**[a]	**5.64**[a]	**4.19**[a]	**5.27**[a]	**2.40**[a]
	f	45	30	30	30	30	30
ATMES-MDEDP	t_0	**−4.42**[b]	**3.50**[a]	**4.98**[a]	**5.26**[a]	**9.18**[a]	2.04
	f	36	30	30	30	30	30
CBBO-MDEDP	t_0	**−5.01**[b]	**32.12**[a]	**16.43**[a]	—	**17.12**[a]	NA
	f	30	30	30	—	30	NA

Table 4: Results of the approximate two-sample t-test between the other two algorithms and our MDEDP for 12 instances.

Algorithms	t-test	g02	g05	g07	g09	g10	g13	g14	g17	g18	g19	g21	g23
Var2-MDEDP	t_0	**−2.80**[b]	1.97	**5.02**[a]	**5.48**[a]	**4.25**[a]	**3.51**[a]	**5.47**[a]	**13.03**[a]	**−2.11**[a]	**7.17**[a]	—	**11.30**[a]
	f	44	30	30	30	30	30	30	53	30	30	—	30
SAMSDE-MDEDP	t_0	**−2.07**[b]	1.52	**5.92**[a]	**8.05**[a]	**4.91**[a]	**3.65**[a]	**19.70**[a]	**8.38**[a]	**−2.11**[a]	**7.48**[a]	**6.60**[a]	**13.93**[a]
	f	51	30	30	30	30	30	30	49	30	30	57	30

problem g13, which illustrates that our MDEDP is competitive compared with other algorithms for the most widely used 13 benchmark problems.

For all the 22 test instances, the t-tests are done between our MDEDP and Var2, SAMSDE-2 on 12 instances (g02, g05, g07, g09, g10, g13, g14, g17, g18, g19, g21, and g23). The t-test is not done on other instances since the optimal solutions are found by at least two algorithms in all runs. The results of t-test are showed in Table 4, in which it is clear that most of the differences are significant except the differences between either one of the compared algorithms and MDEDP on problem g05. Therefore, it is concluded that our MDEDP is competitive with the other two state-of-the-art algorithms for the 22 test instances.

In summary, compared with other five algorithms for constrained optimization problems, MDEDP is very competitive.

In the proposed MDEDP, SPX based local search is incorporated to speed up the convergence of the algorithm. To test the efficiency of the SPX based local search, the comparisons are done on the most widely used 13 test instances between the proposed algorithm with local search and without local search (the algorithm without local search is denoted by DEDP). The parameters setting is the same as MDEDP with the exception of discarding the SPX based local search. It is mentioned here that the proposed algorithm without local search also performed 240000 FEs, which is the same as that of MDEDP. The results for DEDP are also showed in the last column of the Table 1. From Table 1, it is clear that DEDP can find nine optimal solutions (g03, g04, g05, g06, g08, g09, g11, g12, and g13) among 13 test instances, which shows that the used DE search engine is powerful and the proposed fitness function based on ASF can achieve a good balance between objective function and constraint violation. However, the DEDP is poor on the other four test instances which have difficulties of high dimension of variables, many local optimal solutions, and so forth. Compared with DEDP, MDEDP performs clearly better than that of the DEDP, which illustrates that the SPX based local search can obviously improve the efficiency of the DEDP.

5.3. *Effect of the Weighting Vector.*

In this subsection, the effect of the weighting vector which reflects the preference is discussed through various experiments on the most widely used 13 test instances.

In stage one, that is, $\rho = 0$, there is no feasible solution in the population, and the weighting vector ω is set to (0.1, 0.9). A big component of ω_2 illustrates that, in this stage, the search biases to the constraint satisfaction are much more. Meanwhile, ω_1 that is not set to zero means that the objective function should not be ignored absolutely.

In further experiments, the weighting vector is set to (0.2, 0.8), (0.3, 0.7), (0.4, 0.6), and (0.5, 0.5), and the results are shown in Table 5. The results are only for 10 out of the most widely used 13 test functions except for g02, g04, and g12 since the feasible regions for g02, g04, and g12 are relatively large which makes the populations seldom or not experience phase one and the value of ω almost does affect the evolution. For convenient comparison, the results with $\omega = (0.1, 0.9)$ are also shown in Table 5. From Table 5, it can be seen that, with the four different weighting vectors, the proposed algorithm can converge consistently to the global optimum in 30 independent runs for the 10 test functions which all experience phase one at least once. Therefore, the proposed algorithm is not sensitive to the preset parameter vector ω in stage one. The results and the above analysis demonstrated that MDEDP is robust.

In stage two, that is, $0 < \rho < 1$, there are feasible and infeasible solutions in the population at the same time. With the evolution, the feasible solution proportion increases accordingly. In this phase, parameter $\omega_1^{\max}$, the upper bound of the first component of ω, is used to limit the bias to the objective function in evolution. In the numerical experiments, $\omega_1^{\max} = 0.5$. The given upper bound means that the preference to the objective function is at most the same to the constraint satisfaction, which is the primary goal. Further experiments are performed on all test functions with different upper bound values, 0.4, 0.45, 0.6, and 0.7. The upper bound values of 0.4 and 0.45 represent that the preference to objective function is slightly less than that to the constraint satisfaction when there are more feasible solutions in the

Table 5: Statistical results obtained by MDEDP for 10 among 13 test functions over 30 independent runs with different weighting vectors in stage one.

fcn/opt.	Status	$\omega = (0.1, 0.9)$	$\omega = (0.2, 0.8)$	$\omega = (0.3, 0.7)$	$\omega = (0.4, 0.6)$	$\omega = (0.5, 0.5)$
	Best	−15	−15	−15	−15	−15
	Median	−15	−15	−15	−15	−15
g01/−15.0000	Mean	−15	−15	−15	−15	−15
	Worst	−15	−15	−15	−15	−15
	Std.	0	$2.1E-14$	$1.4E-14$	$1.2E-14$	$1.6E-14$
	Best	−1.0005	−1.0005	−1.0005	−1.0005	−1.0005
	Median	−1.0005	−1.0005	−1.0005	−1.0005	−1.0005
g03/−1.0005	Mean	−1.0005	−1.0005	−1.0005	−1.0005	−1.0005
	Worst	−1.0005	−1.0005	−1.0005	−1.0005	−1.0005
	Std.	$2.9E-07$	$2.4E-06$	$3.3E-06$	$4.0E-06$	$1.3E-06$
	Best	5126.49671	5126.49671	5126.49671	5126.49671	5126.49671
	Median	5126.49671	5126.49671	5126.49671	5126.49671	5126.49671
g05/5126.49671	Mean	5126.49671	5126.49671	5126.49671	5126.49671	5126.49671
	Worst	5126.49671	5126.49671	5126.49671	5126.49671	5126.49671
	Std.	$3.0E-12$	$4.8E-12$	$3.9E-12$	$4.0E-12$	$3.0E-12$
	Best	−6961.814	−6961.814	−6961.814	−6961.814	−6961.814
	Median	−6961.814	−6961.814	−6961.814	−6961.814	−6961.814
g06/−6961.814	Mean	−6961.814	−6961.814	−6961.814	−6961.814	−6961.814
	Worst	−6961.814	−6952.814	−6961.814	−6961.814	−6961.814
	Std.	$1.8E-12$	$1.8E-12$	$1.8E-12$	$1.8E-12$	$1.9E-12$
	Best	24.306	24.306	24.306	24.306	24.306
	Median	24.306	24.306	24.306	24.306	24.306
g07/24.306	Mean	24.306	24.306	24.306	24.306	24.306
	Worst	24.306	24.306	24.306	24.306	24.306
	Std.	$4.6E-06$	$3.5E-06$	$7.6E-06$	$3.2E-06$	$5.3E-06$
	Best	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825
	Median	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825
g08/−0.095825	Mean	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825
	Worst	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825
	Std.	$8.4E-17$	$8.4E-17$	$8.4E-17$	$8.4E-17$	$8.4E-17$
	Best	680.630	680.630	680.630	680.630	680.630
	Median	680.630	680.630	680.630	680.630	680.630
g09/680.630	Mean	680.630	680.630	680.630	680.630	680.630
	Worst	680.630	680.630	680.630	680.630	680.630
	Std.	$5.2E-10$	$1.1E-11$	$5.5E-08$	$3.2E-07$	$5.2E-10$
	Best	7049.249	7049.258	7049.252	7049.258	7049.249
	Median	7049.257	7049.357	7049.372	7049.344	7049.250
g10/7049.248	Mean	7049.258	7049.382	7049.405	7049.370	7049.258
	Worst	7049.311	7049.901	7049.773	7049.785	7049.311
	Std.	$1.1E-02$	$1.3E-01$	$1.3E-01$	$1.0E-01$	$2.8E-02$
	Best	0.7499	0.7499	0.7499	0.7499	0.7499
	Median	0.7499	0.7499	0.7499	0.7499	0.7499
g11/0.7499	Mean	0.7499	0.7499	0.7499	0.7499	0.7499
	Worst	0.7499	0.7499	0.7499	0.7499	0.7499
	Std.	$3.4E-16$	$3.4E-16$	$3.4E-16$	$3.4E-16$	$3.4E-16$
	Best	0.05395415	0.05395415	0.05395415	0.05395415	0.05395415
	Median	0.05395415	0.05395415	0.05395415	0.05395415	0.05395415
g13/0.0539415	Mean	0.05395415	0.05395415	0.05395415	0.05395415	0.05395415
	Worst	0.05395415	0.05395415	0.05395415	0.05395415	0.05395415
	Std.	$1.3E-12$	$6.1E-11$	$4.4E-12$	$3.8E-10$	$1.3E-12$

Table 6: Statistical results obtained by MDE-DP over 30 independent runs with different upper bound values of the first component in weighting vector ω in stage two.

fcn/opt.	Status	$\omega_1^{\max} = 0.4$	$\omega_1^{\max} = 0.45$	$\omega_1^{\max} = 0.5$	$\omega_1^{\max} = 0.6$	$\omega_1^{\max} = 0.7$
g01/−15.0000	Best	−15	−15	−15	−15	−15
	Median	−15	−15	−15	−15	−15
	Mean	−15	−15	−15	−15	−15
	Worst	−15	−15	−15	−15	−15
	Std.	$1.3E-14$	$2.4E-14$	0	$1.8E-14$	$1.8E-14$
g02/−0.803619	Best	−0.794882	−0.803617	−0.803616	−0.803615	−0.803613
	Median	−0.757723	−0.771748	−0.763716	−0.771840	−0.773241
	Mean	−0.750477	−0.758338	−0.757713	−0.761655	−0.764268
	Worst	−0.640313	−0.590475	−0.616138	−0.601025	−0.647718
	Std.	$4.0E-02$	$4.4E-02$	$1.9E-02$	$3.8E-02$	$3.5E-02$
g03/−1.0005	Best	−1.0005	−1.0005	−1.0005	−1.0005	−1.0005
	Median	−1.0005	−1.0005	−1.0005	−1.0005	−1.0005
	Mean	−1.0005	−1.0005	−1.0005	−1.0005	−1.0005
	Worst	−1.0005	−1.0005	−1.0005	−1.0005	−1.0005
	Std.	$2.9E-06$	$1.8E-06$	$2.9E-07$	$2.7E-06$	$2.9E-06$
g04/30665.539	Best	30665.539	30665.539	30665.539	30665.539	30665.539
	Median	30665.539	30665.539	30665.539	30665.539	30665.539
	Mean	30665.539	30665.539	30665.539	30665.539	30665.539
	Worst	30665.539	30665.539	30665.539	30665.539	30665.539
	Std.	$1.5E-11$	$1.5E-11$	$1.4E-11$	$1.5E-11$	$1.5E-11$
g05/5126.49671	Best	5126.49671	5126.49671	5126.49671	5126.49671	5126.49671
	Median	5126.49671	5126.49671	5126.49671	5126.49671	5126.49671
	Mean	5126.49671	5126.49671	5126.49671	5126.49671	5126.49671
	Worst	5126.49671	5126.49671	5126.49671	5126.49671	5126.49671
	Std.	$4.7E-12$	$4.0E-12$	$3.0E-12$	$4.0E-12$	$4.8E-12$
g06/−6961.814	Best	−6961.814	−6961.814	−6961.814	−6961.814	−6961.814
	Median	−6961.814	−6961.814	−6961.814	−6961.814	−6961.814
	Mean	−6961.814	−6961.814	−6961.814	−6961.814	−6961.814
	Worst	−6961.814	−6952.814	−6961.814	−6961.814	−6961.814
	Std.	$1.8E-12$	$1.8E-12$	$1.8E-12$	$1.8E-12$	$1.8E-12$
g07/24.306	Best	24.306	24.306	24.306	24.306	24.306
	Median	24.306	24.306	24.306	24.306	24.306
	Mean	24.306	24.306	24.306	24.306	24.306
	Worst	24.307	24.306	24.306	24.307	24.307
	Std.	$1.2E-04$	$8.5E-06$	$4.6E-06$	$6.1E-05$	$1.4E-04$
g08/−0.095825	Best	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825
	Median	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825
	Mean	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825
	Worst	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825
	Std.	$8.5E-17$	$8.4E-17$	$8.4E-17$	$8.4E-17$	$8.4E-17$
g09/680.630	Best	680.630	680.630	680.630	680.630	680.630
	Median	680.630	680.630	680.630	680.630	680.630
	Mean	680.630	680.630	680.630	680.630	680.630
	Worst	680.630	680.630	680.630	680.630	680.630
	Std.	$1.7E-12$	$1.2E-12$	$3.6E-10$	$6.2E-12$	$1.1E-08$
g10/7049.248	Best	7049.260	7049.270	7049.249	7049.259	7049.257
	Median	7049.350	7049.364	7049.257	7049.328	7049.356
	Mean	7049.397	7049.410	7049.258	7049.360	7049.370
	Worst	7049.864	7049.954	7049.311	7049.703	7049.696
	Std.	$1.3E-02$	$1.4E-01$	$1.1E-02$	$9.6E-02$	$7.4E-02$

Table 6: Continued.

fcn/opt.	Status	$\omega_1^{\max} = 0.4$	$\omega_1^{\max} = 0.45$	$\omega_1^{\max} = 0.5$	$\omega_1^{\max} = 0.6$	$\omega_1^{\max} = 0.7$
g11/0.7499	Best	0.7499	0.7499	0.7499	0.7499	0.7499
	Median	0.7499	0.7499	0.7499	0.7499	0.7499
	Mean	0.7499	0.7499	0.7499	0.7499	0.7499
	Worst	0.7499	0.7499	0.7499	0.7499	0.7499
	Std.	$3.4E-16$	$3.4E-16$	$3.4E-16$	$3.4E-16$	$3.4E-16$
g12/−1	Best	−1	−1	−1	−1	−1
	Median	−1	−1	−1	−1	−1
	Mean	−1	−1	−1	−1	−1
	Worst	−1	−1	−1	−1	−1
	Std.	0	0	0	0	0
g13/0.0539415	Best	0.05395415	0.05395415	0.05395415	0.05395415	0.05395415
	Median	0.05395415	0.05395415	0.05395415	0.05395415	0.05395415
	Mean	0.05395415	0.05395415	0.05395415	0.05395415	0.06163874
	Worst	0.05395415	0.05395415	0.05395415	0.05395415	0.43880261
	Std.	$2.7E-13$	$5.6E-10$	$1.3E-12$	$5.6E-10$	$5.4E-02$

population, while upper bound values of 0.6 and 0.7 represent that the preference to objective function is slightly more than that to the constraint satisfaction. The experimental results are shown in Table 6. It can be seen from the results in Table 6 that MDEDP can obtain similar results with different upper bound values of the preference to objective function. From the final results to g02, we can see that the algorithm is the most robust with the upper bound 0.7. This is because the whole search space is almost feasible, and the search biasing to the objective function is helpful to find optimal solution and coincides with a larger value of the first component of the weight vector. The obtained results indicated that the different upper bounds have little effect on the efficiency and robustness of the proposed algorithm.

From the above experimental results, we can conclude that the proposed MDEDP is robust to find the optimal solutions for the test functions with the different preset parameters.

6. Conclusion

The constrained optimization problem is converted into a biobjective optimization problem first and then solved by a new designed memetic differential evolution algorithm with dynamic preference. DE is used as the search engine in global search, which is guided by a novel fitness function based on ASF used in multiobjective optimization. The fitness function ASF dynamically adjusted the preference to different objectives through the adaptive reference point and weighting vector. In local search, SPX is used as the search mechanism and concentrates on the neighborhood the best feasible and infeasible solutions constituted, which makes the algorithm approach the promising region in objective space and accelerates the convergence. Experimental results and comparison with the state-of-the-art algorithms demonstrate that the proposed algorithm is efficient on these standard test problems. Further experiments with different preset parameters also show the robustness of the proposed algorithm.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work is supported by the National Natural Science Foundation of China (no. 61272119).

References

[1] A. W. Mohamed and H. Z. Sabry, "Constrained optimization based on modified differential evolution algorithm," *Information Sciences*, vol. 194, pp. 171–208, 2012.

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

[3] K. Deb, "An efficient constraint handling method for genetic algorithms," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, no. 2–4, pp. 311–338, 2000.

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

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

[6] D. Powell and M. M. Skolnick, "Using genetic algorithms in engineering design optimization with nonlinear constraints," in *Proceedings of the 5th International Conference on Genetic Algorithms (ICGA '93)*, pp. 424–431, San Mateo, Calif, USA, 1993.

[7] Y.-R. Zhou, Y.-X. Li, Y. Wang, and L.-S. Kang, "Pareto strength evolutionary algorithm for constrained optimization," *Journal of Software*, vol. 14, no. 7, pp. 1243–1249, 2003.

[8] Z. Cai and Y. Wang, "A multiobjective optimization-based evolutionary algorithm for constrained optimization," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 658–675, 2006.

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

[10] T. Takahama and S. Sakai, "Constrained optimization by the ε constrained differential evolution with gradient-based mutation and feasible elites," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '06)*, pp. 1–8, July 2006.

[11] T. P. Runarsson and X. Yao, "Search biases in constrained evolutionary optimization," *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 35, no. 2, pp. 233–243, 2005.

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

[13] C.-L. Sun, J.-C. Zeng, and J.-S. Pan, "An improved vector particle swarm optimization for constrained optimization problems," *Information Sciences*, vol. 181, no. 6, pp. 1153–1163, 2011.

[14] A. Ghosh, S. Das, A. Chowdhury, and R. Giri, "An improved differential evolution algorithm with fitness-based adaptation of the control parameters," *Information Sciences*, vol. 181, no. 18, pp. 3749–3765, 2011.

[15] W. Gong, Á. Fialho, Z. Cai, and H. Li, "Adaptive strategy selection in differential evolution for numerical optimization: An Empirical Study," *Information Sciences*, vol. 181, no. 24, pp. 5364–5386, 2011.

[16] M. Weber, F. Neri, and V. Tirronen, "A study on scale factor in distributed differential evolution," *Information Sciences*, vol. 181, no. 12, pp. 2488–2511, 2011.

[17] J. Lampinen, "A constraint handling approach for the differential evolution algorithm," in *Proceedings of the Congress on Evolutionary Computation (CEC '2002)*, pp. 1468–1473, IEEE Press, 2002.

[18] C. Zhang, X. Li, L. Gao, and Q. Wu, "An improved electromagnetism-like mechanism algorithm for constrained optimization," *Expert Systems with Applications*, vol. 40, no. 14, pp. 5621–5634, 2013.

[19] Y. Wang and Z. Cai, "Combining multiobjective optimization with differential evolution to solve constrained optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 1, pp. 117–134, 2012.

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

[21] E. Mezura-Montes, J. Velázquez-Reyes, and C. A. C. Coello, "Promising infeasibility and multiple offspring incorporated to differential evolution for constrained optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '05)*, pp. 225–232, June 2005.

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

[23] V. L. Huang, A. K. Qin, and P. N. Suganthan, "Self-adaptive differential evolution algorithm for constrained real-parameter optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '06)*, pp. 17–24, July 2006.

[24] S. M. Elsayed, R. A. Sarker, and D. L. Essam, "On an evolutionary approach for constrained optimization problem solving," *Applied Soft Computing Journal*, vol. 12, no. 10, pp. 3208–3227, 2012.

[25] I. Boussaïd, A. Chatterjee, P. Siarry, and M. Ahmed-Nacer, "Biogeography-based optimization for constrained optimization problems," *Computers & Operations Research*, vol. 39, no. 12, pp. 3293–3304, 2012.

[26] R. Storn and K. Price, "Differential evolution-a simple and efficient adaptative scheme for global optimization over continuous spaces," Tech. Rep. TR-95-12, International Computer Science, Berkeley, Calif, USA, 1995.

[27] S. Tsutsui, M. Yamamura, and T. Higuchi, "Multi-parent recombination with simplex crossover in real-coded genetic algorithms," in *Proceedings of the Genetic and Evolutionary Computing Conference*, pp. 657–664, Morgan Kaufmann, 1999.

[28] K. Miettinen, *Nonlinear Multiobjective Optimization*, International Series in Operations Research & Management Science, 12, Kluwer Academic Publishers, Boston, Mass, USA, 1999.

[29] S. Ben Hamida and M. Schoenauer, "ASCHEA: new results using adaptive segregational constraint handling," in *Proceedings of the Congress on Evolutionary Computation (CEC '2002)*, pp. 884–889, IEEE Press, 2002.