# A Multiobjective Differential Evolution Algorithm for Constrained Optimization

Wenyin Gong, and Zhihua Cai

***Abstract*— Recently, using multiobjective optimization concepts to solve the constrained optimization problems (COPs) has attracted much attention. In this paper, a novel multiobjective differential evolution algorithm, which combines several features of previous evolutionary algorithms (EAs) in a unique manner, is proposed to COPs. Our approach uses the orthogonal design method to generate the initial population; also the crossover operator based on the orthogonal design method is employed to enhance the local search ability. In order to handle the constraints, a novel constraint-handling method based on Pareto dominance concept is proposed. An archive is adopted to store the nondominated solutions and a relaxed form of Pareto dominance, called $\epsilon$-dominance, is used to update the archive. Moreover, to utilize the archive solution to guide the search, a hybrid selection mechanism is proposed. Experiments have been conducted on 13 benchmark COPs. And the results prove the efficiency of our approach. Compared with five state-of-the-art EAs, our approach provides very good results, which are highly competitive with those generated by the compared EAs in constrained evolutionary optimization. Furthermore, the computational cost of our approach is relatively low.**

## I. Introduction

Evolutionary Algorithms (EAs) are heuristics that have been successfully applied in a wide set of areas [1]. However, many of them are unconstrained search techniques and lack an explicit mechanism to bias the search in constrained search spaces. This has motivated the development of a considerable number of approaches to incorporate constraints into the fitness function of an EA [2] - [5].

Differential evolution (DE) [6] algorithm is a novel EA for faster optimization. Unlike Genetic Algorithm (GA) that uses binary coding to represent problem parameters, DE is a simple yet powerful population based, direct search algorithm using real valued parameters. Among the DE's advantages are its simple structure, ease of use, speed and robustness. Price & Storn [6] gave the working principle of DE with single scheme. Later on, they suggested ten different schemes of DE [7]. It has been successfully used in solving single-objective optimization problems. Hence, several researchers have tried to extend it to handle COPs [8] - [11]. Although many methods have been proposed to handle constraints by DEs, experimental results actually reveal that most of them do not have general capability in handling various COPs.

In this paper, we extend our previous work [12] and propose a novel multiobjective DE algorithm, called DE-MOC, to COPs. The differences between DE-MOC and $\epsilon$-ODEMO [12] are: i) DE-MOC is used to solve COPs, hence the constraint-handling method based on Pareto dominance concept is proposed in DE-MOC; ii) the orthogonal design method is employed to design the crossover operator to enhance the local search ability; and iii) in DE-MOC, the $\epsilon$-dominance method used to update the archive is different from that of $\epsilon$-ODEMO. Our new approach is characterized by a) It uses the orthogonal design method with quantization technique to generate the initial population; also the crossover operator based on the orthogonal design method is employed. b) A novel constraint-handling method based on Pareto dominance concept is proposed to handle the constraints. c) An archive is adopted to store the nondominated solutions and an $\epsilon$-dominance method proposed in [13] is used to update the archive. And d) a hybrid selection mechanism is proposed to use the archive solution to guide the search. With these combined elements, we show that our proposed approach is efficient. And the results of DE-MOC are highly competitive with those generated with an approach that represents the state-of-the-art in constrained evolutionary optimization.

The rest of this paper is organized as follows. Section II reviews the related work of handling COPs via EAs. A detailed description of our proposed DE-MOC is provided in Section III. In Section IV, we test our algorithm through 13 benchmark COPs. In addition, the experiment results are compared with those of some state-of-the-art EAs. Finally, Section V is devoted to conclusions.

## II. Related Work

A global constrained minimization problem can be formalized as a pair $(S, f)$ , where $S \subseteq R^n$ is a bounded set on $R^n$ and $f : S \rightarrow R$ is an $n$-dimensional real-valued function. The problem is to find a point $\boldsymbol{x}_{min} \in S$ such that $f(\boldsymbol{x}_{min})$ is a global minimum on $S$. More specifically, it is required to find an $\boldsymbol{x}_{min} \in S$ such that

$$\forall \boldsymbol{x} \in S : f(\boldsymbol{x}_{min}) \leq f(\boldsymbol{x}) \tag{1}$$

subject to

$$g_i(\boldsymbol{x}) \leq 0, i = 1, 2, \cdots, q \tag{2}$$

$$h_j(\boldsymbol{x}) = 0, j = q + 1, \cdots, m \tag{3}$$

where $\boldsymbol{x}$ is the vector of solutions $\boldsymbol{x} = [x_1, x_2, \cdots, x_n]^T$, $q$ is the number of inequality constraints, and $m-q$ is the number of equality constraints (in both cases, constraints could be linear or nonlinear). Generally, for each variable $x_i$ it satisfies a constrained boundary

$$l_i \leq x_i \leq u_i, i = 1, 2, \cdots, n \tag{4}$$

The authors are all with School of Computer Science, China University of Geosciences, Wuhan, China.
E-mail: cug11100304@yahoo.com.cn; zhcai@cug.edu.cn.



978-1-4244-1823-7/08/$25.00©2008 IEEE

The distance of an individual $\boldsymbol{x}$ from the $j$th constraint can be constructed as

$$G_j(\boldsymbol{x}) = \begin{cases} \max\{0, g_j(\boldsymbol{x})\}, & 1 \leq j \leq q \\ |h_j(\boldsymbol{x})|, & q+1 \leq j \leq m \end{cases} \tag{5}$$

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

Coello [2] provided a comprehensive survey of the most popular constraint-handling techniques currently used with EAs and grouped them into five categories. 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.

Since our approach belongs to the group of techniques in which multiobjective optimization concepts are adopted to handle constraints, we will briefly discuss some of the most relevant work done in this area. In [14], Cai and Wang classified 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.

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 $f(\boldsymbol{x})$, and the second is to minimize the degree of constraint violation $G(\boldsymbol{x})$. Examples of this type of approach are given in [14] and [18].

And 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. Example of this type of approach is given in [15].

In a technical report [16], four of the existing techniques based on multiobjective optimization concepts have been compared. 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 [16] is that additional mechanisms have to be used to improve the effectiveness of these approaches.

## III. Proposed Approach

Inspired by the ideas from the orthogonal design method successfully used in EAs ([17], [12], and [18]) and $\epsilon$-dominance proposed in [13], in this work, we extend our previous work [12] to COPs. Our proposed DE algorithm is named DE-MOC. In DE-MOC, it recasts COPs as bi-objective optimization problems to minimize the original objective function $f(\boldsymbol{x})$ and the degree of constraint violation $G(\boldsymbol{x})$ simultaneously. For the sake of clarity, let $F(\boldsymbol{x}) = (f(\boldsymbol{x}), G(\boldsymbol{x}))$. Without loss of generality, minimization problems are assumed in this paper. Six crucial procedures of DE-MOC will be discussed as follows.

### A. Constraint-handling Method

Since we will be using some multiobjective optimization concepts, it is appropriate to introduce two essential definitions in the context of our approach.

*Definition 1 (*Pareto Dominance*):* A solution space vector $\boldsymbol{x} = (x_1, \cdots, x_k)$ is said to Pareto dominate another solution vector $\boldsymbol{y} = (y_1, \cdots, y_k)$, denoted as $\boldsymbol{x} \prec \boldsymbol{y}$, if and only if

$$\forall i \in 1, \cdots, k, x_i \leq y_i \quad \text{and} \quad \exists i \in 1, \cdots, k, x_i < y_i$$

*Definition 2 (*$\epsilon$-dominance*):* Let $\boldsymbol{f}, \boldsymbol{g} \in \Re^k$. Then $\boldsymbol{f}$ is said to $\epsilon$-dominance $\boldsymbol{g}$ for some $\epsilon > 0$, denoted as $\boldsymbol{f} \prec_\epsilon \boldsymbol{g}$, if and only if for all $i \in \{1, \cdots, k\}$, $f_i - \epsilon \leq g_i$.

In this work, we propose a new constraint-handling method, which does not need any parameters to be tuned for constraint handling. This method is based on the "Constraint-First-Objective-Next" model [19], where the constraints precede the objectives because the feasibility of $\boldsymbol{x}$ is more important than minimization of $\boldsymbol{f}(\boldsymbol{x})$. The method to check the constrained dominance between solution $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is described in Algorithm 1. And to check the constrained $\epsilon$-dominance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is described in Algorithm 2. The dominance in constraint space ($\prec_c$) is defined as [20]:

*Definition 3 (Constraint space dominance):* A solution $\boldsymbol{x}_i$ is said to dominate a solution $\boldsymbol{x}_j$ in constraint space, denoted as $\boldsymbol{x}_i \prec_c \boldsymbol{x}_j$, if both conditions are true:

1. Solutions $\boldsymbol{x}_i$ is no worse than solution $\boldsymbol{x}_j$ in all constraints, *i.e.*,
$$\forall G_k(\boldsymbol{x}_i) \leq G_k(\boldsymbol{x}_j) \tag{6}$$
2. Solution $\boldsymbol{x}_i$ is strictly better than solution $\boldsymbol{x}_j$ in at least one constraint, *i.e.*,
$$\exists G_k(\boldsymbol{x}_i) < G_k(\boldsymbol{x}_j) \tag{7}$$
where $G_k(\boldsymbol{x}) = \max(0, g_k(\boldsymbol{x})), k = 1, \cdots, q$ and $G_k(\boldsymbol{x}) = |h_k(\boldsymbol{x})|, k = q+1, \cdots, m$.

**Algorithm 1** Constrained dominance, *dominance*($\boldsymbol{x}_i$, $\boldsymbol{x}_j$)

```
if x_i ≺_c x_j then
    return 1
else if x_j ≺_c x_i then
    return −1
else
    if x_i ≺ x_j then
        return 1
    else if x_j ≺ x_i then
        return −1
    else
        return 0
    end if
end if
```

### B. Orthogonal Initial Population

Before solving an optimization problem, we usually have no information about the location of the global minimum. It is desirable that an algorithm starts to explore those points that are scattered evenly in the feasible solution space. In our presented manner, the algorithm can evenly scan the feasible solution space once to locate good points for further exploration in subsequent iterations. In this work, we use the orthogonal design method to generate the initial archive and initial evolutionary population (EP). Due to the tight space

**Algorithm 2** Constrained $\epsilon$-dominance, *$\epsilon$-dominance*($\boldsymbol{x}_i$, $\boldsymbol{x}_j$)

**if** $\boldsymbol{x}_i \prec_c \boldsymbol{x}_j$ **then**
    return 1
**else if** $\boldsymbol{x}_j \prec_c \boldsymbol{x}_i$ **then**
    return $-1$
**else**
    **if** $\boldsymbol{x}_i \prec_\epsilon \boldsymbol{x}_j$ **then**
        return 1
    **else if** $\boldsymbol{x}_j \prec_\epsilon \boldsymbol{x}_i$ **then**
        return $-1$
    **else**
        return 0
    **end if**
**end if**

restrictions however, we omit the detailed method to generate the orthogonal initial population. More details can be found in [17].

### C. Orthogonal Crossover Operator

In order to enhance the local search ability and accelerate the convergence speed of our approach, we employ the orthogonal design method with quantization to design the crossover operator called *orthogonal crossover*. It acts on two parents. It quantizes the solution space defined by these parents into a finite number of points, and then applies orthogonal design to select a small, but representative sample of points as the potential offspring. The algorithm of the orthogonal crossover can be found in [17] and [18].

The main difference between our approach and [18] when using the orthogonal crossover is that in DE-MOC only the best individual is selected to replace one individual randomly chosen from the EP. However, in [18], the orthogonal crossover is the main genetic operator; and all of the nondominanted individuals generated by the orthogonal crossover are selected to generate the next population.

### D. Archiving the Candidate Solutions

In the study of Zitzler *et al.* [21], it was clearly shown that elitism helps in achieving better convergence in MOEAs. In DE-MOC, the elitism scheme is also adopted through maintaining an external archive of nondominated solutions found in evolutionary process. In order to update the archive, an $\epsilon$-dominance method proposed in [13], is adopted. The algorithm is described in Algorithm 3.

**Algorithm 3** *update* function for $\epsilon$-approximate Pareto Set

**Input:** $A, f$
**if** $\exists f' \in A$ such that *$\epsilon$-dominance*($f'$, $f$)=1 **then**
    $A' := A$
**else**
    $D := \{f' \in A | \epsilon\text{-}dominance(f', f) = -1\}$
    $A' := A \cup \{f\} \backslash D$
**end if**
**Output:** $A'$

### E. Hybrid Selection Mechanism

In our proposed approach, in order to use the archive solution to guide the search we propose a hybrid selection mechanism, in which a random selection and an elitist selection are interleaved. In elitist selection of DE-MOC, we only randomly choose one solution from the archive as the base parent in DE/rand/1/bin. And the other two parents are selected from EP randomly. We use a *selection parameter* $\lambda \in [0.1, 1.0]$ to regulate the selection pressure.

$$\text{selection} = \begin{cases} \text{random selection}, & eval < (\lambda \times Max\_eval) \\ \text{elitist selection}, & \text{otherwise} \end{cases} \tag{8}$$

where, $eval$ is the current number of fitness function evaluations (NFFEs), and $Max\_eval$ is the maximal NFFEs predefined by the user.

### F. Handling the Constraint of the Variables

After using the DE/rand/1/bin scheme to generate a new solution, if one or more of the variables in the new solution are outside their boundaries, i.e. $x_i \notin [l_i, u_i]$, the following repair rule is applied:

$$x_i = \begin{cases} l_i + \text{rnd}_i[0,1] \times (u_i - l_i) & \text{if } x_i < l_i \\ u_i - \text{rnd}_i[0,1] \times (u_i - l_i) & \text{if } x_i > u_i \end{cases} \tag{9}$$

where $\text{rnd}_i[0,1]$ is the uniform random variable from [0,1] in each dimension $i$.

### G. Main Procedure of DE-MOC

For a COP, the proposed DE-MOC works as Algorithm 4. First, we use the orthogonal design method to generate the orthogonal initial population. At each generation, an offspring is generated using DE/rand/1/bin scheme. The offspring replaces the parent *immediately* if the parent is dominated by the offspring. If the parent dominates the offspring, the offspring is discarded. Otherwise (when the offspring and parent are nondominated), the offspring is added to a temporary child population (CP). Insert the offspring into the archive with constrained $\epsilon$-dominance concept described in Algorithm 3. This step is repeated until $NP$ number of offspring are created. After that, combine the temporary child population CP with EP, and we get a population size between $NP$ and $2\times NP$. If the population has enlarged, we have to truncate it to prepare it for the next step of the algorithm.

In contrast to MOEAs, since nondominated individuals represent the most important feature of the population they belong to, our concern in this work is only the nondominated individuals. The truncation only selects the nondominated individuals from the mixed population. After the truncation, we employ the orthogonal crossover to generate a good individual; and it is used to replace a randomly selected individual from the EP.

Finally, when some halting criterion are reached, the loop is terminated.

## IV. Experimental Study

In order to test the performance of our approach thirteen benchmark functions were used. All test functions are taken from [5]. Here we do not give the expressions to them. Their detail expressions are provided in the reference [5]. Functions g02, g03, g08, and g12 are **maximization** problems, and the remaining nine functions (g01, g04, g05, g06, g07, g09, g10, g11 and g13) are **minimization** problems.

**Algorithm 4** Main procedure of the proposed DE-MOC

Generate the orthogonal initial population
$t := 1$
**while** $eval < Max\_eval$ **do**
  $child\_size := 0$
  **for** $i := 1$ to $NP$ **do**
    Hybrid selection
    Produce the offspring $c$ with DE/rand/1/bin scheme
    Evaluate the offspring $c$ and $eval$++
    **if** *dominance*($c$, $EP_t^i$)=1 **then**
      $EP_t^i := c$
    **else if** *dominance*($c$, $EP_t^i$)=0 **then**
      Add $c$ to the child population $CP$
      $child\_size$++
    **else**
      Discard $c$
    **end if**
    Update the archive with constrained $\epsilon$-dominance concept
  **end for**
  **if** $child\_size \neq 0$ **then**
    Combine $CP$ and $EP_t$
    Find the nondominated individuals in the mixed population
    Get the next evolutionary population $EP_{t+1}$
  **end if**
  Use the orthogonal crossover to generate a good individual $best\_indv$
  Replace a randomly selected individual from EP with $best\_indv$
  $t$++
**end while**

### A. Experimental Setup

For all experiments, we used the following parameters:

- Population size: *NP* = 100;
- Maximal NFFEs: *Max_eval* = 100,000;
- Degree of violation[1]: $\delta = 0.0001$;
- Probability of crossover: *CR* = 0.99;
- Scaling factor of DE: $F = \mathrm{rnd}(0.00001, 0.99999)$;
- Selection parameter: $\lambda = 0.9$;
- Parameters of orthogonal initial population: the same as [12];
- Parameters of orthogonal crossover: the same as [18];
- $\epsilon$ vector: $\epsilon^T = [0.0, 1e-10]$.

### B. General Performance of the Proposed Approach

We perform 50 independent runs for each test function in standard C++. The results obtained with the DE-MOC are presented in Table I. This table shows the known "optimal" solutions for each function, and records the best, mean, and worst of the objective function values, and the standard deviations ("Std. Dev"), found over 50 runs. Where "*anffe*" denotes the average number of fitness function evaluations, and "*ps*" is the percentage of successful run when the final result equals to the optimum in 50 runs.

From Table I we can see that, for each test function, the best solution is almost equivalent to the optimal solution. For functions g01, g02, g04, g06, g07, g08, g09, g10, g12, and g13, the global optimal solutions are consistently found for all 50 runs. For functions g03, g05, and g11, the best results are even better than the optimal solutions of these functions. This is the consequence of using inequalities to approximate each equality, although we use a very small $\delta$.

Furthermore, it is apparent that the standard deviations over 50 runs for all the functions except for g13 are very small. These results confirm that our approach has a substantial capability of handling various COPs and its solution quality is quite robust and stable.

[1] In this work, all equality constraints have been converted into inequality constraints, $|h(\boldsymbol{x})| - \delta \leq 0$

### C. Comparison with State-of-the-art EAs

DEMO is compared with five state-of-the-art approaches: Stochastic Ranking (SR) [5], Improved SR (ISR) [3], Simple Multimembered Evolution Strategy (SMES) [4], Inverted-Shrinkable PAES (IS-PAES) [15], and Orthogonal Design based Constrained Optimization Evolutionary Algorithm (ODCOEA) [18]. The best results obtained by each approach are shown in Table II. The mean values provided are compared in Table III and the worst results are presented in Table IV. The results provided by these approaches were taken from the original references for each method.

As shown in Table II – IV, we can conclude that:

- With respect to SR, one of the most competitive constraint-handling approaches used with EAs, DE-MOC was able to obtain better "best" solutions in seven functions (g02, g03, g05, g07, g10, g11 and g13) and similar "best" solutions in the remaining six functions (g01, g04, g06, g08, g09 and g12). Additionally, DE-MOC got better "mean" and "worst" solutions in eight functions (g02, g03, g05, g06, g07, g09, g10 and g11) and similar "mean" and "worst" solutions in four functions (g01, g04, g08 and g12). However, DE-MOC found worse "mean" and "worst" solutions in function g13.
- With respect to ISR, an improved version of SR, DE-MOC found better "best" solutions for functions g05, and g11 and similar "best" solutions in the nine functions (g01, g02, g04, g06, g07, g08, g09, g10 and g12). ISR provided a better "best" solution for functions g03 and g13. DE-MOC got better "mean" and "worst" solutions in five functions (g02, g05, g07, g10 and g11) and similar "mean" and "worst" solutions in six functions (g01, g04, g06, g08, g09 and g12). ISR found better "mean" solutions for functions g03 and g13.
- With respect to SMES, DE-MOC obtained better "best" results in eight functions (g02, g03, g05, g07, g09, g10, g11 and g13) and similar "best" solutions in five functions (g01, g04, g06, g08 and g12). In addition, our approach got better "mean" and "worst" solutions in eight functions (g02, g03, g05, g06, g07, g09, g10, and g11) and similar "mean" and "worst" solutions in the remaining four functions (g01, g04, g08 and g12). However, SMES was able to provide a better "mean" result for function g13.
- With respect to IS-PAES, which uses the multiobjective optimization concept to handle the constraints, DE-MOC provided better "best" solutions in six functions (g02, g03, g07, g10, g11, and g13) and similar "best" results in five functions (g01, g04, g06, g08, and g09). The proposed method found better "mean" results in nine functions (g01, g02, g03, g06, g07, g09, g10, g11, and g13) and similar "mean" results in two (g04,

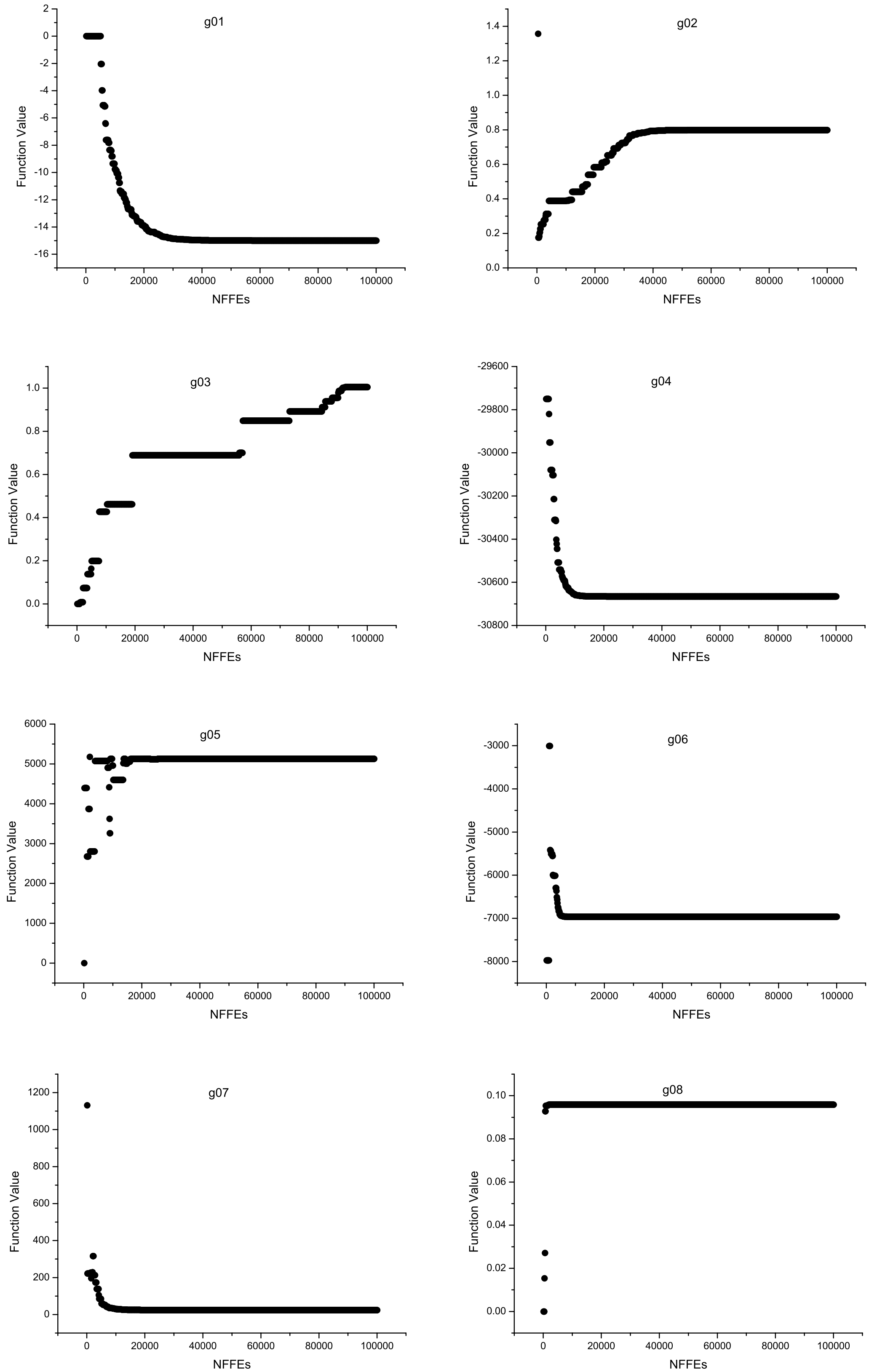


Fig. 1. Convergence results for test problems g01 - g08

TABLE I

STATISTICAL RESULTS OBTAINED BY DE-MOC FOR THE 13 TEST FUNCTIONS OVER 50 INDEPENDENT RUNS. A RESULT IN **BOLDFACE** INDICATES THAT THE GLOBAL OPTIMUM (OR BEST KNOWN SOLUTION) WAS REACHED. WHERE "OPTIMAL" IN COLUMN 2 INDICATES THE GLOBAL OPTIMUM (OR BEST KNOWN SOLUTION), SIMILARLY HEREINAFTER.

| F | Optimal | Best | Mean | Worst | Std. Dev | $anffe$ | $ps$ |
|---|---|---|---|---|---|---|---|
| Min g01 | -15 | **-15** | -15 | -15 | 1.977E-15 | 45,161 | 100% |
| Max g02 | 0.803619 | **0.803619** | 0.79703 | 0.76493 | 0.00803 | 100,000 | 68.0% |
| Max g03 | 1 | **1.0005** | 1.0005 | 1.0005 | 2.325E-10 | 91,245 | 100% |
| Min g04 | -30665.539 | **-30665.539** | -30665.539 | -30665.539 | 0 | 48,299 | 100% |
| Min g05 | 5126.498 | **5126.49671** | 5126.49671 | 5126.49671 | 0 | 34,674 | 100% |
| Min g06 | -6961.814 | **-6961.814** | -6961.814 | -6961.814 | 5.820E-12 | 14,618 | 100% |
| Min g07 | 24.30621 | **24.30621** | 24.30621 | 24.30621 | 2.160E-6 | 68,355 | 100% |
| Max g08 | 0.095825 | **0.095825** | 0.095825 | 0.095825 | 0 | 2,955 | 100% |
| Min g09 | 680.63 | **680.63** | 680.63 | 680.63 | 0 | 18,760 | 100% |
| Min g10 | 7049.248 | **7049.248** | 7049.248 | 7049.248 | 1.849E-5 | 90,989 | 100% |
| Min g11 | 0.75 | **0.7499** | 0.7499 | 0.7499 | 0 | 15,542 | 100% |
| Max g12 | 1 | **1** | 1 | 1 | 0 | 1,974 | 100% |
| Min g13 | 0.05395 | **0.05395** | 0.19755 | 0.438494 | 0.08637 | 86,176 | 54.0% |

TABLE II

COMPARISON OF THE BEST SOLUTIONS OBTAINED BY OUR DE-MOC AGAINST THE SR, ISR, SMES, IS-PAES AND ODCOEA. A RESULT IN **BOLDFACE** INDICATES A BETTER RESULT WAS FOUND. NA = NOT AVAILABLE, SIMILARLY HEREINAFTER.

| F | Optimal | SR [5] | ISR [3] | SMES [4] | IS-PAES [15] | ODCOEA [18] | DE-MOC |
|---|---|---|---|---|---|---|---|
| Min g01 | -15 | **-15** | **-15** | **-15** | **-15** | **-15** | **-15** |
| Max g02 | 0.803619 | 0.803515 | **0.803619** | 0.803601 | 0.803376 | 0.0802996 | **0.803619** |
| Max g03 | 1 | 1 | **1.001** | 1 | 1 | 1 | 1.0005 |
| Min g04 | -30665.539 | **-30665.539** | **-30665.539** | **-30665.539** | **-30665.539** | -30665.536 | **-30665.539** |
| Min g05 | 5126.498 | 5126.497 | 5126.4981 | 5126.599 | NA | 5126.498 | **5126.49671** |
| Min g06 | -6961.814 | **-6961.814** | **-6961.814** | **-6961.814** | **-6961.814** | -6961.699 | **-6961.814** |
| Min g07 | 24.30621 | 24.307 | **24.30621** | 24.327 | 24.311 | 24.585 | **24.30621** |
| Max g08 | 0.095825 | **0.095825** | **0.095825** | **0.095825** | **0.095825** | **0.095825** | **0.095825** |
| Min g09 | 680.63 | **680.63** | **680.63** | 680.632 | **680.63** | 680.636 | **680.63** |
| Min g10 | 7049.248 | 7054.316 | **7049.248** | 7051.903 | 7062.019 | 7054.547 | **7049.248** |
| Min g11 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | **0.7499** |
| Max g12 | 1 | **1** | **1** | **1** | NA | **1** | **1** |
| Min g13 | 0.05395 | 0.053957 | **0.053942** | 0.053986 | 0.05517 | NA | 0.05395 |

TABLE III

COMPARISON OF THE MEAN SOLUTIONS OBTAINED BY OUR DE-MOC AGAINST THE SR, ISR, SMES, IS-PAES AND ODCOEA.

| F | Optimal | SR [5] | ISR [3] | SMES [4] | IS-PAES [15] | ODCOEA [18] | DE-MOC |
|---|---|---|---|---|---|---|---|
| Min g01 | -15 | **-15** | **-15** | **-15** | -14.494 | -14.999 | **-15** |
| Max g02 | 0.803619 | 0.781975 | 0.772078 | 0.785238 | 0.793281 | 0.792587 | **0.79703** |
| Max g03 | 1 | 1 | **1.001** | 1 | 1 | 1 | 1.0005 |
| Min g04 | -30665.539 | **-30665.539** | **-30665.539** | **-30665.539** | **-30665.539** | -30665.428 | **-30665.539** |
| Min g05 | 5126.498 | 5128.881 | 5126.4981 | 5174.492 | NA | 5158.922 | **5126.49671** |
| Min g06 | -6961.814 | -6875.940 | **-6961.814** | -6961.284 | 6961.813 | -6960.669 | **-6961.814** |
| Min g07 | 24.30621 | 24.374 | 24.30621 | 24.475 | 24.338 | 25.886 | **24.30621** |
| Max g08 | 0.095825 | **0.095825** | **0.095825** | **0.095825** | **0.095825** | **0.095825** | **0.095825** |
| Min g09 | 680.63 | 680.656 | **680.63** | 680.643 | 680.631 | 680.774 | **680.63** |
| Min g10 | 7049.248 | 7559.192 | 7049.249 | 7253.047 | 7342.944 | 7552.225 | **7049.248** |
| Min g11 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | **0.7499** |
| Max g12 | 1 | **1** | **1** | **1** | NA | **1** | **1** |
| Min g13 | 0.05395 | **0.067543** | 0.096276 | 0.166385 | 0.28184 | NA | 0.19755 |

and g08). It also found better "worst" results in eight functions, except for functions g04 and g08 in which the "worst" results were similar to that obtained with IS-PAES. A better "worst" result was obtained by IS-PAES in g02. However, the analysis was incomplete because IS-PAES did not test in functions g05 and g12.

- Compared with ODCOEA, which also adopts multiobjective optimization concept to handle the constraints and employs orthogonal crossover to generate the offspring, our approach got better "best" results in nine functions (g02, g03, g04, g05, g06, g07, g09, g10, and g11) and a similar "best" solution in three functions (g01, g08, g09, and g12). DE-MOC found better "mean" and "worst" solutions in ten functions (g01, g02, g03, g04, g05, g06, g07, g09, g10, and g11) and similar better "mean" and "worst" solutions in the remaining two (g08 and g12). Function g13 was not tested by ODCOEA. It is worth to point out that in ODCOEA the tolerance degree of $\delta = 0.001$. However, in DE-MOC $\delta = 0.0001$, which makes the functions more difficult to solve.

Regarding computational cost, we can say that the NFFEs

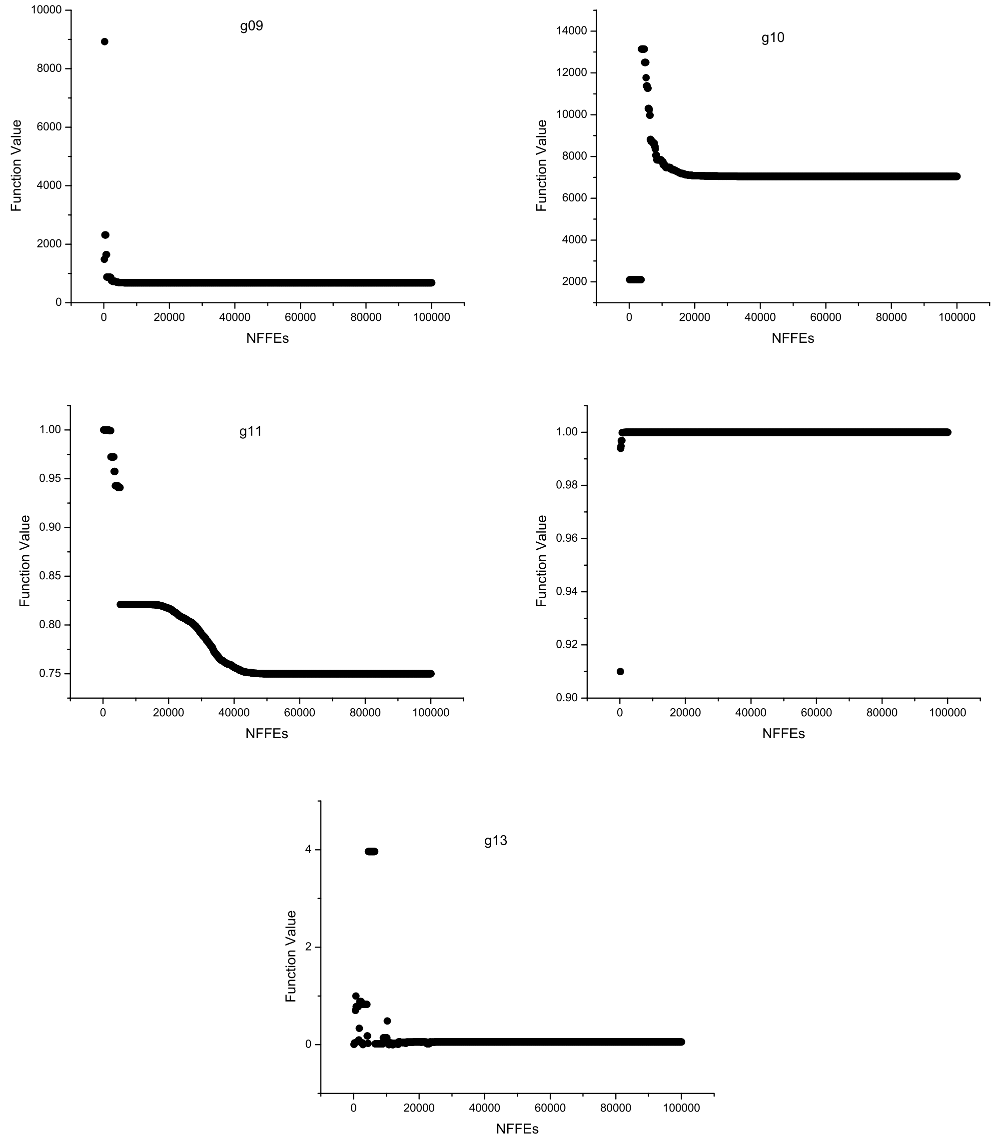


Fig. 2. Convergence results for test problems g09 - g13

TABLE IV
COMPARISON OF THE WORST SOLUTIONS OBTAINED BY OUR DE-MOC AGAINST THE SR, ISR, SMES, IS-PAES AND ODCOEA.

| F | Optimal | SR [5] | ISR [3] | SMES [4] | IS-PAES [15] | ODCOEA [18] | DE-MOC |
|---|---|---|---|---|---|---|---|
| Min g01 | -15 | **-15** | **-15** | **-15** | -12.446 | -14.999 | **-15** |
| Max g02 | 0.803619 | 0.726288 | 0.683055 | 0.751322 | **0.768291** | 0.762508 | 0.76493 |
| Max g03 | 1 | 1 | **1.001** | 1 | 1 | 1 | 1.0005 |
| Min g04 | -30665.539 | **-30665.539** | **-30665.539** | **-30665.539** | **-30665.539** | -30664.909 | **-30665.539** |
| Min g05 | 5126.498 | 5142.472 | 5126.4981 | 5304.167 | NA | 5249.496 | **5126.49671** |
| Min g06 | -6961.814 | 6350.262 | **-6961.814** | -6952.482 | 6961.810 | -6959.281 | **-6961.814** |
| Min g07 | 24.30621 | 24.642 | 24.308 | 24.843 | 24.995 | 27.262 | **24.30621** |
| Max g08 | 0.095825 | **0.095825** | **0.095825** | **0.095825** | **0.095825** | **0.095825** | **0.095825** |
| Min g09 | 680.63 | 680.763 | **680.63** | 680.719 | 680.634 | 680.998 | **680.63** |
| Min g10 | 7049.248 | 8835.655 | 7049.296 | 7638.366 | 7588.054 | 8482.351 | **7049.248** |
| Min g11 | 0.75 | 0.75 | 0.75 | 0.75 | 0.751 | 0.75 | **0.7499** |
| Max g12 | 1 | **1** | **1** | **1** | NA | **1** | **1** |
| Min g13 | 0.05395 | **0.216915** | 0.438803 | 0.468294 | 0.5471 | NA | 0.438494 |

performed by our approach is lower than the five techniques compared. Our approach performed only 100,000 NFFEs. SR, ISR, and IS-PAES performed 350,000 NFFEs, SMES performed 240,000 NFFEs, and ODCOEA required 150,000 NFFEs. Furthermore, as described in Table I, we can see that the average NFFEs were very small in eight functions (g01, g04, g05, g06, g08, g09, g11, and g12).

### *D. Convergence of DE-MOC*

In order to discuss the quality and robustness of the proposed approach, it is necessary to verify the rate at which the algorithm is able to achieve optimal or near-optimal solutions, i.e. the convergence dynamics of the approach. The convergence result for each problem is shown in figures 1 and 2 so that the efficiency of DE-MOC can be demonstrated more explicitly.

From Fig 1 and Fig 2, it can be seen that DE-MOC converges very fast.For ten out of thirteen problems it converges to a stable value after about 20,000 NFFEs except for g02, g03, and g11.

In summary, we can conclude that DE-MOC is slightly superior to SR, SMES, and IS-PAES. And it is substantially superior to ODCOEA in terms of the quality of the resulting solutions. Both DE-MOC and ISR are the competitive approaches for COPs. In general, DE-MOC can be considered as a good trade-off between efficiency and effectiveness for constrained evolutionary optimization.

## V. CONCLUSIONS

This paper presents an efficient multiobjective DE algorithm to COPs. Our approach integrates established techniques in existing EA's in a single unique algorithm. It is characterized by a) it uses the orthogonal design method with quantization technique to generate the initial population; also the crossover operator based on the orthogonal design method is employed; b) a novel constraint-handling method based on Pareto dominance concept is proposed to handle the constraints; c) an archive is adopted to store the nondominated solutions and the $\epsilon$-dominance is used to update the archive; and d) the hybrid selection mechanism is proposed to use the archive solution to guide the search.

Our approach is tested on 13 benchmark functions and the results indicate that this approach can be used to solve a range of COPs with linear/nonlinear equality/inequality constraints, as well as continuous/discontinuous search spaces. Compared with five state-of-the-art EAs, our approach provide a highly competitive performance. Moreover, this approach is easy to implement and its computational cost is relatively low.

## ACKNOWLEDGMENT

The authors would like to thank the three anonymous reviewers for their constructive advice. This work is support by the Humanities Base Project of Hubei Province under grant no. 2004B0011 and the Natural Science Foundation of Hubei Province under grant no. 2003ABA043.

## REFERENCES

[1] T. Bäck and H.P. Schwefel: An overview of evolutionary algorithms for parameter optimization, Evol. Comput., 1, 1-23, 1993.

[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., 191, 1245-1287, 2002.

[3] T.P. Runarsson and X. Yao, Search biases in constrained evolutionary optimization, IEEE Trans. Syst. Man. Cybern. C, 35(2), 233-243, 2005.

[4] E. Mezura-Montes and C.A.C. Coello, A simple multimembered evolution strategy to solve constrained optimization problems, IEEE Trans. Evol. Comput., 9(1), 1-17, 2005.

[5] T.P. Runarsson and X. Yao, Stochastic ranking for constrained evolutionary optimization, IEEE Trans. Evol. Comput., 4(3), 284-294, 2000.

[6] R. Storn and K. Price, Differential evolution–A simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization, 11, 341-359, 1997.

[7] R. Storn and K. Price, Home Page of Differential Evolution. Available: http://www.ICSI.Berkeley.edu/˜storn/code.html, 2003.

[8] M.F. Tasgetiren, and P.N. Suganthan, A Multi-Populated Differential Evolution Algorithm for Solving Constrained Optimization Problem, 2006 IEEE Cong. on Evol. Comput. (CEC-06), 33-40, 2006.

[9] S. Kukkonen and J. Lampinen, Constrained Real-Parameter Optimization with Generalized Differential Evolution, 2006 IEEE Cong. on Evol. Comput. (CEC-06), 207-214, 2006.

[10] E. Mezura-Montes, J. Velázquez-Reyes and C.A.C. Coello, Modified Differential Evolution for Constrained Optimization, 2006 IEEE Cong. on Evol. Comput. (CEC-06), 25-32, 2006.

[11] T. Takahama and S. Sakai, Constrained Optimization by the $\epsilon$ Constrained Differential Evolution with Gradient-Based Mutation and Feasible Elite, 2006 IEEE Cong. on Evol. Comput. (CEC-06), 1-8, 2006.

[12] Z.H. Cai, W.Y. Gong, Y.Q. Huang, A Novel Differential Evolution Algorithm based on $\epsilon$-domination and Orthogonal Design Method for Multiobjective Optimization, Fourth Int. Conf. on Evol. Multi-Criterion Optim. (EMO-07), LNCS 4403, 286-301, 2007.

[13] M. Laumanns, L. Thiele, K. Deb, E. Zitzler, Combining Convergence and Diversity in Evolutionary Multi-objective Optimization, Evol. Comput., 10(3), 263-282, 2002.

[14] Z.X. Cai and Y. Wang, A multiobjective optimization-based evolutionary algorithm for constrained optimization, IEEE Trans. Evol. Comput., 10(6), 658-675, 2006

[15] A.H. Aguirre, S.B. Rionda, C.A.C. Coello, et al, Handling constraints using multiobjective optimization concepts, Int. J. Numerical Methods in Eng., 59(15), 1989-2017, 2004.

[16] E. Mezura-Montes and C.A.C. Coello, A numerical comparison of some multiobjective-based techniques to handle constraints in genetic algorithms. Technical Report EVOCINV-03-2002, Evolutionary Computation Group at CINVESTAV-IPN, México, D.F. 07300, 2002.

[17] Y.W. Leung, and Y. Wang, An Orthogonal Genetic Algorithm with Quantization for Global Numerical Optimization. IEEE Trans. Evol. Comput., 5(1), 41-53, 2001.

[18] Y. Wang, H. Liu, Z.X. Cai, et al, An orthogonal design based constrained evolutionary optimization algorithm, Eng. Optim., 39(6), 715-736, 2007.

[19] A. Kurpati, S. Azarm, J. Wu,Constraint Handling Improvements for Multiobjective Genetic Algorithms, Struct. Multidisc. Optim., 23, 204-213, 2002.

[20] A. Oyama, K. Shimoyama, K. Fujii, New Constraint-handling Method for Multi-objective and Multi-Monstraint Evolutionary Optimization, Transactions of the Japan Society for Aeronautical and Space Sciences, 50, 56-62, 2007.

[21] E. Zitzler, K. Deb, L. Thiele, Comparison of Multiobjective Evolutionary Algorithms: Empirical Results, Evol. Comput., 8, 173-195, 2000.