Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 349781, 17 pages
http://dx.doi.org/10.1155/2015/349781

Research Article

A Multiobjective Genetic Algorithm Based on a Discrete Selection Procedure

Qiang Long,[1] Changzhi Wu,[2] Xiangyu Wang,[2,3] Lin Jiang,[4] and Jueyou Li[5]

[1]*School of Science, Southwest University of Science and Technology, Mianyang 621010, China*
[2]*Australasian Joint Research Centre for Building Information Modelling School of Built Environment, Curtin University, Perth, WA 6845, Australia*
[3]*Department of Housing and Interior Design, Kyung Hee University, Seoul 136701, Republic of Korea*
[4]*School of Mathematics, Anhui Normal University, Wuhu 430000, China*
[5]*School of Mathematics, Chongqing Normal University, Chongqing 404100, China*

Correspondence should be addressed to Changzhi Wu; changzhiwu@yahoo.com

Received 12 November 2014; Revised 15 January 2015; Accepted 20 January 2015

Academic Editor: Jianxiong Ye

Multiobjective genetic algorithm (MOGA) is a direct search method for multiobjective optimization problems. It is based on the process of the genetic algorithm; the population-based property of the genetic algorithm is well applied in MOGAs. Comparing with the traditional multiobjective algorithm whose aim is to find a single Pareto solution, the MOGA intends to identify numbers of Pareto solutions. During the process of solving multiobjective optimization problems using genetic algorithm, one needs to consider the elitism and diversity of solutions. But, normally, there are some trade-offs between the elitism and diversity. For some multiobjective problems, elitism and diversity are conflicting with each other. Therefore, solutions obtained by applying MOGAs have to be balanced with respect to elitism and diversity. In this paper, we propose metrics to numerically measure the elitism and diversity of solutions, and the optimum order method is applied to identify these solutions with better elitism and diversity metrics. We test the proposed method by some well-known benchmarks and compare its numerical performance with other MOGAs; the result shows that the proposed method is efficient and robust.

1. Introduction

In this paper, we consider the following multiobjective optimization problem:

$$\begin{aligned} &(\text{MOP}) \quad \text{Minimize} \quad \mathbf{F}(x) \\ &\qquad\qquad \text{Subject to} \quad x \in X, \end{aligned} \tag{1}$$

where $\mathbf{F}(x) = (f_1(x), f_2(x), \ldots, f_p(x))^T$ is a multiobjective function (vector valued function), $X = \{x \in \mathbb{R}^n : l_b \le x \le u_b\} \subset \mathbb{R}^n$ is a box set, and l_b and u_b are lower bound and upper bound, respectively. We assume that each function in $\mathbf{F}(x)$ is Lipschitz continuous but not necessarily differentiable.

The multiobjective optimization has extensive applications in engineering and management [1–3]. Most of the optimization problems appearing in the real-world application have multiple objectives; they can be modeled as multiobjective optimization problems. However, due to the theoretical and computational challenges, it is not easy to numerically solve multiobjective optimization problems. Therefore, the multiobjective optimization attracted lots of researches over the last decades [4–8].

So far, there are two types of methods to solve multiobjective optimization problems: the indirect method and direct method. The indirect method converts multiple objectives into a single one. One strategy is to combine the multiple objective functions using the utility theory [9] or the weighted sum method [10, 11]. The difficulty for such method is the selection of the utility function or proper weights so as to satisfy the decision-maker's preference. Another indirect method is to formulate the multiple objectives, except one, as constraints. However, it is not easy to determine the upper bounds of these objectives. On the one hand, small upper bounds could exclude some Pareto solutions; on the other hand, large upper bounds could enlarge the objective function value space which leads to some sub-Pareto solutions. Additionally, the indirect method can only obtain a single Pareto solution in each run. However, in practical applications, decision-makers often prefer a number of Pareto solutions so that they can choose one strategy according to their preferences.

Direct methods are devoted to exploring the entire set of Pareto solutions or a representative subset. However, it is extremely hard or impossible to obtain the entire set of Pareto solutions for most multiobjective optimization problems, except for some simple cases. Therefore, stepping back to a representative subset is preferred. Genetic algorithm, as a population-based algorithm, is a good choice to achieve this goal. The generic single-objective genetic algorithm can be modified to find a set of nondominated solutions in a single run [12–14]. The ability of the genetic algorithm to simultaneously search different regions of a solution space makes it possible to find a diverse set of solutions for difficult problems. The crossover and mutation operators of the genetic algorithm can be applied to various domains defined by different objectives, which in return creates new nondominated solutions in unexplored parts of the Pareto front. In addition, multiobjective genetic algorithm does not require the user to prioritize, scale, or weight objectives. Therefore, the genetic algorithm is one of the most popular metaheuristic approaches for solving multiobjective optimization problems [15–17].

The first multiobjective optimization method based on the genetic algorithm, called the vector evaluated genetic algorithm (VEGA), was proposed by Schaffer [18]. Afterwards, several multiobjective evolutionary algorithms were developed, such as multiobjective genetic algorithm (MOGA) [19], niched Pareto genetic algorithm (NPGA) [20], weight-based genetic algorithm (WBGA) [21], random weighted genetic algorithm (RWGA) [22], nondominated sorting genetic algorithm (NSGA) [23], strength Pareto evolutionary algorithm (SPEA) [24], improved SPEA (SPEA2) [25], Pareto-archived evolution strategy (PAES) [26], Pareto envelope-based selection algorithm (PESA) [27].

There are two basic criteria to measure a set of solutions for the multiobjective optimization problem [28].

(1) *Elitism*. The obtained solutions should be as close to the real Pareto solutions as possible. This can be measured by the closeness between the real Pareto frontier and the image of the obtained solutions since the image set of the real Pareto solutions is the real Pareto frontier.

(2) *Diversity*. In order to extensively describe Pareto solutions, the obtained solutions should distribute uniformly over the set of real Pareto solutions. The diversity of the obtained solutions is measured by the diversity of their images.

These two criteria of Pareto solutions are often in conflict with each other. Therefore, one has to balance the trade-off between elitism and diversity. The aim of this paper is to introduce new techniques to tackle these issues. The rest of the paper is organized as follows. In Section 2, we review some basic definitions of multiobjective optimization and the process of genetic algorithm. In Section 3, we propose an improved genetic algorithm for solving multiobjective optimization problems. In Section 4, some numerical experiments are carried out and the results are analyzed. Section 5 concludes the paper.

2. Preliminaries

In this section, we first review some definitions and theorems in the multiobjective optimization and then introduce the general procedure of genetic algorithm.

2.1. Definitions in Multiobjective Optimization. First of all, we present the following notations which are often used in vector optimization. Given two vectors

$$x = (x_1, x_2, \dots, x_n)^T, \qquad y = (y_1, y_2, \dots, y_n)^T \in \mathbb{R}^n, \quad (2)$$

then

(i) $x = y \Leftrightarrow x_i = y_i$ for all $i = 1, 2, \dots, n$;

(ii) $x < y \Leftrightarrow x_i < y_i$ for all $i = 1, 2, \dots, n$;

(iii) $x \leq y \Leftrightarrow x_i \leq y_i$ for all $i = 1, 2, \dots, n$, and there is at least one $i \in \{1, 2, \dots, n\}$ such that $x_i < y_i$; that is, $x \neq y$;

(iv) $x \leqq y \Leftrightarrow x_i \leq y_i$ for all $i = 1, 2, \dots, n$.

"$>$," "$\geq$," and "$\geqq$" can be defined similarly. In this paper, we call $x \leq y$ by x dominates y or y is dominated by x (in some literatures, $x \geq y$ is called x dominates y or y is dominated by y; we reverse this definition since we are solving minimization problem in this paper).

Definition 1. Suppose that $x \subseteq \mathbb{R}^n$ and $x^* \in X$. If $x^* \leqq x$ for any $x \in X$, then x^* is called an absolute optimal point of X.

Absolute optimal point is an ideal point, but it may not exist.

Definition 2. Let $x \in \mathbb{R}^n$ and $x^* \in X$. If there is no $x \in X$ such that

$$x \leq x^* \quad (\text{or } x < x^*), \quad (3)$$

then x^* is called an *efficient point* (or *weakly efficient point*).

The sets of absolute optimal points, efficient points, and weakly efficient points of X are denoted as X_{ab}, X_{ep}, and X_{wp}, respectively. For the problem MOP, $X \subseteq \mathbb{R}^n$ is called the *decision variable space* and its image set $\mathbf{F}(X) = \{y \in \mathbb{R}^p \mid y = \mathbf{F}(x),\ x \in X\} \subset \mathbb{R}^p$ is called the *objective function value space*.

Definition 3. Suppose that $x^* \in X$. If

$$\mathbf{F}(x^*) \leqq \mathbf{F}(x), \quad (4)$$

for any $x \in X$, x^* is called an *absolute optimal solution* of the problem MOP. The set of absolute optimal solutions is denoted as S_{as}.

The concept of the absolute optimal solution is a direct extension of that for single-objective optimization. It is the ideal solution but may not exist for most cases.

Definition 4. Suppose that $x^* \in X$. If there is no $x \in X$ such that

$$\mathbf{F}(x) \leq \mathbf{F}(x^*) \quad (\text{or } \mathbf{F}(x) < \mathbf{F}(x^*)); \tag{5}$$

that is, $\mathbf{F}(x^*)$ is an efficient point (or weakly efficient point) of the objective function value space $\mathbf{F}(X)$, then x^* is called an *efficient solution* (or *weakly efficient solution*) of the problem MOP. The sets of efficient solutions and weakly efficient solutions are denoted as S_{es} and S_{ws}, respectively.

Another name of the efficient solution is *Pareto solution*, which was introduced by Koopmans and Reiter in 1951 [29]. The meaning of the Pareto solution is that if $x^* \in S_{\text{es}}$, then there is no feasible solution $x \in X$, such that any $f_i(x)$ of $\mathbf{F}(x)$ is not worse than that of $\mathbf{F}(x^*)$. In other words, x^* is the best solution in the sense of "$\leq$." Another intuitive interpretation of Pareto solution is that it cannot be improved with respect to any objective without worsening at least one of the other objectives. The set of Pareto solutions is denoted by $\mathscr{P}^*$. Its image set $\mathbf{F}(\mathscr{P}^*)$ is called the *Pareto frontier*, denoted by $\mathscr{P}\mathscr{F}^*$. The following two theorems are well known.

Theorem 5 (see [6]). *For the multiobjective optimization, it holds that*

$$S_{as} \subset S_{es}(\mathscr{P}^*) \subset S_{ws} \subset X. \tag{6}$$

Theorem 6 (see [6]). *For the objective function value space* $\mathbf{F}(X)$*, if the sets of efficient points and weakly efficient points (i.e.,* F_{es} *and* F_{ws}*, resp.) are known, then, in the feasible set X, it holds that*

$$\begin{aligned} S_{es} &= \bigcup_{\overline{\mathbf{F}} \in F_{es}} \{x \in X \mid \mathbf{F}(x) = \overline{\mathbf{F}}\}, \\ S_{ws} &= \bigcup_{\overline{\mathbf{F}} \in F_{ws}} \{x \in X \mid \mathbf{F}(x) = \overline{\mathbf{F}}\}. \end{aligned} \tag{7}$$

Theorem 5 illustrates the relationship of the sets of absolute optimal solutions, efficient solutions, and weakly efficient solutions. Theorem 6 reveals that the preimage of efficient points (or weakly efficient points) in $F(X)$ is efficient solutions (or weakly efficient solutions) of the problem MOP.

2.2. Genetic Algorithm. Genetic algorithm is one of the most important evolutionary algorithms. It was introduced by Holland in the 1960s and then developed by his students and colleagues at the University of Michigan between the 1960s and 1970s [30]. Over the last two decades, the genetic algorithm was increasingly enriched by plenty of literatures, such as [31–34]. Nowadays various genetic algorithms are applied in different areas, for example, mathematical programming, combinational optimization, automatic control, and image processing.

Suppose that $P(t)$ and $O(t)$ represent parents and offspring at the tth generation, respectively. Then, the general structure of genetic algorithm can be described in the following pseudocode.

General Structure of Genetic Algorithm

```
(1) Initialization
    (1.1) Generate the initial population $P(0)$.
    (1.2) Set crossover rate, mutation rate, and maximal
          generation time.
    (1.3) Let $t \leftarrow 0$.
(2) Since the maximal generation time is not reached, do
    the following.
    (2.1) Crossover operator: generate $O_1(t)$.
    (2.2) Mutation operator: generate $O_2(t)$.
    (2.3) Evaluate $O_1(t)$ and $O_2(t)$: compute the fitness
          function.
    (2.4) Select operator: build the next population.
    (2.5) $t \leftarrow t + 1$, go to (2.1).
          End
    End.
```

From the pseudocode, we can see that there are three important operators in a general genetic algorithm: the crossover, mutation, and selection operators. The implementation of these operators is highly dependent on the way of encoding.

3. A New Multiobjective Genetic Algorithm

In this section, we present a new multiobjective genetic algorithm for solving the problem MOP. We first propose a ranking strategy called the optimum order method and then metrics for the elitism and diversity of solutions. Finally, a new selection operator for genetic algorithm is designed using the optimum order method and the elitism and diversity metrics.

Theoretically, the terminology "solution" means the point in the decision variable space, while the corresponding point in the objective function value space is named as "image of solution." However, most of the following discussions are in the objective function value space. In order to simplify the description, we indiscriminately call the point from the decision variable space and its corresponding image from the objective function value space as the "solution," if there is no confusion to do so.

3.1. The Optimum Order Method. In numerical optimization, in order to compare the numerical performance of different solutions, it is necessary to assign a fitness value for each solution. For the single-valued function, fitness is normally assigned as its function value. However, to assign the fitness of a multiobjective function is not straightforward. So far, there are three typical approaches. The first one is weighted sum approach [22, 35], which converts the multiple objectives into a single one using normalized weight $\sum_{i=1}^{p} \lambda_i = 1$,

$\lambda_i > 0, i = 1, 2, \ldots, p$. The decision of weight parameters is not an easy task for this approach. The second one is to alter the objective functions [18, 25], which randomly divides the current population into p equal subpopulations: $P_1, P_2, \ldots, P_p$. Then, each solution in the subpopulation P_i is assigned a fitness value based on the objective function f_i. In fact, this approach is a straightforward extension of the single-objective genetic algorithm. The last one is Pareto-ranking approach [4, 23, 36], which is a direct application of the definition of Pareto solution. In the following, we present a rank strategy called the optimum order method [6].

Definition 7. Let $P = \{1, 2, \ldots, p\}$ and $N = \{1, 2, \ldots, n\}$; for any $x^i, x^j \in X$, define

$$a_{ijl} = \begin{cases} 1, & \text{if } f_l(x^i) < f_l(x^j); \\ 0.5, & \text{if } f_l(x^i) = f_l(x^j); \\ 0, & \text{if } f_l(x^i) > f_l(x^j) \text{ or } i = j. \end{cases} \tag{8}$$

Then,

$$a_{ij} = \sum_{l \in P} a_{ijl} \quad i, j \in N \tag{9}$$

is called the *optimal number* of x^i corresponding to x^j for all objectives. Furthermore, $K_i = \sum_{j \in N} a_{ij}$ is defined as *the total optimal number* of x^i corresponding to all the other solutions for all objectives.

Obviously, for a minimization problem, a larger optimal number corresponds to a better solution. Therefore, optimal numbers can be considered as criteria for ranking a set of solutions. Due to this observation, we propose the following algorithm to rank a population of solutions.

Algorithm 8 (optimum order method (OOM)). Consider the following steps.

Step 1 (input). It includes the population of solutions and their objective function values.

Step 2 (compute optimal numbers). Compute the optimal number and total optimal number of each solution; fill these numbers into Table 1 according to (8).

Step 3 (rank the solution). Rearrange the order of solutions according to the decreasing order of the total optimal numbers K_i. More precisely, denote

$$\begin{aligned} K_{\alpha_1} &= \max_{i \in N} \{K_i\}, \\ K_{\alpha_2} &= \max_{i \in N \setminus \{\alpha_1\}} \{K_i\}, \end{aligned} \tag{10}$$

and so on.

The solutions x^{α_1} and x^{α_2}, which are corresponding to $K_{\alpha_1}, K_{\alpha_2}$, and so forth, are called *the best solution*, *the second best solution*, and so forth. The new order is called the *optimal order*. It is worth mentioning that the optimum order method does not necessarily rank Pareto solutions in the frontier positions; however, those solutions who are more reasonable with respect to all objectives are more likely to be ranked in the foremost positions. This property is different from Pareto-ranking approach.

Table 1: Table of optimal numbers.

x^i	x^j						K_i
	x^1	x^2	$\cdots$	x^j	$\cdots$	x^n	
x^1	0	a_{12}	$\cdots$	a_{1j}	$\cdots$	a_{1n}	K_1
x^2	a_{21}	0	$\cdots$	a_{2j}	$\cdots$	a_{2n}	K_2
$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$	$\vdots$
x^i	a_{i1}	a_{i2}	$\cdots$	a_{ij}	$\cdots$	a_{in}	K_i
$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$	$\vdots$
x^n	a_{n1}	a_{n2}	$\cdots$	a_{nj}	$\cdots$	0	K_n

Lemma 9. *Suppose that $x^i, x^j \in X$. If $\mathbf{F}(x^i) \le \mathbf{F}(x^j)$, then $a_{ij} > a_{ji}$.*

Proof. Let

$$\begin{aligned} A &= \{l \in P \mid f_l(x^i) < f_l(x^j)\}, \quad a = |A|; \\ B &= \{l \in P \mid f_l(x^i) = f_l(x^j)\}, \quad b = |B|, \end{aligned} \tag{11}$$

where $|\cdot|$ denotes the number of components in a set. Obviously, we have $a > 0, b > 0$ and $a + b = p$. Then,

$$\begin{aligned} a_{ijl} &= \begin{cases} 1, & l \in A, \\ 0.5, & l \in B, \end{cases} \\ a_{jil} &= \begin{cases} 0, & l \in A, \\ 0.5, & l \in B. \end{cases} \end{aligned} \tag{12}$$

Therefore,

$$\begin{aligned} a_{ij} &= \sum_{l \in A} a_{ijl} + \sum_{l \in B} a_{ijl} = a + 0.5b, \\ a_{ji} &= \sum_{l \in A} a_{jil} + \sum_{l \in B} a_{jil} = 0.5b. \end{aligned} \tag{13}$$

Since $a > 0$, $a_{ij} > a_{ji}$. □

Lemma 10. *Suppose that $x^i, x^j \in X$. If $\mathbf{F}(x^i) \le \mathbf{F}(x^j)$, then, for any x^k $(k \ne i, j,\ k \in N)$, $a_{ik} > a_{jk}$.*

Proof. For any x^k $(k \ne i, j,\ k \in N)$, let

$$\begin{aligned} A &= \{l \in P \mid f_l(x^k) < f_l(x^i)\}, \quad a = |A|; \\ B &= \{l \in P \mid f_l(x^k) > f_l(x^i)\}, \quad b = |B|; \\ C &= \{l \in P \mid f_l(x^k) = f_l(x^i)\}, \quad c = |C|. \end{aligned} \tag{14}$$

Then, we have $a, b, c \ge 0$ and $a + b + c = p$.

(i) When $l \in A$. Since $\mathbf{F}(x^i) \le \mathbf{F}(x^j)$, we have

$$f_l(x^k) < f_l(x^i), \quad f_l(x^k) < f_l(x^j). \tag{15}$$

Therefore,

$$a_{ik}^1 = \sum_{l\in A} a_{ikl} = 0 \cdot a = 0, \quad a_{jk}^1 = \sum_{l\in A} a_{jkl} = 0 \cdot a = 0. \tag{16}$$

(ii) When $l \in B$. We have $f_l(x^k) > f_l(x^i)$. Thus,

$$a_{ik}^2 = \sum_{l\in B} a_{ikl} = 1 \cdot b = b. \tag{17}$$

On the other hand, let

$$\begin{aligned} M_1 &= \{l \in B \mid f_l(x^k) < f_l(x^j)\}, \quad m_1 = |M_1|; \\ M_2 &= \{l \in B \mid f_l(x^k) = f_l(x^j)\}, \quad m_2 = |M_2|; \\ M_3 &= \{l \in B \mid f_l(x^k) > f_l(x^j)\}, \quad m_3 = |M_3|. \end{aligned} \tag{18}$$

Obviously, $m_1, m_2, m_3 \ge 0$ and $m_1 + m_2 + m_3 = b$. Therefore,

$$a_{jk}^2 = m_1 \cdot 0 + m_2 \cdot 0.5 + m_3 \cdot 1 \le b. \tag{19}$$

(iii) When $l \in C$. We have

$$a_{ik}^3 = \sum_{l\in C} a_{ikl} = 0.5 \cdot c. \tag{20}$$

On the other hand, let

$$\begin{aligned} M_1 &= \{l \in C \mid f_l(x^k) < f_l(x^j)\}, \quad m_1 = |M_1|; \\ M_2 &= \{l \in C \mid f_l(x^k) = f_l(x^j)\}, \quad m_2 = |M_2|. \end{aligned} \tag{21}$$

Note that it is not possible to have $f_l(x^k) > f_l(x^j)$. Otherwise, we will have $f_l(x^i) > f_l(x^j)$, which is contrary to $\mathbf{F}(x^i) \le \mathbf{F}(x^j)$. Again, we have $m_1, m_2 \ge 0$ and $m_1 + m_2 = c$. Thus,

$$a_{jk}^3 = \sum_{l\in C} a_{jkl} = m_1 \cdot 0 + m_2 \cdot 0.5 \le 0.5c. \tag{22}$$

In light of (16)–(22), we have

$$a_{ik} = \sum_{\alpha=1}^{3} \alpha_{ik}^{\alpha} \ge a_{jk} = \sum_{\alpha=1}^{3} a_{jk}^{\alpha}. \tag{23}$$

□

Theorem 11. *If $K_e = \max_{i\in N}\{K_i\}$, then the solution x^e corresponding to K_e must be an efficient solution (Pareto solution).*

Proof. If x^e is not an efficient solution, then there exists an $x^s \in X$, such that

$$\mathbf{F}(x^s) \le \mathbf{F}(x^e). \tag{24}$$

Therefore, according to Lemma 9,

$$a_{es} < a_{se}. \tag{25}$$

Due to Lemma 10, we have, for any x^j $(j \in N, j \ne e, s)$,

$$a_{ej} \le a_{sj}. \tag{26}$$

Based on (25), (26) and $a_{ee} = a_{ss} = 0$, we have

$$K_e = \sum_{j\in N} a_{ej} < \sum_{j\in N} a_{sj} = K_s, \tag{27}$$

which is a contradiction to the assumption $k_e = \max_{i\in N}\{K_i\}$. The proof is completed. □

Based on Theorem 11, the following results are obvious.

Corollary 12. *If x^a is an absolute optimal solution, then*

$$K_a = \max_{j\in N} \{K_j\}. \tag{28}$$

Corollary 13. *If x^e is corresponding to $K_e = \max_{j\in N}\{K_j\}$ and x^s is corresponding to $K_s = \max_{j\in N\setminus\{e\}}\{K_j\}$, then x^s is an efficient solution of the population without x^e.*

Theorem 11 and Corollary 13 reveal the rationality of the optimum order method because the best solution must be an efficient solution, and the second best solution, although may not be an efficient solution, is an efficient solution without the best one. This statement is true until the solution with the smallest total optimal number (ranked as the last reasonable solution). Therefore, It is reasonable for us to choose solutions with large total optimal numbers as parents for the next generation.

3.2. The Elitism Metric. Over the last two decades, a number of different evolutionary algorithms were suggested to solve multiobjective optimization problems, such as MOGA [37], NSGA [23], NPGA [20], SPEA [24], PAES [38], and NSGA-II [4]. Among them, MOGA, NSGA, and NPGA did not introduce any elitism strategy at all. Therefore, although they were successful in finding solutions for many test problems and a number of engineering problems, researchers realized that they are still needed to be improved in the sense of obtaining better Pareto solutions [28]. Elitism is one of the main issues considered to be improved. After that, SPEA, PAES, and NSGA-II started to take into account the elitism of solutions.

SPEA introduces an external population which stores all nondominated solutions discovered so far beginning from the initial population. At each generation, a combined population based on the external population and the current population is generated firstly. Then, all the nondominated solutions in the combined population are identified and assigned a fitness based on the number of solutions they dominate. Furthermore, the dominated solutions are assigned a fitness which is worse than the worst fitness of the nondominated solution. Then, the next generation is selected based on the fitness of each solution. This elitism strategy makes sure that the search is directed toward the nondominated solutions. Although the computational complexity of SPEA is $O(MN^3)$ (where M is the number of objectives and N is the size of the population), with proper bookkeeping, the complexity can be reduced to $O(MN^2)$.

PAES uses a single-parent single-offspring evolutionary algorithm which is similar to $(1 + 1)$-evolution strategy. In PAES, the elitism strategy is implemented by comparing between one parent and one offspring. If the offspring dominates the parent, the offspring is accepted as the next parent; on the other hand, if the parent dominates the offspring, the offspring is replaced by another mutated solution (a new offspring) and the comparison between the parent and offspring is made again. However, if the parent and the offspring do not dominate each other, the choice between them is made by comparing them with an archive of the best solutions found so far. More specifically, if the offspring dominates any solution stored in the archive, then the offspring is accepted by the archive and all the dominated solutions are eliminated from the archive. On the contrary, if the offspring does not dominate any solution stored in the archive, then the parent and offspring are accepted by the archive after checking the diversity. The computational complexity of PAES is calculated as $O(aMN)$, where a is the length of the archive. Since the archive size is usually chosen proportional to the population size N, the overall complexity of PAES is $O(MN^2)$.

NSGA implements the elitism strategy using naive nondominated sorting approach. For a population of solutions, the naive nondominated sorting first identifies the nondominated solutions in the population and then the nondominated solutions without considering the solutions which have already been identified. This process is repeated until all the solutions in the population have been identified. In this way, the solutions are classified into different Pareto frontiers and selected based on their Pareto ranking. The native nondominated sorting approach is based on the definition of Pareto frontier. It is easy to be implemented, but in the worst case, the task of identifying all the Pareto frontier requires a computational complexity of $O(MN^3)$. In order to reduce the complexity of NSGA, NSGA-II improved the sorting approach and presented the fast nondominated sorting approach. NSGA-II introduced two entities, domination count (n_p) and dominated count (S_p). The domination count represents the number of solutions which dominates the solution p; the dominated count represents the number of solutions which are dominated by the solutions p.

All solution in the first nondominated front will have their domination count as zero. Now, for each solution p with $n_p = 0$, we visit each member (q) of its set S_p and reduce its domination count by one. In doing so, if for any member q the domination count becomes zero, we put it in a separate list Q. These members belong to the second nondominated front. Now, the above procedure is continued with each member of Q and the third front is identified. This process continues until all frontiers are identified. It is evident to observe that the naive nondominated sorting approach and the fast nondominated sorting approach lead to the same Pareto ranking for a population of solutions. The overall computational complexity of the fast nondominated sorting approach is $O(MN^2)$.

The previous elitism strategies use the definitions of domination or Pareto ranking, but they do not quantify the elitism of solutions. In this paper, we introduce a new elitism strategy which is still based on Pareto ranking but quantifies the elitism of solutions.

Definition 14. Suppose that P_1 is a population of solutions, we define the *elitism metric* of the nondominated solutions in P_1, say F_1, as 1; define the elitism metric of the nondominated solutions in $P_2 = P_1 \setminus F_1$, say F_2, as 2; and so forth; and define the elitism metric of the nondominated solutions in $P_i = P_{i-1} \setminus F_{i-1}$, say F_i as i.

Remark 15. The process in Definition 14 is the same as the native nondominated sorting approach. And the elitism metric of each solution is actually its index of Pareto ranking.

According to Definition 14, the elitism measure of solutions can be taken as a function of the solutions, the value of the function is the elitism metric of solutions. We call this function *the elitism function*. Mathematically,

$$\begin{gathered}\theta : P_1 \longrightarrow \mathbb{Z}^+, \\ \text{where } P_1 \subset \mathbb{R}^n \text{ is a population and} \\ \mathbb{Z}^+ \text{ is the positive integer.}\end{gathered} \tag{29}$$

It is obvious that a small value of $\theta(x)$ means that x enjoys a good Pareto ranking. Therefore, in order to find solutions with better Pareto ranking, we should minimize function $\theta(x)$. In the following, we present a method to calculate the value of function θ.

Suppose that $P_1 = \{x_1, x_2, \ldots, x_N\}$, we calculate the value of $\alpha(x_i, x_j)$ by

$$\alpha\left(x_i, x_j\right) = \begin{cases} 1, & \text{if } x_i \text{ dominates } x_j; \\ -1, & \text{if } x_i \text{ dominated by } x_j; \\ 0, & \text{if } x_i \text{ and } x_j \text{ are not dominated by each other.} \end{cases} \tag{30}$$

Then we construct the domination table (see Table 2).

Obviously, entries of Table 2 are just −1, 0, and 1. Now we successively check every row of the table, if any row, say the ith one, whose entries are only 0 or 1; then we can claim that there is no solution dominating x_i, which means that x_i's elitism metric is 1. Then discard the rows and columns

Table 2: The domination table of P_1.

	x_1	x_2	$\cdots$	x_{N-1}	x_N
x_1	—	$\alpha(x_1, x_2)$	$\cdots$	$\alpha(x_1, x_{N-1})$	$\alpha(x_1, x_N)$
x_2	$\alpha(x_2, x_1)$	—	$\cdots$	$\alpha(x_2, x_{N-1})$	$\alpha(x_2, x_N)$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
x_{N-1}	$\alpha(x_{N-1}, x_1)$	$\alpha(x_{N-1}, x_2)$	$\cdots$	—	$\alpha(x_{N-1}, x_N)$
x_N	$\alpha(x_N, x_1)$	$\alpha(x_N, x_2)$	$\cdots$	$\alpha(x_N, x_{N-1})$	—

in which these solutions with elitism metric of 1 locate. This means that we do not consider these solutions any more in the following process since they have already been identified. This process is repeated again and the identified solutions are assigned with elitism metric of 2 and so forth until all the solutions are identified.

In Table 2, it is evident that $\alpha(x_i, x_j) = -\alpha(x_j, x_i)$, where $1 \le i \ne j \le N$. So the computational complexity of assigning elitism metrics for the population is $O((1/2)M(N-1)^2)$. We provide the following example to illustrate the process described above.

Figure 1 illustrated a population of solutions. There are 35 solutions in this population. The numbers on the side are indexes of these solutions. Table 2 is the domination table of this population. First, we successively check rows of Table 3; rows with only 0 and 1 are 1, 2, 5, 8, 9, 10, 16, 17, 19, 20, and 28. So solutions corresponding to these rows are in the first Pareto frontier and should be assigned elitism metric 1. Then, discard the corresponding rows and columns from the domination table and do the same search again. We assign the solutions corresponding to the identified rows with elitism metric 2. This process is repeated until all the rows of the domination table are discarded. We depict the elitism metric assignment in Figure 2.

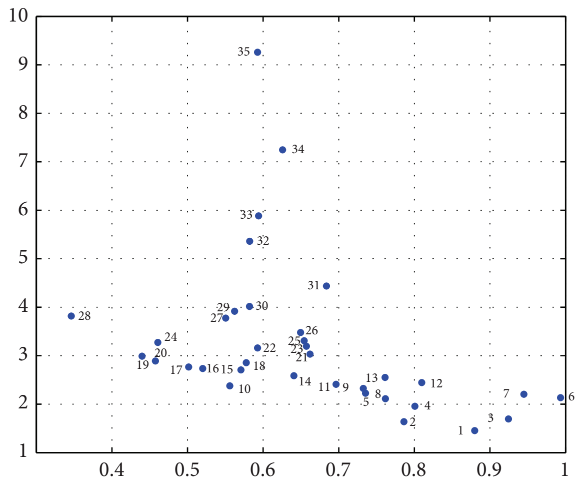


Figure 1: The population of solutions.

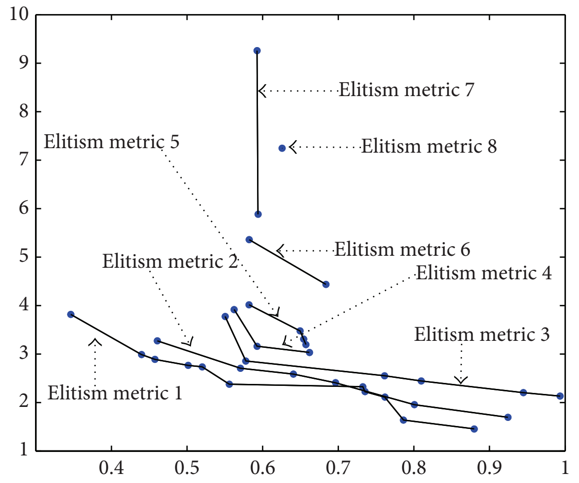


Figure 2: The elitism metric assignment for the population.

3.3. The Diversity Metric. As discussed before, diversity of solutions is another criterion for solutions obtained by the multiobjective genetic algorithm. In order to obtain uniformly distributed solutions, it is important to maintain the diversity of the population in the iteration process. Without this, solutions in the population tend to form relatively few clusters. This phenomenon is known as *genetic drift*. Several approaches have been developed to maintain the diversity of solutions. Among them, the sharing function approach [19, 39] is one of the earliest approaches. The sharing function approach degrades the objective fitness of solutions by setting a threshold θ_{share}. This threshold is still called *sharing parameter*. For a solution x_i in the current population, a *niche count* m_i is first calculated. The niche count $m_i = 1$ if there is no solution in the θ_{share}-niche (neighborhood) of x_i (i.e., there is no solution whose distance with x_i is less than θ_{share}); the niche count $m_i > 1$ if there are some solutions in the θ_{share}-niche (neighborhood) of x_i (i.e., there are some solutions whose distances with x_i) less than θ_{share}. Then, the degradation is obtained through dividing the objective fitness by the niche count, that is, f_i/m_i. This proportion is called the *shared fitness*. The original NSGA [23] used the sharing function approach to maintain the diversity of solutions. The performance of the sharing function approach terribly depends on the sharing parameter θ_{share}. However, θ_{share} is a user-decided parameter and it is difficult to choose a proper one. Since each solution must be compared with all other solutions in the population, the overall complexity of the sharing function approach is $O(MN^2)$.

Deb et al. [4] introduced the crowding distance approach to maintain the diversity of solutions. This approach calculates the density estimation of a solution by summarizing the average distance of two solutions on either side of this solution along each of the objective function. The density estimation, which is still called crowding distance, serves as an estimate of the perimeter of the cuboid formed by using the nearest neighbors as the vertices. Obviously, a solution with a smaller value of crowding distance is more crowded by other solutions. Thus, the crowding distance can be considered as a measure of the diversity of a solution. The crowding distance approach does not require any user-defined parameter.

Table 3: The domination table of P_1.

x	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
1	—	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	—	1	1	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	–1	–1	—	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	–1	0	—	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	—	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	–1	–1	–1	–1	–1	—	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	–1	–1	–1	–1	–1	0	—	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	—	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	—	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	—	1	1	1	1	1	0	0	1	0	0	1	1	1	0	1	1	0	0	1	1	1	1	1	1	1
11	0	0	0	0	0	0	0	0	0	–1	—	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	–1	0	–1	–1	0	0	–1	–1	–1	–1	—	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	–1	–1	–1	–1	0	—	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	–1	0	0	0	—	0	0	0	0	0	0	1	0	1	0	1	1	0	0	0	0	1	0	0	0	0
15	0	0	0	0	0	0	0	0	0	–1	0	0	0	0	—	0	0	1	0	0	1	1	1	0	1	1	0	0	0	1	1	1	1	1	1
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	—	0	1	0	0	1	1	1	0	1	1	1	0	1	1	1	1	1	1	1
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	—	1	0	0	1	1	1	0	1	1	1	0	1	1	1	1	1	1	1
18	0	0	0	0	0	0	0	0	0	–1	0	0	0	0	–1	1	–1	—	0	0	1	1	1	0	1	1	0	0	0	1	1	1	1	1	1
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	—	0	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	—	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
21	0	0	0	0	0	0	0	0	0	–1	0	0	0	–1	–1	1	–1	–1	–1	–1	—	0	0	0	0	0	0	0	0	0	1	0	0	0	0
22	0	0	0	0	0	0	0	0	0	–1	0	0	0	0	–1	1	–1	–1	–1	–1	0	—	1	0	1	1	0	0	0	0	1	0	1	1	1
23	0	0	0	0	0	0	0	0	0	–1	0	0	0	–1	–1	1	–1	–1	–1	–1	0	–1	—	0	0	0	0	0	0	0	1	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–1	–1	0	0	0	—	1	1	1	0	1	1	1	1	1	1	1
25	0	0	0	0	0	0	0	0	0	–1	0	0	0	–1	–1	1	–1	–1	–1	–1	0	–1	0	–1	—	0	0	0	0	0	1	0	0	0	0
26	0	0	0	0	0	0	0	0	0	–1	0	0	0	–1	–1	1	–1	–1	–1	–1	0	–1	0	–1	0	—	0	0	0	0	1	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	–1	0	–1	–1	0	0	0	–1	0	0	—	0	1	1	1	1	1	1	1
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	—	1	1	1	1	1	1	1
29	0	0	0	0	0	0	0	0	0	–1	0	0	0	0	0	1	–1	0	–1	–1	0	0	0	–1	0	0	–1	–1	—	1	1	1	1	1	1
30	0	0	0	0	0	0	0	0	0	–1	0	0	0	0	–1	1	–1	–1	–1	–1	0	0	0	–1	0	0	–1	–1	–1	—	1	1	1	1	1
31	0	0	0	0	0	0	0	0	0	–1	0	0	0	–1	–1	1	–1	–1	–1	–1	–1	–1	–1	–1	–1	–1	–1	–1	–1	–1	—	0	0	0	0
32	0	0	0	0	0	0	0	0	0	–1	0	0	0	0	–1	1	–1	–1	–1	–1	0	0	0	–1	0	0	–1	–1	–1	–1	0	—	1	1	1
33	0	0	0	0	0	0	0	0	0	–1	0	0	0	0	–1	1	–1	–1	–1	–1	0	–1	0	–1	0	0	–1	–1	–1	–1	0	–1	—	1	0
34	0	0	0	0	0	0	0	0	0	–1	0	0	0	0	–1	1	–1	–1	–1	–1	0	–1	0	–1	0	0	–1	–1	–1	–1	0	–1	–1	—	0
35	0	0	0	0	0	0	0	0	0	–1	0	0	0	0	–1	1	–1	–1	–1	–1	0	–1	0	–1	0	0	–1	–1	–1	–1	0	–1	0	0	—

Its computational complexity is $O(MN \log N)$, which is better than the sharing function approach. However, the crowing distance just reflects the local diversity of a solution.

The cell-based density approach [6, 26, 38, 40, 41] divides the objective space into p-dimensional cells. The number of solutions in each cell is defined as the *density of the cell*, and the *density of a solution* is equal to the density of the cell in which the solution locates. An efficient approach to dynamically divide the objective function space into cells is proposed by Lu and Yen [40, 41]. The main advantage of the cell-based density approach is that a global density map of the objective function value space can be obtained. But the process of the cell-based density approach is complicated. For a single solution, in the worst case, it requires $O(LM)$ comparison to locate the solution into a proper cell, where L is the number of cells for each objective function. Thus, the total complexity of the cell-based density approach is $O(LMN)$. And furthermore, L is a user-defined parameter which is not easy to be chosen properly.

In this section, we define the metric about measuring the degree of crowdedness around a solution. Then, a specifical diversity metric is introduced.

Definition 16. Suppose that P_1 is a population, x_i and x_j are two solutions from P_1, and ϕ_i and ϕ_j are two scalars corresponding to x_i and x_j, respectively. We call ϕ_i and ϕ_j the *diversity metrics* of x_i and x_j, respectively, if $\phi_i < \phi_j$ implies that solutions around x_j are more crowded than those around x_i.

Definition 16 is just a qualitative description of the diversity metric. In the following, we introduce a specifical definition of the diversity metric which quantitatively measures the degree of crowdedness around a solution.

Definition 17. Suppose that P_1 is a population with N solutions, x_i is a solution in P_1. We provide a function $\phi : P_1 \mapsto \mathbb{R}^+$ as

$$\phi(x_i) = \sum_{j=1, j\neq i}^{N} \beta(d_{ij}), \tag{31}$$

where $\beta : \mathbb{R}^+ \mapsto \mathbb{R}^+$ is a decrease function and d_{ij} is the Euclidean distance between solutions x_i and x_j. Then we call the value of $\phi(x_i)$ the *global diversity metric* of x_i.

It is obvious that the global diversity metric is coincident with the definition of diversity metric. One improvement of the global diversity metric is that it reflects the diversity of x_i with respect to all the other solutions since the function ϕ involves the whole population. The requirement for calculating the global diversity metric is only the Euclidean distance between any two solutions, so the computational complexity of the global diversity metric is $O((1/2)(N-1)^2)$. A proper choice of function β is

$$\beta(d) = 1 - \frac{1}{D}d, \tag{32}$$

where $D = \max_{1\leq i,j\leq N}\{d_{ij}\}$.

As an example, Figure 3 illustrates the global diversity metric of the population presented in Figure 1. From Figure 3, one can observe that these solutions which are less crowded enjoy smaller global diversity metrics, such as solutions x_{35} and x_{34} (the index of the solution can be found in Figure 1) with 7.1157 and 15.6234, respectively. On the contrary, these solutions which are more crowded have larger global diversity metrics, such as solutions x_{15}, x_{17}, and x_{21} with 19.3926, 29.3777, and 29.3826, respectively.

3.4. A New Selection Operator. As discussed before, the core criteria for multiobjective optimization are to obtain solutions with better elitism and, at the same time, maintain the diversity of solutions. If we tackle these criteria from a multiobjective optimization point of view, we need to solve a multiobjective optimization problem with two objectives: minimizing the elitism metric and the global diversity metric of the solution. Noting the definition of the elitism metric $\theta(x)$ and the global diversity metric $\phi(x)$ presented in the previous subsections, we design the following multiobjective subproblem:

$$\begin{aligned} &\text{Minimize} \quad \mathbf{H}(x) \\ &\text{Subject to} \quad x \in P_1, \end{aligned} \tag{33}$$

where $\mathbf{H}(x) = (\theta(x), \phi(x))^T$ is a multiobjective function and $P_1 \subset X$ is a population of candidate solutions. It is clear that the image of $\mathbf{H}(\cdot)$

$$H = \left\{(\theta(x), \phi(x))^T \mid x \in P_1\right\} \tag{34}$$

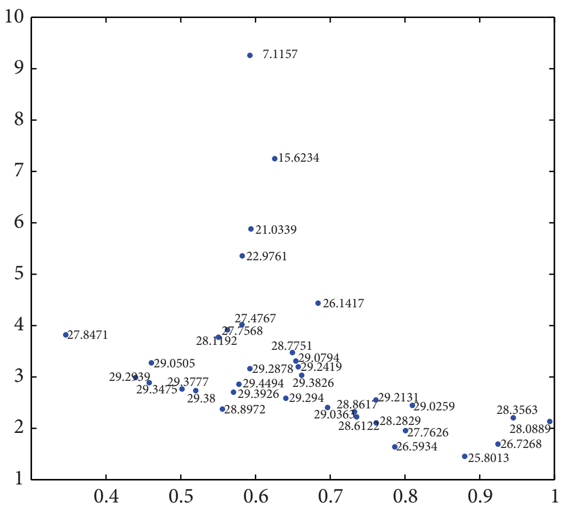


Figure 3: The global diversity metric for the population.

is a finite set whose cardinal is the same as P_1. One property of the multiobjective subproblem (33) is that its domain P_1 and image set H are finite; that is, the problem (33) is discrete. Thus, if we rank points in H by the optimum order method (Algorithm 8), it is reasonable to say that those points who have smaller optimal orders are better points; that is, solutions corresponding to these points should be maintained to the next generation. This process is summarized by the following algorithm.

Algorithm 18 (multiobjective selection operator). Consider the following steps.

Step 1. Input the selection pool P_1 and its objective function values $F(P_1) = \{\mathbf{F}(x) \mid x \in P_1\}$. Input the prefixed population size N (N is less than the cardinal of P_1).

Step 2. Calculate the elitism metric and global diversity metric of the solution in P_1 using the function (29) and (31), respectively. Let the result be

$$\begin{aligned} \Theta &= \{\theta(x) \mid x \in P_1\}, \\ \Phi &= \{\phi(x) \mid x \in P_1\}. \end{aligned} \tag{35}$$

Step 3. Construct the objective function value space of the multiobjective subproblem

$$H = \left\{(\theta_x, \phi_x)^T \mid \theta_x \in \Theta,\ \phi_x \in \Phi\right\}. \tag{36}$$

Step 4. Rank points in H by the optimum order method (Algorithm 8) and take the first N solutions according to their optimal orders as parents for the next generation.

In Table 4, we use the multiobjective selection operator to evaluate the population presented in Figure 1. In this table, θ_i, ϕ_i, K_i, and σ_i represent the elitism metric, global diversity metric, total optimal number, and optimal order of solution

Table 4: The total optimal number and optimal order without pretreatment of the diversity metric.

x_i	θ_i	ϕ_i	K_i	σ_i
x_1	1	25.8013	59.0	1
x_2	1	26.5934	57.0	2
x_3	2	26.7268	47.5	5
x_4	2	27.7626	44.5	8
x_5	1	28.2829	49.0	4
x_6	3	28.0889	36.5	10
x_7	3	28.3563	33.5	17
x_8	1	28.6122	47.0	6
x_9	1	28.8617	45.0	7
x_{10}	1	28.8972	44.0	9
x_{11}	2	29.0363	33.5	18
x_{12}	3	29.0259	28.5	26
x_{13}	3	29.2131	24.5	28
x_{14}	2	29.2940	26.5	27
x_{15}	2	29.3926	21.5	30
x_{16}	1	29.3800	32.0	25
x_{17}	1	29.3777	33.0	20
x_{18}	3	29.4494	14.5	34
x_{19}	1	29.3475	34.0	16
x_{20}	1	29.2939	36.0	11
x_{21}	4	29.3826	12.0	35
x_{22}	4	29.2878	18.0	31
x_{23}	5	29.2419	15.5	33
x_{24}	2	29.0505	32.5	22
x_{25}	5	29.0794	17.5	32
x_{26}	5	28.7751	23.5	29
x_{27}	3	28.1192	35.5	12
x_{28}	1	27.8471	52.0	3
x_{29}	4	27.7568	35.0	14
x_{30}	5	27.4767	32.5	23
x_{31}	6	26.1417	32.5	24
x_{32}	6	22.9761	34.5	15
x_{33}	7	21.0339	33.5	19
x_{34}	8	15.6234	33.0	21
x_{35}	7	7.1157	35.5	13

Table 5: The total optimal number and optimal order with pretreatment of the diversity metric.

x_i	θ_i	ϕ_i	K_i	σ_i
x_1	1	25.8013	59.0	1
x_2	1	26.5934	57.0	2
x_3	2	26.7268	47.5	4
x_4	2	27.7626	44.5	6
x_5	1	28.2829	49.0	5
x_6	3	28.0889	36.5	15
x_7	3	28.3563	33.5	17
x_8	1	28.6122	47.0	7
x_9	1	28.8617	45.0	12
x_{10}	1	28.8972	44.0	8
x_{11}	2	29.0363	33.5	21
x_{12}	3	29.0259	28.5	24
x_{13}	3	29.2131	24.5	26
x_{14}	2	29.2940	—	—
x_{15}	2	29.3926	—	—
x_{16}	1	29.3800	—	—
x_{17}	1	29.3777	33.0	19
x_{18}	3	29.4494	—	—
x_{19}	1	29.3475	34.0	18
x_{20}	1	29.2939	36.0	10
x_{21}	4	29.3826	12.0	27
x_{22}	4	29.2878	18.0	22
x_{23}	5	29.2419	15.5	25
x_{24}	2	29.0505	32.5	13
x_{25}	5	29.0794	17.5	23
x_{26}	5	28.7751	23.5	20
x_{27}	3	28.1192	35.5	9
x_{28}	1	27.8471	52.0	3
x_{29}	4	27.7568	35.0	11
x_{30}	5	27.4767	32.5	14
x_{31}	6	26.1417	32.5	16
x_{32}	6	22.9761	—	—
x_{33}	7	21.0339	—	—
x_{34}	8	15.6234	—	—
x_{35}	7	7.1157	—	—

x_i, respectively. We can observe that the multiobjective selection operator systematically considers the criteria of elitism and diversity. We select 15 solutions whose optimal orders are the smallest as parents of the next generation. They are marked as red squares in Figure 4. However, from Figure 4, distribution of the selected points is not ideal. This is because there are some irregular distributed points, such as points 33, 34, and 35. They are far away from the others, which means that they should be eliminated. But because of their very small diversity metric, they are selected by the selection operator. In order to tackle this issue, we pretreat the diversity metric before running the selection operator.

In sport matches, the statistician always eliminates the highest and the lowest marks before calculating the average. We borrow this idea to pretreat the diversity metric. We simply identify the 10% points with the highest and lowest diversity metric and eliminate them before the running of selection operator. In the example, the 10% points with the highest and lowest diversity metrics are 14, 15, 16, and 18 and 32, 33, 34, and 35, respectively. Thus, we run the selection operator without considering them.

Table 5 presents the total optimal number and optimal order with pretreatment of the diversity metric. It can be observed that points 14–16, 18, and 32–35 are excluded in the ranking process. This gives more chance to some point close to the Pareto frontier. For example, points 24 and 30 (which were ranked as 22 and 23, resp.) were not selected without pretreatment of the diversity metric, but they are selected (ranked as 13 and 14, resp.) after considering the pretreatment of the diversity metric. From Figure 5, we can see that the

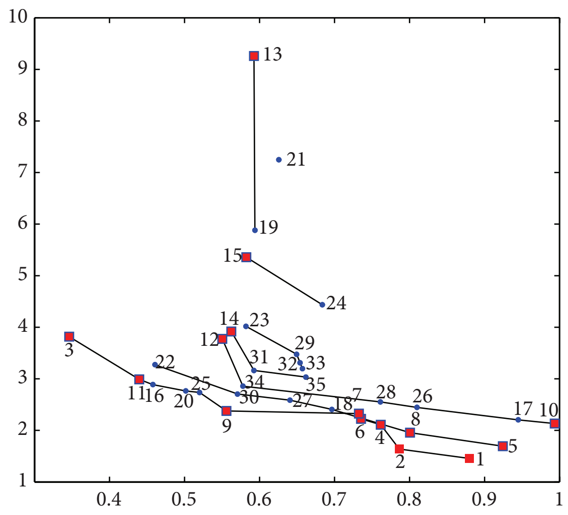


Figure 4: The selected point without pretreatment of the diversity metric.

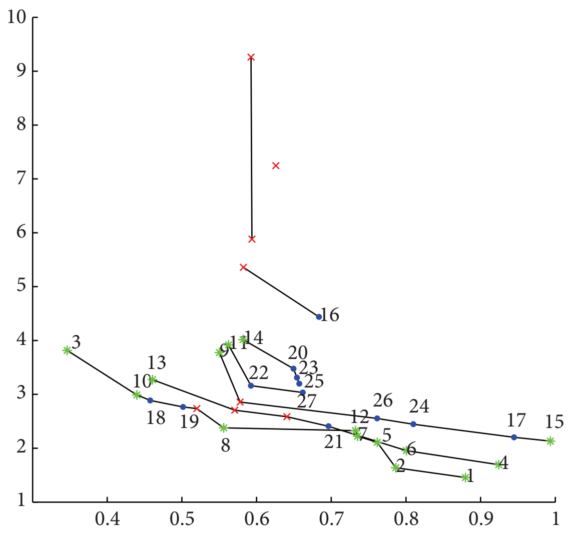


Figure 5: The selected point with pretreatment of the diversity metric.

selected points are now well distributed and closer to the Pareto frontier.

3.5. A New Multiobjective Genetic Algorithm. Given the multiobjective selection operator, we are now ready to propose a new multiobjective genetic algorithm.

Algorithm 19 (optimum order multiobjective genetic algorithm (OOMOGA)). Consider the following.

```
(1) Initialization
    (1.1) Generate the initial population $P(0)$.
    (1.2) Set the crossover rate, mutation rate, and number of maximal generations.
    (1.3) Let $t \leftarrow 0$.
(2) Since the number of maximal generations has not been reached, do the following.
    (2.1) Run the crossover operator: generate the offspring $O_1(t)$.
    (2.2) Run the mutation operator: generate the offspring $O_2(t)$.
    (2.3) Construct a selection pool $S(t)$ by combining $P(t)$, $O_1(t)$, and $O_2(t)$; that is, $S(t) = P(t) \cup O_1(t) \cup O_2(t)$. Compute their multiobjective function values; that is, $FS(t) = \{\mathbf{F}(x) \mid x \in S(t)\}$.
    (2.4) Run the multiobjective selection operator (Algorithm 18) by inputting $S(t)$ and $FS(t)$; the selected solutions are maintained as the parents of the next generation, that is, $P(t+1)$.
    (2.5) Let $t \leftarrow t+1$; go to (2.1).
          End
    End.
```

4. Numerical Experiments

In this section, we investigate the numerical performance of OOMOGA. In order to evaluate the numerical performance of solvers, we use the performance metric IGD proposed in [42]. Suppose that P^* is a set of uniformly distributed points along the Pareto frontier (in the objective function value space). Let A be a set of solutions obtained by a certain solver. Then, the average distance from P^* to A is defined as

$$\mathrm{IGD}(A, P^*) = \frac{\sum_{v \in P^*} d(v, A)}{|P^*|}, \tag{37}$$

where $d(v, A)$ is the minimum Euclidean distance between v and the points in A; that is,

$$d(v, A) = \min_{y \in A} \|v - y\|. \tag{38}$$

In fact, P^* represents a sample set of the real Pareto frontiers, if $|P^*|$ is large enough to approximate the Pareto frontier very well, $\mathrm{IGD}(A, P^*)$ could measure both the diversity and convergence of A in a sense. A smaller $\mathrm{IGD}(A, P^*)$ means the set A is closer to the real Pareto frontier and has better diversity.

The referential algorithms are those submitted to the special session on performance assessment of unconstrained/bound constrained multiobjective optimization algorithms at CEC'09. There are 13 algorithms submitted to the special session. They are listed as follows:

(1) MOEAD [43];

(2) GDE3 [44];

(3) MOEADGM [45];

(4) MTS [46];

(5) LiuLiAlgorithm [47];

(6) DMOEADD [48];

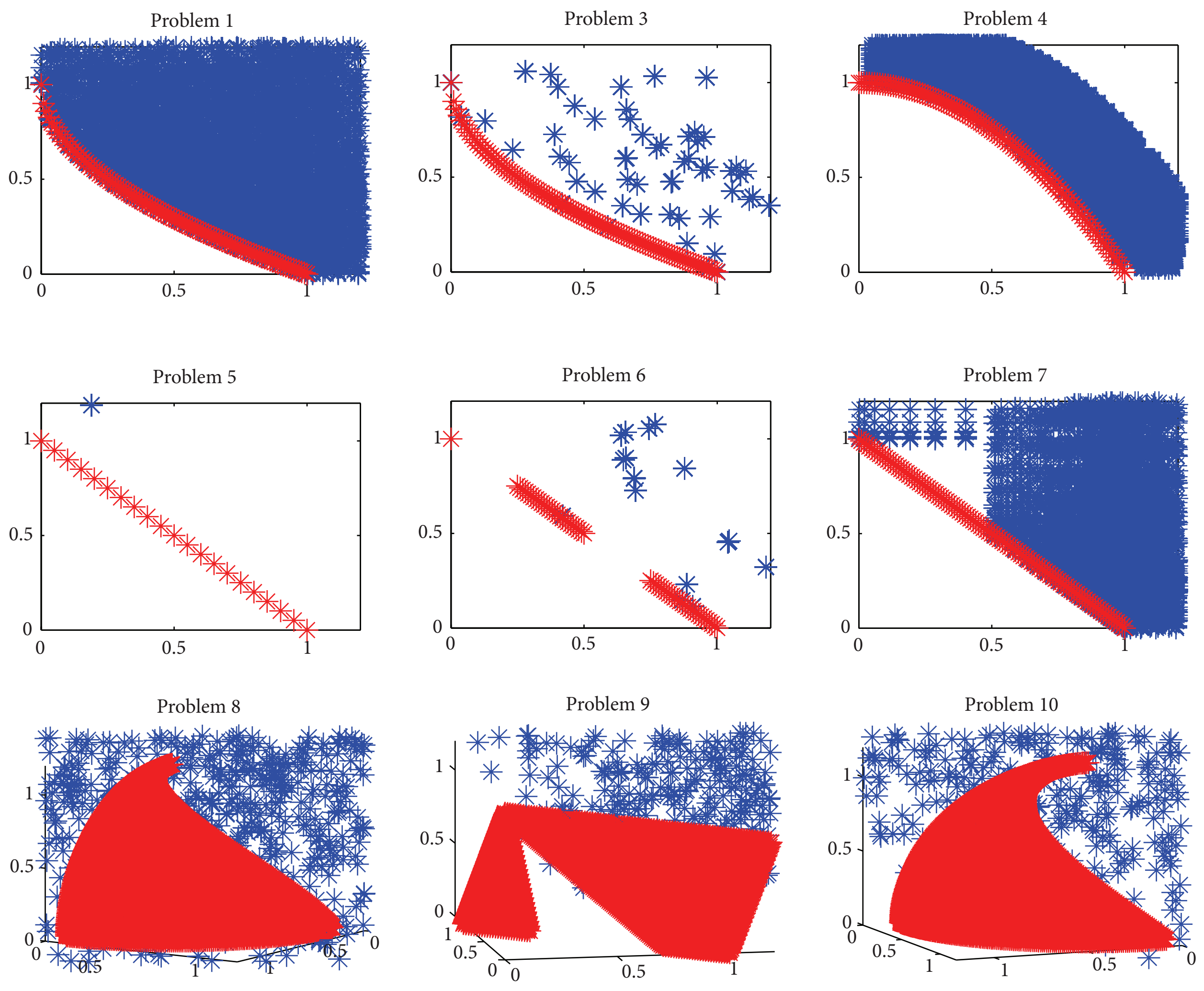


Figure 6: The objective function value space for test problems.

(7) NSGAIILS [49];

(8) OWMOSaDE [50];

(9) ClusteringMOEA [51];

(10) AMGA [52];

(11) MOEP [53];

(12) DECMOSA-SQP [54];

(13) OMOEAII [55].

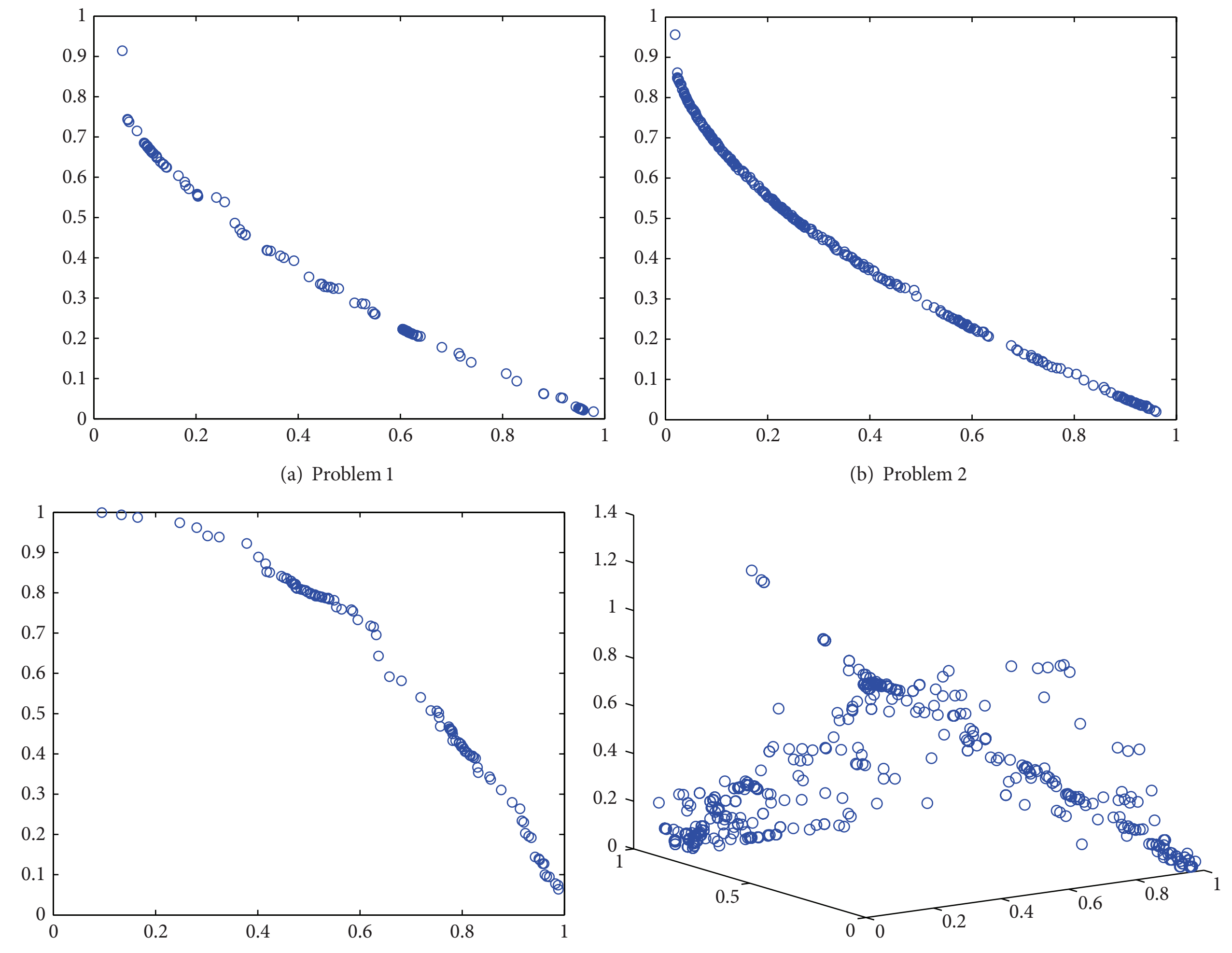
The benchmarks are taken from [42] which are still used in CEC'09. Figure 6 illustrates the objective function value space of these test problems (the figure of test problem 2 is ignored here since it is similar to that of the test problem 1); the red curve/surface represents their Pareto frontiers. Among these test problems, Problems 1–7 have two objective functions, whereas Problems 8–10 have three objective functions. The Pareto solutions of Problems 5, 6, and 9 are disconnected, while the others are connected.

In order to keep consistency with the final report of CEC'09 [56], in the implementation of OOMOGA, we set the population size as 100 for problems with two objectives and 150 for problems with three objectives; the number of function evaluations is less than 300,000. For each test instance, we run OOMOGA independently for 30 times. The numerical performance evaluated by IGD is illustrated in Table 6.

From Table 6, the proposed solver OOMOGA performs the best at solving the test problem 2; its IGD evaluation is 0.00527, which is better than all the other solvers. In solving the test Problem 9, OOMOGA (whose IGD evaluation is 0.0601) performs only worse than DMOEADD (whose IGD evaluation is 0.4896) but better than all the other solvers. In solving the test Problem 3, the IGD evaluation of OOMOGA (0.0331) is ranked in the third position, worse than MOEAD (0.0072) and LiuLiAlgorithm (0.01497). In solving the test Problem 4, the IGD evaluation of OOMOGA (0.0347) is ranked in the fourth one, worse than MTS (0.02356), GDE3 (0.0265), and DECMOSA-SQP (0.03392). The numerical performance of OOMOGA is moderate in solving test Problems 1, 6, and 8; the IGD ranks are 8, 9, and 9, respectively. For test Problems 5 and 10, the numerical performance of the solver

Table 6: The numerical performance evaluated by IGD.

Rank	UF1		UF2		UF3	
1	MOEAD	0.00435	**OOMOGA**	**0.00527**	MOEAD	0.00742
2	GDE3	0.00534	MTS	0.00615	LiuLiAlgorithm	0.01497
3	MOEADGM	0.0062	MOEADGM	0.0064	**OOMOGA**	**0.0331**
4	MTS	0.00646	DMOEADD	0.00679	DMOEADD	0.03337
5	LiuLiAlgorithm	0.00785	MOEAD	0.00679	MOEADGM	0.049
6	DMOEADD	0.01038	OWMOSaDE	0.0081	MTS	0.0531
7	NSGAIILS	0.01153	GDE3	0.01195	ClusteringMOEA	0.0549
8	**OOMOGA**	**0.0118**	LiuLiAlgorithm	0.0123	AMGA	0.06998
9	OWMOSaDE	0.0122	NSGAIILS	0.01237	DECMOSA-SQP	0.0935
10	ClusteringMOEA	0.0299	AMGA	0.01623	MOEP	0.099
11	AMGA	0.03588	MOEP	0.0189	OWMOSaDE	0.103
12	MOEP	0.0596	ClusteringMOEA	0.0228	NSGAIILS	0.10603
13	DECMOSA-SQP	0.07702	DECMOSA-SQP	0.02834	GDE3	0.10639
14	OMOEAII	0.08564	OMOEAII	0.03057	OMOEAII	0.27141
Rank	UF4		UF5		UF6	
1	MTS	0.02356	MTS	0.01489	MOEAD	0.00587
2	GDE3	0.0265	GDE3	0.03928	MTS	0.05917
3	DECMOSA-SQP	0.03392	AMGA	0.09405	DMOEADD	0.06673
4	**OOMOGA**	**0.0347**	LiuLiAlgorithm	0.16186	OMOEAII	0.07338
5	AMGA	0.04062	DECMOSA-SQP	0.16713	ClusteringMOEA	0.0871
6	DMOEADD	0.04268	OMOEAII	0.1692	MOEP	0.1031
7	MOEP	0.0427	MOEAD	0.18071	DECMOSA-SQP	0.12604
8	LiuLiAlgorithm	0.0435	MOEP	0.2245	AMGA	0.12942
9	OMOEAII	0.04624	ClusteringMOEA	0.2473	**OOMOGA**	**0.1411**
10	MOEADGM	0.0476	DMOEADD	0.31454	LiuLiAlgorithm	0.17555
11	OWMOSaDE	0.0513	OWMOSaDE	0.4303	OWMOSaDE	0.1918
12	NSGAIILS	0.0584	NSGAIILS	0.5657	GDE3	0.25091
13	ClusteringMOEA	0.0585	**OOMOGA**	**0.7695**	NSGAIILS	0.31032
14	MOEAD	0.06385	MOEADGM	1.7919	MOEADGM	0.5563
Rank	UF7		UF8		UF9	
1	MOEAD	0.00444	MOEAD	0.0584	DMOEADD	0.04896
2	LiuLiAlgorithm	0.0073	DMOEADD	0.06841	**OOMOGA**	**0.0601**
3	MOEADGM	0.0076	LiuLiAlgorithm	0.08235	NSGAIILS	0.0719
4	DMOEADD	0.01032	NSGAIILS	0.0863	MOEAD	0.07896
5	MOEP	0.0197	OWMOSaDE	0.0945	GDE3	0.08248
6	NSGAIILS	0.02132	MTS	0.11251	LiuLiAlgorithm	0.09391
7	ClusteringMOEA	0.0223	AMGA	0.17125	OWMOSaDE	0.0983
8	DECMOSA-SQP	0.02416	OMOEAII	0.192	MTS	0.11442
9	GDE3	0.02522	**OOMOGA**	**0.2137**	DECMOSA-SQP	0.14111
10	OMOEAII	0.03354	DECMOSA-SQP	0.21583	MOEADGM	0.1878
11	MTS	0.04079	ClusteringMOEA	0.2383	AMGA	0.18861
12	AMGA	0.05707	MOEADGM	0.2446	OMOEAII	0.23179
13	OWMOSaDE	0.0585	GDE3	0.24855	ClusteringMOEA	0.2934
14	**OOMOGA**	**0.1267**	MOEP	0.423	MOEP	0.342
Rank	UF10					
1	MTS	0.15306				
2	DMOEADD	0.32211				
3	AMGA	0.32418				
4	MOEP	0.3621				
5	DECMOSA-SQP	0.36985				
6	ClusteringMOEA	0.4111				
7	GDE3	0.43326				
8	LiuLiAlgorithm	0.44691				
9	MOEAD	0.47415				
10	MOEADGM	0.5646				
11	OMOEAII	0.62754				
12	**OOMOGA**	**0.7269**				
13	OWMOSaDE	0.743				
14	NSGAIILS	0.84468				



(a) Problem 1

(b) Problem 2

(c) Problem 4

(d) Problem 9

Figure 7: The approximated Pareto frontier of Problems 1, 2, 4, and 9.

OOMOGA is not so good, ranking in 13 and 12, respectively. The worst performance of OOMOGA appears in solving the test Problem 7; the IGD evaluation is 0.1267, which is worse than all the other solvers.

It is not uncommon that the numerical performance of the proposed solver OOMOGA is unstable among different test problems because the numerical results are not only affected by the performance of solvers, but also impacted by the linearity and structure of objective function themselves. Furthermore, the crossover and mutation operators are also affected by the distribution of points in the objective function value space. Generally speaking, if a new point generated by the crossover or mutation operators has a higher probability of locating around the Pareto frontier, then the Pareto frontier can be well approximated by the solver, for example, the test Problems 2, 3, and 4. On the contrary, if it is hard for the crossover or mutation operators to generate new point around the Pareto frontier, then the problem is difficult to be solved by MOGAs, for instance, the test Problems 5, 6, and 7. In fact, this instability still appears in the other solvers, such as MOEAD which is reported to be the best solver in [56]. The IGD evaluation of MOEAD in solving the test Problems 4, 5, and 10 is not very good, ranking in 14, 7, and 9, respectively.

Figure 7 demonstrates Pareto frontiers of test Problems 1, 2, 4, and 9, respectively. From Figures 7(a) and 7(b), we can observe that, for the test Problems 1 and 2, the proposed solver obtained very good representations of their Pareto frontiers. Results for the test Problem 4 (see Figure 7(c)) is not very uniformly distributed, which may affect the performance of IGD evaluation. Figure 7(d) illustrates the approximate Pareto frontier of the test Problem 9; we can see that the structure is more or less an approximation of the real Pareto frontier.

5. Conclusion

This paper proposed a solver based on genetic algorithm for multiobjective optimization. In the process of using genetic algorithm to solve multiobjective optimization, the evolutionary procedure prefers to select individuals with better elitism and diversity. Therefore, in the algorithm, we have to consider the trade-off between elitism and diversity. To tackle this issue, we first developed strategies to measure the elitism and diversity of populations. Then we used the proposed elitism and diversity metrics to construct a discrete multiobjective subproblem. Finally, the new selection operator is designed by applying the optimum order method to solve the multiobjective subproblem. We tested the proposed solver by the test instances used in CEC'09 and compared the numerical result with the referential solvers proposed in CEC'09. The numerical performance analysis shows that the proposed solver is good at solving problems whose objective function value space is high density around the Pareto frontier.

We will further study this topic from the following two points of view.

(i) Improving the performance of crossover and mutation operators of the genetic algorithm: the crossover and mutation operators are very important for the diversity of solutions. However, the distribution of new individuals generated by the current crossover and mutation operators is coincident with the distribution of parents. This leads the excessive search of the "rich" area, but insufficient search of the "poor" area.

(ii) Introducing some more reasonable diversity metrics: it is very important to maintain the diversity of population in the evolutionary process of genetic algorithm. However, measuring diversity is not an easy task. The global diversity metric proposed in this paper is a proper diversity metric, but it is far from perfect.

Conflict of Interests

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

Acknowledgments

Changzhi Wu was partially supported by Australian Research Council Linkage Program LP140100873, Natural Science Foundation of China (61473326), and Natural Science Foundation of Chongqing (cstc2013jcyjA00029 and cstc2013jjB0149).

References

[1] C. J. Li, C. D. Li, X. F. Liao, and T. W. Huang, "Impulsive effects on stability of high-order BAM neural networks with time delays," *Neurocomputing*, vol. 74, no. 10, pp. 1541–1550, 2011.

[2] C. J. Li, W. W. Yu, and T. W. Huang, "Impulsive synchronization schemes of stochastic complex networks with switching topology: average time approach," *Neural Networks*, vol. 54, pp. 85–94, 2014.

[3] C. J. Li, D. Y. Gao, C. Liu, and G. Chen, "Impulsive control for synchronizing delayed discrete complex networks with switching topology," *Neural Computing and Applications*, vol. 24, no. 1, pp. 59–68, 2014.

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

[5] M. Ehrgott, J. Ide, and A. Schöbel, "Minmax robustness for multi-objective optimization problems," *European Journal of Operational Research*, vol. 239, no. 1, pp. 17–31, 2014.

[6] Q. Long, "A constraint handling technique for constrained multi-objective genetic algorithm," *Swarm and Evolutionary Computation*, vol. 15, pp. 66–79, 2014.

[7] G. Mavrotas, O. Pechak, E. Siskos, H. Doukas, and J. Psarras, "Robustness analysis in Multi-Objective Mathematical Programming using Monte Carlo simulation," *European Journal of Operational Research*, vol. 240, no. 1, pp. 193–201, 2015.

[8] Y. T. Qi, X. L. Ma, F. Liu, L. C. Jiao, J. Y. Sun, and J. S. Wu, "MOEA/D with adaptive weight adjustment," *Evolutionary Computation*, vol. 22, no. 2, pp. 231–264, 2014.

[9] P. C. Fishburn, "Utility theory for decision making," DTIC Document, 1970.

[10] I. Y. Kim and O. L. De Weck, "Adaptive weighted-sum method for bi-objective optimization: pareto front generation," *Structural and Multidisciplinary Optimization*, vol. 29, no. 2, pp. 149–158, 2005.

[11] R. T. Marler and J. S. Arora, "The weighted sum method for multi-objective optimization: new insights," *Structural and Multidisciplinary Optimization*, vol. 41, no. 6, pp. 853–862, 2010.

[12] J. J. Liu, C. Z. Wu, G. Wu, and X. Y. Wang, "A novel differential search algorithm and applications for structure design," Submitted.

[13] Q. Long and C. Z. Wu, "A hybrid method combining genetic algorithm and Hooke-Jeeves method for constrained global optimization," *Journal of Industrial and Management Optimization*, vol. 10, no. 4, pp. 1279–1296, 2014.

[14] Q. Long, C. Z. Wu, T. W. Huang, and X. Y. Wang, "A genetic algorithm for unconstrained multi-objective optimization," *Swarm and Evolutionary Computation*, 2015.

[15] D. F. Jones, S. K. Mirrazavi, and M. Tamiz, "Multi-objective meta-heuristics: an overview of the current state-of-the-art," *European Journal of Operational Research*, vol. 137, no. 1, pp. 1–9, 2002.

[16] P. Kumar, D. Gospodaric, and P. Bauer, "Improved genetic algorithm inspired by biological evolution," *Soft Computing*, vol. 11, no. 10, pp. 923–941, 2007.

[17] K. Sindhya, S. Ruuska, T. Haanpää, and K. Miettinen, "A new hybrid mutation operator for multiobjective optimization with differential evolution," *Soft Computing*, vol. 15, no. 10, pp. 2041–2055, 2011.

[18] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," in *Proceedings of the 1st international Conference on Genetic Algorithms*, pp. 93–100, L. Erlbaum Associates, 1985.

[19] C. M. Fonseca and P. J. Fleming, "Multiobjective genetic algorithms," in *Proceedings of the IEE Colloquium on Genetic Algorithms for Control Systems Engineering*, pp. 6/1–6/5, IET, London, UK, May 1993.

[20] J. Horn, N. Nafpliotis, and D. E. Goldberg, "A niched Pareto genetic algorithm for multiobjective optimization," in *Proceedings of the 1st IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, vol. 1, pp. 82–87, IEEE, Orlando, Fla, USA, June 1994.

[21] P. Hajela and C.-Y. Lin, "Genetic search strategies in multicriterion optimal design," *Structural Optimization*, vol. 4, no. 2, pp. 99–107, 1992.

[22] T. Murata and H. Ishibuchi, "MOGA: multi-objective genetic algorithms," in *Proceedings of the IEEE International Conference on Evolutionary Computation*, vol. 1, pp. 289–294, IEEE, December 1995.

[23] N. Srinivas and K. Deb, "Muiltiobjective optimization using nondominated sorting in genetic algorithms," *Evolutionary Computation*, vol. 2, no. 3, pp. 221–248, 1994.

[24] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[25] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: improving the strength Pareto evolutionary algorithm," Tech. Rep., Eidgenössische Technische Hochschule Zurich (ETH) Institut fur Technische Informatik und Kommunikationsnetze (TIK), 2001.

[26] J. D. Knowles and D. W. Corne, "Approximating the nondominated front using the pareto archived evolution strategy," *Evolutionary computation*, vol. 8, no. 2, pp. 149–172, 2000.

[27] D. W. Corne, J. D. Knowles, and M. J. Oates, "The Pareto envelope-based selection algorithm for multiobjective optimization," in *Parallel Problem Solving from Nature PPSN VI*, vol. 1917 of *Lecture Notes in Computer Science*, pp. 839–848, Springer, Berlin, Germany, 2000.

[28] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: empirical results," *Evolutionary computation*, vol. 8, no. 2, pp. 173–195, 2000.

[29] T. C. Koopmans and S. Reiter, "A model of transportation," in *Activity Analysis of Production and Allocation*, p. 222, John Wiley & Sons, New York, NY, USA, 1951.

[30] J. H. Holland, *Adaptation in Natural and Artificial Systems*, vol. 53, University of Michigan Press, Ann Arbor, Mich, USA, 1975.

[31] D. E. Goldberg, "A note on Boltzmann tournament selection for genetic algorithms and population-oriented simulated annealing," *Complex Systems*, vol. 4, no. 4, pp. 445–460, 1990.

[32] D. E. Goldberg, B. Korb, and K. Deb, "Messy genetic algorithms: motivation, analysis, and first results," *Complex Systems*, vol. 3, no. 5, pp. 493–530, 1989.

[33] J. J. Grefenstette, "Optimization of control parameters for genetic algorithms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 16, no. 1, pp. 122–128, 1986.

[34] H. Kitano, "Neurogenetic learning: an integrated method of designing and training neural networks using genetic algorithms," *Physica D: Nonlinear Phenomena*, vol. 75, no. 1–3, pp. 225–238, 1994.

[35] T. Murata, H. Ishibuchi, and H. Tanaka, "Multi-objective genetic algorithm and its applications to flowshop scheduling," *Computers and Industrial Engineering*, vol. 30, no. 4, pp. 957–968, 1996.

[36] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, Mass, USA, 1989.

[37] C. M. Fonseca and P. J. Fleming, "Genetic algorithms for multiobjective optimization: formulation discussion and generalization," in *Proceedings of the 5th International Conference on Genetic Algorithms (ICGA '93)*, pp. 416–423, 1993.

[38] J. Knowles and D. Corne, "The pareto archived evolution strategy: a new baseline algorithm for Pareto multiobjective optimisation," in *Proceedings of the Congress on Evolutionary Computation (CEC '99)*, vol. 1, pp. 98–105, July 1999.

[39] D. E. Goldberg and J. Richardson, "Genetic algorithms with sharing for multimodal function optimization," in *Proceedings of the 2nd International Conference on Genetic Algorithms and Their Applications*, pp. 41–49, Lawrence Erlbaum Associates, 1987.

[40] H. Lu and G. G. Yen, "Rank-density-based multiobjective genetic algorithm and benchmark test function study," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 4, pp. 325–343, 2003.

[41] G. G. Yen and H. Lu, "Dynamic multiobjective evolutionary algorithm: adaptive cell-based rank and density estimation," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 3, pp. 253–274, 2003.

[42] Q. F. Zhang, A. Zhou, S. Z. Zhao, P. N. Suganthan, W. D. Liu, and S. Tiwari, "Multiobjective optimization test instances for the CEC 2009 special session and competition," Tech. Rep. CES-487, University of Essex and Nanyang Technological University, 2009.

[43] Q. F. Zhang, W. D. Liu, and H. Li, "The performance of a new version of MOEA/D on CEC09 unconstrained MOP test instances," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '09)*, pp. 203–208, IEEE, May 2009.

[44] S. Kukkonen and J. Lampinen, "Performance assessment of Generalized Differential Evolution 3 (GDE3) with a given set of problems," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '07)*, pp. 3593–3600, IEEE, September 2007.

[45] C.-M. Chen, Y.-P. Chen, and Q. F. Zhang, "Enhancing MOEA/D with guided mutation and priority update for multi-objective optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '09)*, pp. 209–216, May 2009.

[46] L.-Y. Tseng and C. Chen, "Multiple trajectory search for unconstrained/constrained multi-objective optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '09)*, pp. 1951–1958, IEEE, May 2009.

[47] H.-L. Liu and X. Q. Li, "The multiobjective evolutionary algorithm based on determined weight and sub-regional search," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '09)*, pp. 1928–1934, IEEE, May 2009.

[48] M. H. Liu, X. F. Zou, C. Yu, and Z. J. Wu, "Performance assessment of DMOEA-DD with CEC 2009 MOEA competition test instances," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '09)*, pp. 2913–2918, IEEE, Trondheim, Norway, May 2009.

[49] K. Sindhya, A. Sinha, K. Deb, and K. Miettinen, "Optimization algorithm for constrained and unconstrained problems," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '09)*, pp. 2919–2926, May 2009.

[50] V. L. Huang, S. Z. Zhao, R. Mallipeddi, and P. N. Suganthan, "Multi-objective optimization using self-adaptive differential evolution algorithm," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '09)*, pp. 190–194, May 2009.

[51] Y. P. Wang, C. Y. Dang, H. C. Li, L. X. Han, and J. X. Wei, "A clustering multi-objective evolutionary algorithm based on orthogonal and uniform design," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '09)*, pp. 2927–2933, IEEE, Trondheim, Norway, May 2009.

[52] S. Tiwari, G. Fadel, P. Koch, and K. Deb, "Performance assessment of the hybrid archive-based micro genetic algorithm (AMGA) on the CEC09 test problems," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '09)*, pp. 1935–1942, IEEE, Trondheim, Norway, May 2009.

[53] B. Y. Qu and P. N. Suganthan, “Multi-objective evolutionary programming without non-domination sorting is up to twenty times faster,” in *Proceedings of the 11th Conference on Congress on Evolutionary Computation (CEC ’09)*, pp. 2934–2939, IEEE, 2009.

[54] A. Zamuda, J. Brest, B. Bošković, and V. Žumer, “Differential evolution with self-adaptation and local search for Constrained multiobjective optimization,” in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC ’09)*, pp. 195–202, May 2009.

[55] S. Gao, S. Y. Zeng, B. Xiao et al., “An orthogonal multi-objective evolutionary algorithm with lower-dimensional crossover,” in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC ’09)*, pp. 1959–1964, IEEE, May 2009.

[56] Q. F. Zhang and P. N. Suganthan, “Final report on CEC’09 MOEA competition,” in *IEEE Congress on Evolutionary Computation (CEC ’09)*, Piscataway, NJ, USA, 2009.