Combination of Evolutionary Algorithms with Decomposition Techniques for Multiobjective Optimization

A dissertation presented
by

Hui Li

to
The Department of Computer Science
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy
in the subject of

Computer Science

University of Essex
Colchester, United Kingdom
August 2007

Acknowledgments

First of all, I would like express my gratitude to my supervisor Dr. Qingfu Zhang, who provided me the constant support and guidance throughout my doctoral studies at the University of Essex. He introduced me to the field of evolutionary computation and multiobjective optimization.

Also, I would like to thank Dr. Abdellah Salhi and Dr. Yaochu Jin for the time they took for examining this thesis. Their constructive comments are quite helpful for improving the quality of this thesis and my future research work. I want to thank José Antonio and J Parkes Andrew for helping me revise the manuscript of this thesis.

I am grateful to Professor Edward Tsang and Dr. John Ford, who helped me with my first publication. I also wish to say thank you to the administrative and technical staffs in the department of computer science for their excellent service. Many thanks go to Mr. Aimin Zhou for his help providing the source code of his algorithm.

Finally, I want to truly acknowledge the endless support and love of my parents and wife.

Dedicated to my parents and my wife

Abstract

Multiobjective optimization problems (MOPs) arise in many real-life applications. In recent years, multiobjective evolutionary algorithms (MOEAs) have attracted a growing attention since they are able to find multiple compromise solutions (known as Pareto-optimal solutions) in a single run. Decomposition (i.e., the use of multiple aggregation functions) is one of the basic strategies for fitness assignment in MOEAs. However, the advantages of decomposition techniques have not been fully utilized in MOEAs. This thesis mainly studies the combination of evolutionary algorithms with decomposition techniques for multiobjective optimization. The major contributions of this thesis are as follows.

- A simple yet efficient multiobjective evolutionary algorithm based on decomposition, called MOEA/D, is proposed. It decomposes a MOP into a number of scalar subproblems. These subproblems are solved simultaneously and collaboratively by evolving a population of solutions. The main advantages of MOEA/D are: 1) low computational complexity in fitness assignment, 2) adaptive diversity maintenance, and 3) easy to incorporate single objective optimization methods.

- The investigation on the performance of MOEA/D with different decomposition techniques is presented in this thesis. A new decomposition method, called penalty-based boundary intersection approach (PBI), is proposed. We demonstrate that the performance of MOEA/D can be improved by using PBI.

- The influence of the mating restriction on the performance of MOEAs is investigated. In MOEA/D, the current solutions of neighboring subproblems, which are close in the decision space, have more chance to mate. We experimentally demonstrate that the mating restriction does improve the performance of MOEAs.

- The combinations of MOEA/D with local heuristics and differential evolution are also developed for solving discrete and continuous MOPs. Our experimental results show that MOEA/D with these optimization methods outperforms other state-of-the-art MOEAs.

- A set of continuous multiobjective test problems with nontrivial Pareto sets are suggested in this thesis. We demonstrate that these test problems are more difficult to solve and therefore more challenging for the current MOEAs.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter, some basic definitions and concepts in multiobjective optimization are first introduced. A brief overview of MOEAs is then given. The major issues in MOEAs are also discussed. Afterwards, the motivations and structure of this thesis are presented.

1.1 Multiobjective Optimization Problem

Many optimization problems in science, engineering, economics, finance and business [20][8][87][88], have more than one objective to be optimized. For example, in portfolio optimization, the investors need to consider both the minimization of risk and the maximization of profits. Very often, these objectives conflict with each other or are incommensurable. Generally, there doesn't exist one single solution in the search space optimizing all objectives simultaneously. The optimal solutions of multiobjective optimization problem (MOP) are the best tradeoffs among the objectives, known as Pareto-optimal solutions.

Mathematically, a MOP can be formulated as:

$$\begin{aligned} \text{minimize} \quad & F(x) = (f_1(x), \dots, f_m(x))^T \\ \text{subject to} \quad & x \in \Omega \subset X \end{aligned} \tag{1.1}$$

where the decision variable x belongs to the decision (variable) space X, $F : \Omega \to \mathbb{R}^m$ consists of m real-valued objective functions. The objective space is $\mathbb{R}^m$ and the feasible region is Ω. The multiobjective optimization problems (MOPs) can be classified as continuous or discrete according to the search space.

The variables $x \in X$ of continuous MOPs are real-valued. The feasible region Ω of continuous MOPs could be specified by the following equality and inequality constraints.

$$\Omega = \{x \in X \subset \mathbb{R}^n | g_i(x) \leq 0 \text{ and } h_j(x) = 0, i = 1, \dots, p; j = 1, \dots, q; \} \tag{1.2}$$

where p and q are the numbers of equality and inequality constraints respectively. The attainable objective set (AOS) can be defined as $\{F(x)|x \in \Omega\}$.

In contrast, the variables of discrete MOPs only take integer values. The search space of discrete MOPs is finite or countably infinite. Note that some MOPs contain both continuous and integer variables. Fuzzy and dynamic MOPs have also been studied by some researchers [78][21].

The optimal solutions of a MOP can be defined in terms of Pareto optimality. In the following, some basic concepts and definitions in multiobjective optimization are described[1].

Definition 1. For any two objective vectors $u = (u_1, \dots, u_m)^T$ and $v = (v_1, \dots, v_m)^T$, v is said to *dominate* u, denoted by $v \prec u$, if and only if $v_i \leq u_i$ for every $i \in \{1, 2, \dots, m\}$ and $v_j < u_j$ for at least one index $j \in \{1, 2, \dots, m\}$. u is *nondominated* to v if v doesn't dominate u.

Definition 2. A solution $x^* \in \Omega$ is said to be *(globally) Pareto-optimal* if there does not exist another $x \in \Omega$ such that $F(x)$ dominates $F(x^*)$. $F(x^*)$ is called the *Pareto objective vector*. The set of all Pareto-optimal objective vectors is called the *Pareto-optimal front*,

[1] The Pareto optimality for the MOPs with maximization objectives can be defined similarly.

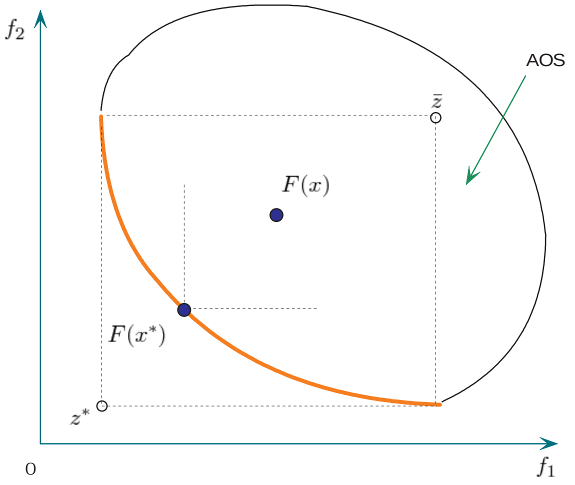


Figure 1.1: The globally Pareto-optimal solutions in the objective space.

denoted by PF. The set of all Pareto-optimal solutions in the decision space is called the *Pareto-optimal set*, denoted by PS.

In Figure 1.1, all Pareto-optimal objective vectors are in the lower-left boundary of the AOS, which is marked by a fat line. A continuous MOP usually has infinite number of Pareto-optimal solutions.

Definition 3. A decision vector $x^* \in \Omega$ is said to be *weakly Pareto-optimal* if there does not exist another decision vector $x \in \Omega$ such that $f_i(x) < f_i(x^*)$ for all $i = 1, \dots, m$.

In Figure 1.2, all objective vectors in the bold line are weakly Pareto-optimal. In fact, the Pareto-optimal set is the subset of weakly Pareto-optimal set. That is, a Pareto-optimal solution is also weakly Pareto-optimal. But the inverse is not necessarily true.

Definition 4. A solution $x^* \in \Omega$ is said to be *locally Pareto-optimal* if there exists $r > 0$ such that $F(x^*)$ is nondominated in $\Omega \cap B(x^*, r)$ where $B(x^*, r)$ is an open ball centered at x^*.

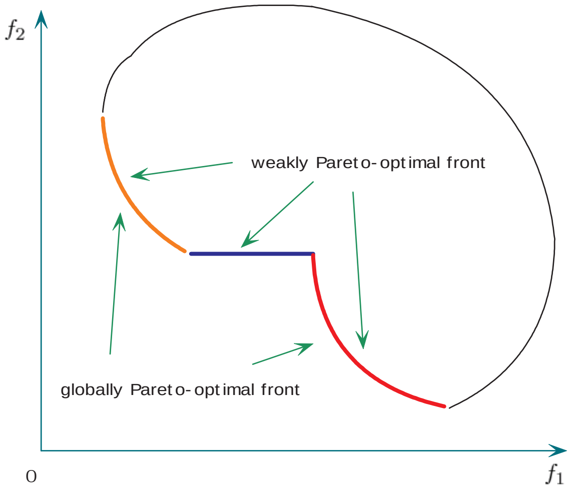


Figure 1.2: The weakly Pareto-optimal solutions in the objective space.

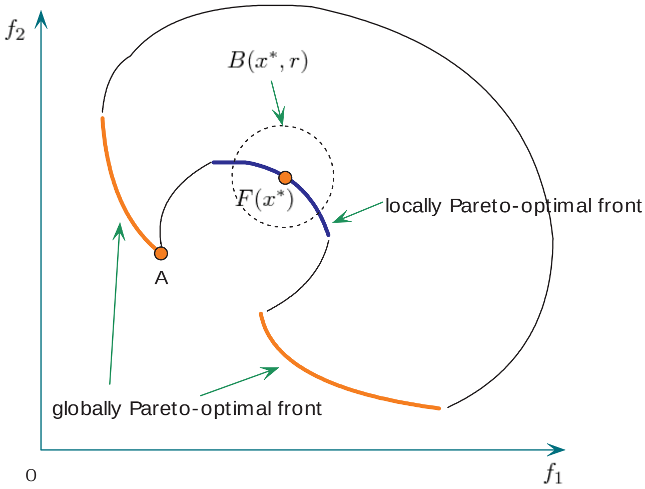


Figure 1.3: The locally Pareto-optimal solutions in the objective space.

The local Pareto-optimal solutions are illustrated in Figure 1.3. It can be seen from this figure that $F(x^*)$ is not dominated by any objective vectors in $B(x^*, r)$. But it is dominated by the global Pareto-optimal solution A. Naturally, global Pareto-optimal solutions are also local Pareto-optimal.

Definition 5. The objective vector z^* is called the ideal point in the objective space if $z_i^* = \min_{x \in \Omega} f_i(x), i = 1, \dots, m$. Correspondingly, the nadir point is defined as an objective vector $\bar{z}$ with $\bar{z}_i = \max\{f_i(x), x \in PS\}, i = 1, \dots, m$.

Both the ideal z^* point and the nadir point $\bar{z}$ are plotted in Figure 1.1. Generally, the ideal point z^* is an infeasible point in the objective space due to the possible conflicting objectives.

Definition 6. [66] A function $f_i : \mathbb{R}^n \to \mathbb{R}$ is *convex* if for all x, $y \in \mathbb{R}^n$, $f_i(\beta x + (1-\beta)y) \leq \beta f_i(x) + (1-\beta) f_i(y)$ for all $0 \leq \beta \leq 1$. A set $\Omega \subset \mathbb{R}^n$ is *convex* if $x, y \in \Omega$ implies $\beta x + (1-\beta)y \in \Omega$ for all $0 \leq \beta \leq 1$. The MOP (1.1) is *convex* if all the objective functions and feasible region are convex.

In addition to Pareto optimality, several other terms (such as noninferiority, efficiency and nondominance) are sometimes used for the optimality concepts described above. In the sense of Pareto optimality, every Pareto-optimal objective vector is an equally acceptable solution. In decision-making, the decision maker needs to select one Pareto-optimal solution which meets his/her preference.

1.2 Brief Overview of MOEAs

1.2.1 Evolutionary Algorithms

Inspired by biological evolution, evolutionary algorithms (EAs) have been developed since the 1970s. Representative EAs include genetic algorithms [29][27], genetic programming [56], evolution strategy [2], particle swarm optimization [5], scatter search [10], differential evolution [76], estimation of distribution algorithm [75][95][97], and others.

EAs operate on a population of candidate solutions (individuals) and produce better approximation to the optimal solution by applying the principle of ”survival of the fittest”. At each generation, a number of child solutions are generated by breeding the selected mating parents with good fitness. Then, the solutions with better fitness among the child and parent solutions survive at the next generation by performing selection.

Recombination, mutation and selection are three main operators in EAs. The first two operators are used to create offspring solutions. Selection operators determine the mating parents and the individuals retained in the next population. Examples of selection operators are roulette-wheel selection, truncation selection and tournament selection.

In the past twenty years, multiobjective evolutionary algorithms (MOEAs) have attracted a lot of attention [6][15][9][87][24]. The main goal of MOEAs is to find a set of nondominated solutions which approximate the PF as closely as possible (convergence) and cover the PF as widely and uniformly as possible (diversity). Compared with traditional multiobjective methods, MOEAs have the following advantages:

- MOEAs are able to find multiple nondominated solutions in a single run and adaptively maintain the diversity of these solutions;

- The performance of MOEAs is less susceptible to the features of PFs, such as nonconvexity and disconnectedness;

- For continuous MOPs, the differentiability of the objective functions and constraints are not necessarily required in MOEAs.

1.2.2 Major Issues in MOEAs

To find a good approximation of PFs, the issues including fitness assignment, diversity maintenance, elitism and reproduction operators, have been addressed in MOEAs. In the following, we briefly introduce these issues.

Fitness Assignment

To guide the search into the promising area in the decision space, MOEAs needs a fitness function to rank the individuals in population. However, defining such a fitness function is nontrivial since Pareto dominance is not a complete order. The general principle of fitness assignment to assign better fitness values to those individuals that are closer to the PF and locate in the less crowded region in the objective space. Pareto-based ranking and decomposition are two basic strategies for fitness assignment in MOEAs.

David E. Goldberg first suggested the use of Pareto dominance to rank the population in his seminal book on genetic algorithm [29]. In Pareto-based ranking, each individual is assigned a scalar value by comparing it with other individuals in the population on the basis of Pareto dominance. Generally, all nondominated solutions in the population have the same best rank value. Therefore, these solutions have higher probability to become the mating parents and survive in the next generation. Following Goldberg's idea, many Pareto-based MOEAs have been developed [15]. Of them, the representatives include MOGA [22], NPGA [32], NSGA [82], SPEA [106], PAES [53], SPEA-II [105] and NSGA-II [16].

In decomposition-based fitness assignment, all individuals in the population can be compared and ranked regarding a weighted scalar function, which is obtained by aggregating objectives linearly or nonlinearly. Unlike Pareto-based ranking, the fitness value of each individual obtained by decomposition is independent of other individuals in the population. Since the optimal solution of the weighted scalar function is also Pareto-optimal to the original MOP, the optimization of the weighted scalar function should be able to guide the search towards the PF. One weighted scalar function usually corresponds to one weight vector, which represents a certain search direction towards the PF. To approximate the whole PF, one needs to change the weight vector of weighted scalar function during the search. The representative MOEAs based on decomposition contain IMMOGLS [38], MOGLS [44], UGA [58], cMOGA [67] and MOSPS [34].

Recently, indicator-based selection [104] and ϵ-dominance [57] have also been studied by some researchers for fitness assignment.

Diversity Maintenance

There are two major purposes for diversity maintenance in MOEAs. On the one hand, like in single EAs, maintaining the diversity of population can prevent the search from getting trapped in local nondominated fronts; on the other hand, maintaining the diversity of nondominated solutions can provide a good approximation to the PF. The popular approaches for diversity maintenance include niche sharing [22][82][32], clustering [106], crowding density estimation [16], nearest neighbor method [105] and grid-based density estimation [52][61].

Elitism

The basic idea of elitism is to retain the nondominated solutions found during the search in the current population or an external population. The use of elitism in MOEAs not only avoids the loss of nondominated solutions previously found but also speeds up the convergence of population. Second population (also called external population) [102] and $(\mu + \lambda)$-selection mechanism [16] are two basic strategies for incorporating elitism into MOEAs. Two issues are often addressed in MOEAs regarding the use of a second population: 1) maintenance of the diversity of second population, and 2) interaction between second population and current population.

Solution Sampling

Generally, reproduction operators designed for scalar EAs can also be used in MOEAs if the search space is same. For continuous optimization problems, many real-valued reproduction operators, such as linear crossover, simulated binary crossover (SBX) [17], unimodal normally distributed crossover (UNDX) [72], simplex crossover (SPX) [31], have been widely used both in single objective and multiobjective EAs [15]. When dealing with the continuous MOPs with variable linkages, some advanced sampling techniques, such as differential evolution approaches (DEs) [10][62][55] and estimation of distribution algorithms (EDAs) [75] [60][71], are more effective. Very recently, reproduction operators based on problem-specific knowledge, such as the regularity property of MOPs, have also been considered in MOEAs [100]. In contrast, the reproduction operators for discrete MOPs are often problem-dependent.

1.2.3 MOEAs Using Pareto Dominance

Strength Pareto Evolutionary Approach II (SPEA-II) [105], Pareto Archive Evolution Strategy (PAES) [53], and Nondominated Sorting Genetic Algorithm II (NSGA-II) [16], are three well-known elitist-based MOEAs using Pareto dominance. In the following, we briefly introduce each of them.

SPEA-II proposed by Zitzler is the improved version of SPEA. In this algorithm, the fitness value of an individual takes into account both the number of individuals that it dominates and the number of individuals by which it is dominated by. A nearest neighborhood density estimation technique is suggested to evaluate the density of nondominated solutions. This technique is more efficient than the clustering technique in its predecessor. Besides, an enhanced archive truncation method is also employed to preserve the boundary solutions.

PAES introduced by Knowles is a multiobjective (1+1) ES. It maintains an external archive to store the nondominated solutions previously found. At each iteration, one parent generates one offspring. During the selection, the offspring wins the competition in the following two scenarios. 1) The offspring dominates the parent; 2) the offspring dominates the member in the archive and the parent doesn't when they are not dominated by each other. In all other scenarios, the offspring is discarded. If the offspring is not dominated by any member in the external archive, then add it into the external archive and remove those members dominated by it. To maintain the diversity of the external archive, a crowding procedure based adaptive grids is used to estimate the density of each member.

NSGA-II developed by Deb et. al is the improved version of NSGA [82]. Both versions employ the same nondominated sorting procedure for fitness assignment. In NSGA-II, a fast scheme to calculate the nondoimination rank is suggested. During the selection, the solutions with the better nondominated ranks are preferred. If two solutions have the same nondomination rank, the one resides in the less crowded region is preferred. The density of each individual is estimated by computing the average distance of two nearest individuals with the same rank on either side of this individual along each objective. Unlike the previous two Pareto-based MOEAs, NSGA-II considers the elitism by selecting the solutions with the best rank values from the union of parent and offspring solutions for the next generation (i.e. $(\mu+\lambda)$ selection). NSGA-II has become a landmark in the community of MOEAs due to its excellent performance for many MOPs.

1.2.4 MOEAs Based on Decomposition

In this section, some work on MOEAs based on decomposition is described in the following.

Ishibuchi and Murata suggested a multiobjective genetic local search, called IMMOGLS [38]. At each generation, the selection of mating parents is performed based on a weighted sum function with a random weight vector. IMMOGLS uses an external population to store all nondominated solutions previously found. The population at the next generation consists of both offspring solutions generated by genetic operators and part of elitist solutions from the external population. An improved version of IMMOGLS was further developed in [41], which took the normalization of objectives into consideration.

A cellular multiobjective genetic algorithm, called cMOGA, was developed based on IMMOGLS in [67]. In cMOGA, each individual is located in a cell associated with a weighted sum function. When generating a new individual for each cell, only the individuals in the neighboring cells are allowed to mate and selected using proportional selection regarding the related aggregation function. The new individual replaces the solution of the cell directly. At the end of each generation, individuals in some cells are replaced by some elitist solutions randomly chosen from an external elitist population.

Jaszkiewicz presented another version of multiobjective genetic local search [44], in which a temporary population of mating parents are selected by using the weighted sum approach or weighted Tchebycheff approach in fitness assignment at each iteration. Unlike IMMOGLS, the size of current population in this algorithm can be increased during the search and the external population is only used for retaining the nondominated solutions found during the search.

Leung and Wang proposed a multiobjective genetic algorithm based on uniform design techniques called UGA [58]. A set of weighted sum functions with fixed and uniform weight vectors are used for fitness assignment. To keep the diversity of population, uniform design techniques are used for generating weight vectors and initial population and designing crossover and mutation operators.

Jin et al. suggested an evolutionary dynamic weighted aggregation for multiobjective optimization called EDWA [45]. Unlike other MOEAs that use the weighted sum approach, its performance doesn't depend on the shape of PFs. When the weight vector in EDWA is dynamically changed, the current population should stably move towards the related optimal solution along the PFs. The nondominated solutions encountered during the search are stored in an elitist population.

Hughes developed a multiple single objective Pareto sampling approach, called MSOPS [34]. Instead of using one weighted scalarizing function at each generation, MSOPS uses multiple scalarizing functions with uniform weight vectors for fitness assignment. Each individual is associated with multiple ranks, which are obtained by sorting the population with respect to multiple scalarizing functions.

Compared with MOEAs using Pareto dominance, MOEAs based on decomposition seem more popular for solving multiobjective combinatorial optimization problems since local search and problem-specific heuristics can be easily applied in these algorithms [38][41]. A few other multiobjective metaheuristic algorithms based on decomposition can also be found in the literature. Some examples are PSA[12], MOSA [89] and MOTS[30].

1.2.5 Performance Assessment

When comparing the performance of MOEAs, one needs to measure the quality of a set of nondominated solutions rather than one single solution in single objective optimization. Two aspects are mainly addressed: the convergence and the diversity of nondominated solutions. On the one hand, the former describes the closeness between the obtained nondominated fronts and the true PF. On the other hand, the latter focuses on the distribution of the obtained nondominated solutions along the PF. Many performance metrics (also called performance indicators) have been suggested [15], such as error ratio, generational distance [91], set coverage [102], hypervolume [91], spacing [79], maximum spread [102]. As mentioned in [107][70], there doesn't exist a single metric that can reliably evaluate both the convergence and the diversity of nondominated solutions. This suggests the use of multiple metrics in the performance assessment of MOEAs.

1.3 Motivations and Outline of The Thesis

The main purpose of this thesis is to develop an efficient multiobjective evolutionary algorithm based on decomposition techniques called MOEA/D. It solves multiple single objective subproblems simultaneously and collaboratively by using an EA. Mating restriction is highly addressed in MOEA/D. The following are the major motivations of our work in this thesis.

- Decomposition is a basic strategy for multiobjective optimization in traditional mathematical programming [66]. If a MOP is decomposed into a number of scalar subproblems appropriately, the optimal solutions of these subproblems should be uniformly distributed along the PF [13]. However, this idea has not been widely used in MOEAs.
- Some subproblems obtained by decomposition techniques often share some similarity.

That is, the optimal solutions of these subproblems are close in the decision space. For this reason, the optimization of a certain subproblem can benefit from the information from its neighboring subproblems [13].

- In the community of MOEAs, mating restriction has been ignored by many researchers in a sense [90]. They believed that recombining the solutions on different portions of the PF without mating restriction can produce promising offspring solutions, which are very likely to be members of the PF. However, this may not be true when the Pareto-optimal solutions don't share too much similarity in the decision space [40].

- The methodologies for single objective optimization are rich. However, not many of them can be directly incorporated into MOEAs using Pareto dominance. But this is not a problem in MOEAs based on decomposition since only one single objective subproblem is optimized at each iteration or generation [43].

- The geometries of the PFs, which can cause difficulties for MOEAs, have been widely studied for constructing continuous multiobjective test problems [14]. In contrast, the work on constructing test problems based on PS geometries is still rare. Some evidences have shown that the MOPs with nontrivial PS geometries are more difficult to solve [69].

The structure of this thesis is organized as follows. Chapter 2 introduces two commonly-used weight-based decomposition techniques - weighted sum approach and weighted Tchebycheff approach. The Pareto-optimal conditions of these decomposition techniques are also described. Besides, the general boundary intersection approaches are reviewed and a new penalty-based boundary intersection approach is proposed. All decomposition techniques introduced in this chapter will be used and investigated in the following chapters.

Chapter 3 presents the general framework of MOEA/D, which provides the basis for the other versions of MOEA/D developed in the following three chapters. Then, the main issues in MOEA/D including neighboring subproblems, mating restriction, elitist strategies and problem-specific heuristics/local search, are explained. The similarity and difference between MOEA/D and cMOGA are discussed.

Chapter 4 first presents the version of MOEA/D with Tchebycheff approach. MOEA/D is then compared with NSGA-II and normal boundary intersection approach on a set of benchmark continuous MOPs. The investigation on MOEA/D with other decomposition techniques, such as weighed sum approach, objective normalization and penalty-based boundary intersection approach, are also presented. Finally, small population, sensitivity and scalability in MOEA/D are analyzed.

Chapter 5 describes the multiobjective knapsack problem and introduces two repair heuristics. Then, the combination of MOEA/D with local heuristics is presented. In this version of MOEA/D, an external population is maintained to store all nondominated solutions previously found. Afterwards, MOEA/D is compared with MOGLS on a set of nine test instances. The sensitivity of neighborhood size is also investigated.

In Chapter 6, a set of continuous multiobjective test problems with prescribed PSs are suggested. A theorem on the PF and PS of these new test problems is also provided. Then, a new version of MOEA/D, called DE-MOEA/D, is developed, where DE reproduction operators and two new diversity strategies are employed. DE-MOEA/D is compared with other two efficient MOEAs. Finally, the sensitivity of parameters and the influence of reproduction operators in DE-MOEA/D are investigated.

Chapter 7 summarizes the major conclusions of the research work in the whole thesis. Future research areas are also given.

Chapter 2

Decomposition Techniques for Multiobjective Optimization

In the community of mathematical programming, a lot of decomposition techniques have been developed [83][66]. This chapter first reviews the weight-based decomposition techniques for continuous MOPs[1]. The general boundary intersection methods are then introduced and a new penalty-based boundary intersection approach is proposed.

2.1 Weight-based Decomposition Techniques

In this section, two commonly-used weight-based decomposition techniques - weighted sum approach and weighted Tchebycheff approach are introduced.

2.1.1 Weighted Sum Approach

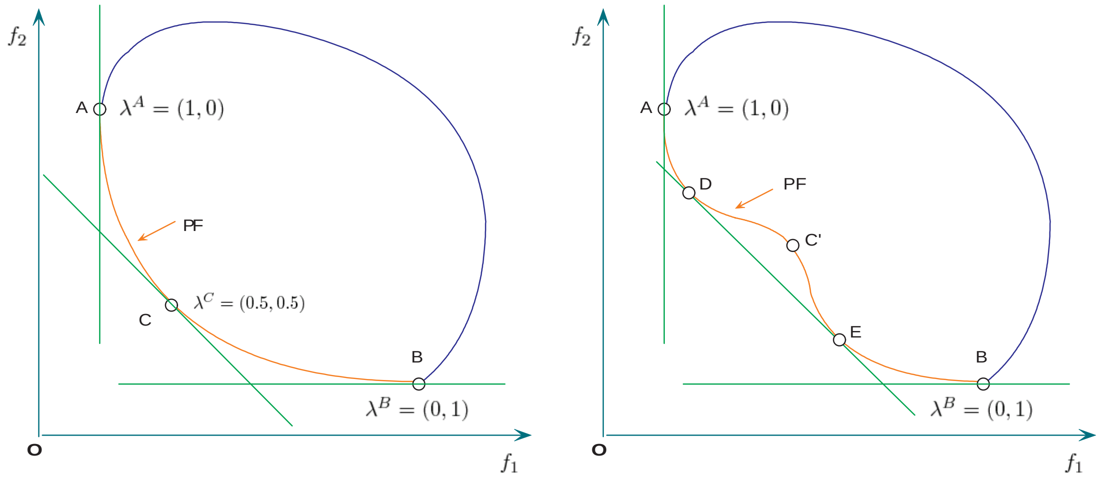
The weighed sum approach was presented in [26][94]. This approach considers a convex combination of the different objective functions. Each objective function f_i is

[1] These decomposition techniques can also be applied to solve discrete MOPs.

associated with a weight coefficient λ_i. The weighting scalar problem is of the form [66]:

$$\begin{aligned} &\text{minimize} \quad g^{ws}(x|\lambda) = \textstyle\sum_{i=1}^{m} \lambda_i f_i(x) \\ &\text{subject to} \qquad x \in \Omega \subset \mathbb{R}^n \end{aligned} \tag{2.1}$$

where $\lambda = (\lambda_1, \ldots, \lambda_m)^T$ is the weight vector with $\lambda_i \geq 0$ for all $i = 1, \ldots, m$ and $\sum_{i=1}^{m} \lambda_i = 1$.



(a) (b)

Figure 2.1: The weighted sum method for MOPs with (a) convex or (b) nonconvex PFs. Point C in the convex PF is the optimal solution of weighted sum function with weight vector $(0.5, 0.5)$ while point C' in the nonconvex PF is not the optimal solution for any weighted sum function.

Theorem 2.1 [66] The solution of weighting problem (2.1) is Pareto-optimal to the MOP (1.1) if the weighting coefficients are positive.

Theorem 2.2 [66] If $x^* \in \Omega$ is a unique solution of the weighting scalar optimization problem (2.1), then x^* is Pareto-optimal to the MOP (1.1).

Theorem 2.3 [66] Let the MOP (1.1) be convex, if $x^* \in \Omega$ is Pareto-optimal, then there exists a weighting vector λ with $\lambda_i \geq 0, i = 1, \ldots, m$, and $\sum_{i=1}^{m} \lambda_i = 1$ such that x^* is the optimal solution of the weighting scalar optimization problem (2.1).

Theorems 2.1 and 2.2 state the conditions for the Pareto optimality of the solutions of the weighting problem in (2.1). According to Theorem 2.3, any Pareto-optimal solution of a MOP can be found by the weighted sum approach if the MOP is convex. In this case, different Pareto-optimal solutions can be obtained by altering the weighting coefficients in the weighting problem. However, if the MOP is not convex, not all of the Pareto-optimal solutions can be found by the weighted sum approach. As we can see from Figure 2.1(a), all solutions in the convex PF can be obtained by the weighted sum approach if appropriate weight vectors are applied. As shown in Figure 2.1(b), it is impossible to find the Pareto-optimal solutions in the concave part of PF since they are not the optimal solutions of any weighting function $g^{ws}(x|\lambda)$.

Due to its simplicity, the weighted sum method has been widely used in many multiobjective metaheuristics [38][12][89][58][25]. To deal with the concave PFs, some effort has been made to incorporate other techniques such as ϵ-constraint or external population into this approach [66][45].

2.1.2 Weighted Tchebycheff Approach

The weighted Tchebycheff approach was originally proposed in [3]. In this approach, the weighted Tchebycheff problem has the aggregation function in the form:

$$\begin{array}{ll} \text{minimize} & g^{te}(x|\lambda, z^*) = \max_{i\in\{1,\ldots,m\}} \lambda_i |f_i(x) - z_i^*| \\ \text{subject to} & x \in \Omega \subset \mathbb{R}^n \end{array} \tag{2.2}$$

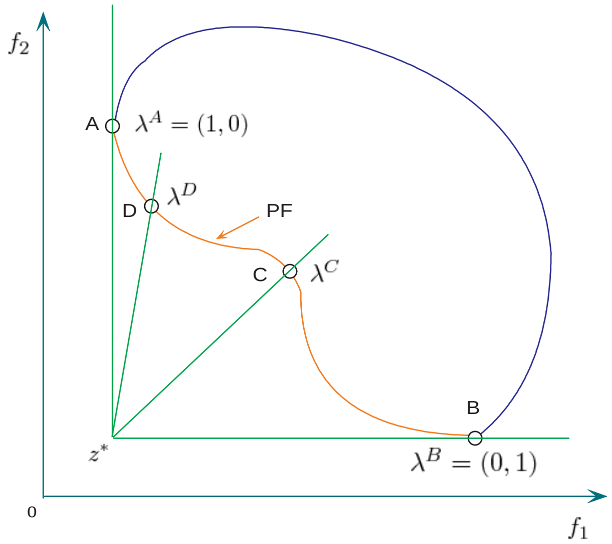


Figure 2.2: The weighted Tchebycheff approach for MOP with nonconvex PF. Point C and D are the optimal solutions of $g^{te}(x|\lambda^C, z^*)$ and $g^{te}(x|\lambda^D, z^*)$ respectively.

where $z^* = (z_1^*, \dots, z_m^*)^T$ is the reference point, i.e., $z_i^* = \min\{f_i(x)|x \in \Omega\}$ for each $i = 1, \dots, m$.

Theorem 2.4 [66] The optimal solution of the weighted Tchebycheff problem (2.2) is weakly Pareto-optimal, if all weighting coefficients are positive.

Theorem 2.5 [66] If the weighted Tchebycheff problem (2.2) has a unique solution, then it is Pareto-optimal.

Theorem 2.6 [83] Let x^* be weakly Pareto optimal, then there exists a weight vector λ with $\lambda_i \geq 0, i = 1, \dots, m$, and $\sum_{i=1}^{m} \lambda_i = 1$ such that x^* is the optimal solution of weighting scalar optimization problem (2.2).

According to Theorem 2.6, for each Pareto-optimal solution x^* of the MOP (1.1), there exists a weight vector λ such that x^* is the optimal solution of the weighted Tchebycheff problem (2.2) even if the MOP is not convex. As visualized in Figure 2.2, the Pareto-optimal solutions in the concave part of PF can also be obtained by weighted Tchebycheff approach.

In the weighted Tchebycheff approach, the ideal objective vector z^* is needed as reference point. If such a vector is unknown, then (weakly) Pareto-optimal solutions may not be found by the weighted Tchebycheff approach. In this case, an approximation of the ideal point is usually used.

Note that absolute sign is involved in this approach. For a continuous MOP, its aggregation function is not smooth. The traditional gradient-based methods are not applicable to such an aggregation function. Correspondingly, the differentiable form of the weighted Tchebycheff approach can be formulated as follows:

$$\begin{aligned} \text{minimize} \quad & g^{te2}(x|\lambda, z^*) = \alpha \\ \text{subject to} \quad & \alpha \geq \lambda_i(f_i(x) - z_i^*), \text{for all } i = 1, \dots, m, \\ & x \in \Omega \subset \mathbb{R}^n, \end{aligned} \tag{2.3}$$

where both x and $\alpha \in \mathbb{R}$ are variables.

Similarly to the weighted Tchebycheff approach, the achievement scalarizing function approach considers the following weighting problem [66]:

$$\begin{aligned} \text{minimize} \quad & g^{ach}(x|\lambda, z) = \max_{i=1,\dots,m}[\lambda_i(f_i(x) - z_i)] \\ \text{subject to} \quad & x \in \Omega \subset \mathbb{R}^n \end{aligned} \tag{2.4}$$

where $z = (z_1, \dots, z_m)^T$ is the reference point. Compared with the weighted Tchebycheff problem (2.2), the problem (2.4) has no absolute sign and the reference point z can be a feasible objective vector. This approach can obtain (weakly) Pareto-optimal solutions independently of the feasibility or infeasibility of the reference point.

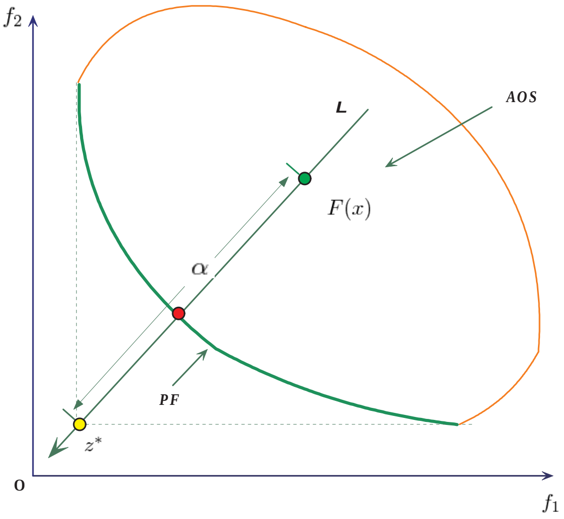


Figure 2.3: Illustration of general boundary intersection method

2.2 Boundary Intersection Methods

2.2.1 General Boundary Intersection Approach

Several recent MOP decomposition methods such as normal-boundary intersection method [13] and normalized normal constraint method [64] can be classified as boundary intersection (BI) approaches. They were designed for a continuous MOP. Under some regularity conditions, the PF of a continuous MOP is part of the lowest boundary of its attainable objective set (AOS). Geometrically, BI approaches aim to find intersection points of the lowest boundary and a set of lines. If these lines are evenly distributed in a sense, one can expect that the resultant intersection points provide a good approximation to the whole PF. These approaches are able to deal with nonconvex PFs. In our work, we use a set of lines emanating from the reference point. For a continuous MOP, we consider the following scalar optimization subproblem:

$$\begin{aligned} \text{minimize} \quad & g^{bi}(x|\lambda, z^*) && = \alpha \\ \text{subject to} \quad & F(x) - z^* && = \alpha\lambda, \\ & x \in \Omega \subset \mathbb{R}^n, \end{aligned} \tag{2.5}$$

where

- x and $\alpha \in \mathbb{R}$ are variables;
- λ and z^*, as in the above subsection, are a weight vector and a reference point, respectively.

As illustrated in Figure 2.3, the constraint $F(x) - z^* = \alpha\lambda$ ensures that $F(x)$ is always in L, the line with direction λ and passing through z^*. The goal is to push $F(x)$ as low as possible so that it reaches the boundary of the AOS.

NBI is a special case of BI methods. The weighting subproblems in NBI can also be specified according to (2.5). However, its direction vector λ is defined by the normal vector to the set $Q = \{\Phi\omega| \sum_{i=1}^{m} \omega_i = 1 \text{ and } \omega_i \geq 0, i = 1, \ldots, m\}$ pointing towards the origin, where $\Phi = [F(\tilde{x}^1), \ldots, F(\tilde{x}^m)]$ is a $m \times m$ matrix with $\tilde{x}^i = \min_{x \in \text{PS}} f_i(x), i = 1, \ldots, m$. The reference z^* is set to one of the points in Q instead of the ideal point. Given a fixed normal vector λ and a set of uniformly-distributed reference points, the optimal solutions of the resultant NBI subproblems should have a very good distribution along the PF.

2.2.2 Penalty-based Boundary Intersection Method

One of the drawbacks of the boundary intersection method described in the previous section is that it has to handle the equality constraint. To overcome this drawback, we propose a penalty method to deal with the constraint. More precisely, we consider:

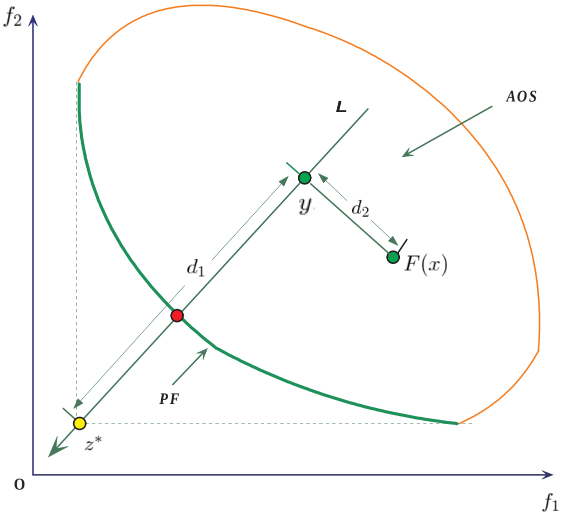


Figure 2.4: Illustration of the penalty-based intersection method.

$$\begin{aligned} \text{minimize} \quad & g^{pbi}(x|\lambda, z^*) = d_1 + c \cdot d_2 \\ \text{subject to} \quad & x \in \Omega \subset \mathbb{R}^n, \end{aligned} \tag{2.6}$$

where

- $d_1 = (F(x) - z^*)^T \lambda / ||\lambda\|$ and $d_2 = \|F(x) - (z^* + d_1\lambda)\|$;
- $c > 0$ is a preset penalty parameter.

Let y be the projection of $F(x)$ on the line L. As shown in Figure 2.4, d_1 is the distance between z^* and y. d_2 is the distance between $F(x)$ and L. If c is set appropriately, the solutions to (2.5) and (2.6) should be very close. Hereafter, we call this method the Penalty-based Boundary Intersection (PBI) approach.

The advantages of the PBI approach (or general BI approaches) over the Tchebycheff approach are:

- In the case of more than two objectives, let both the PBI approach and the Tchebycheff approach use the same set of evenly distributed weight vectors, the resultant optimal solutions in the PBI should be much more uniformly distributed than those obtained by the Tchebycheff approach [13][66], particularly when the number of weight vectors is not large.
- If x dominates y, it is still possible that $g^{te}(x|\lambda, z^*) = g^{te}(y|\lambda, z^*)$, while it is rare for g^{pbi} and other BI aggregation functions.

However, these benefits come with price. One has to set the value of the penalty factor. It is well-known that a too large or too small penalty factor will worsen the performance of a penalty method.

The above approaches can be used to decompose the approximation of the PF into a number of scalar optimization problems. A reasonably large number of evenly distributed weight vectors usually lead to a set of Pareto-optimal objective vectors, which may not be evenly spread but could approximate the PF very well.

2.3 Summary

This chapter first described two weight-based decomposition methods - the weighted sum approach and the weighted Tchebycheff approach. Compared with the weighted sum approach, the weighted Tchebycheff approach is less sensitive to the shape of PF. The general boundary intersection approach was then introduced and a penalty-based intersection approach was presented. If the weight vectors are set appropriately, boundary intersection methods should be able to produce a set of evenly-distributed Pareto-optimal solutions.

Chapter 3

MOEA/D: A Multiobjective Approach Based on Decomposition

3.1 Introduction

The majority of current state-of-the-art MOEAs are based on Pareto dominance. In these algorithms, the individuals in the population are ranked by performing comparison between them in terms of Pareto dominance. In this way, the convectional selection operators designed for scalar EAs can be used. To enhance the performance of these algorithms, some other techniques, such as mating restriction, diversity maintenance, the use of the properties of MOPs and external population, may also be needed. However, Pareto-based MOEAs have the following weaknesses:

- The selection pressure of these algorithms towards the PF will be decreased as the number of objectives increases [15]. If the number of objectives is large, many individuals in the initial population may have the same best Pareto-based rank. In this case, the Pareto-based selection cannot guide the search towards the PF;

- Single objective optimization methods and problem-specific heuristics cannot be easily used in MOEAs using Pareto dominance [43][42]. For example, it is not easy to define the acceptance function of local search methods guiding the search towards the PF efficiently in terms of Pareto dominance;
- The computational complexities of fitness assignment and diversity maintenance in many Pareto-based MOEAs are about $O(mN^2)$ [15], where N is the population size and m is the number of objectives. The efficiency of these algorithms will be decreased as the number of objectives and the population size increase.

As reviewed in the first Chapter, many MOEAs have used decomposition techniques for their fitness assignment. At each iteration or generation of these algorithms, one or multiple single objective subproblems obtained by using decomposition techniques are considered. Under mild conditions, the optimal solutions of these subproblems are still Pareto-optimal to the original MOP. Decomposition-based MOEAs aim at finding the optimal solutions of these subproblems. For this reason, the selection pressure in decomposition-based MOEAs is less sensitive to the number of objectives than that of MOEAs using Pareto dominance. Since each subproblem is associated with a scalar objective function, single objective optimization methods can be easily used in decomposition-based MOEAs. Another advantage of decomposition-based MOEAs is that the population can be forced to explore the desired regions of PFs if the single objective subproblems with appropriate scalar functions are optimized [43][42].

In this thesis, we propose a simple but efficient multiobjective objective evolutionary algorithm based on decomposition called MOEA/D. It decomposes a MOP N scalar subproblems and solves these subproblems simultaneously and collaboratively by evolving a population of solutions. MOEA/D has the following main features:

- MOEA/D provides a simple yet efficient way of introducing decomposition approaches into multiobjective evolutionary computation. A decomposition approach developed by the community of mathematical programming, can be readily incorporated into EAs in the framework of MOEA/D for solving MOPs.

- Since MOEA/D optimizes N scalar optimization problems rather than directly solving a MOP as a whole, issues such as fitness assignment and diversity maintenance that cause difficulties for nondecomposition MOEAs could become easier in the framework of MOEA/D.

- MOEA/D explicitly utilizes the neighboring relationships between subproblems in order to optimize multiple subproblems in an efficient way.

- It is very natural to use single objective optimization methods or problem-specific heuristics in MOEA/D since each solution is associated with a scalar optimization problem.

The rest of this chapter is organized as follows. The next section presents the general framework of MOEA/D. The main issues in MOEA/D are also discussed. In the following section, the similarity and difference between MOEA/D and cMOGA is also described. The final section summarizes the chapter.

3.2 MOEA/D

3.2.1 General Framework

The aim of multiobjective evolutionary algorithm based on decomposition, called MOEA/D, is to decompose the MOP under consideration. All decomposition techniques in traditional mathematical programming can serve this purpose. Assume that $g(x|\lambda^1), \dots, g(x|\lambda^N)$

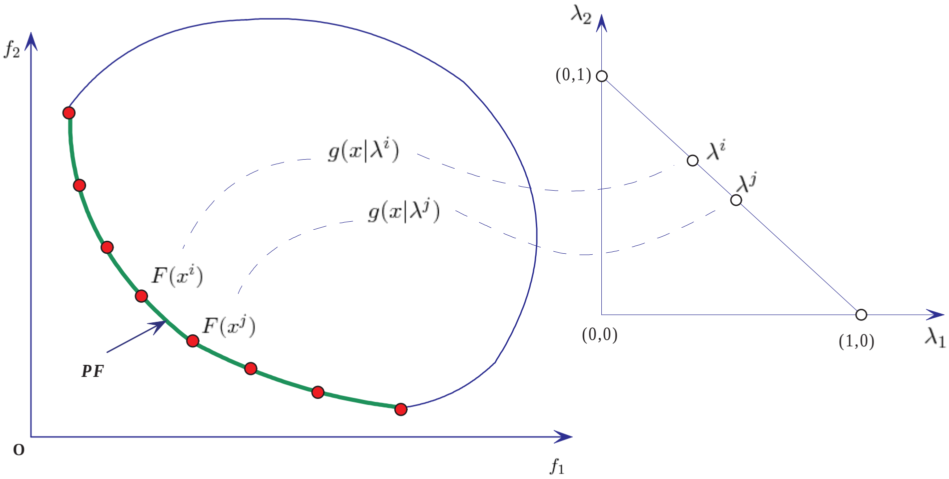


Figure 3.1: Neighboring relationship between subproblems and weight vectors

are N aggregation functions obtained by using a certain decomposition technique where $\lambda^1, \ldots, \lambda^N$ are N evenly-distributed weight vectors.

Generally, the aggregation function $g(x|\lambda)$ is continuous in λ, the optimal solution of $g(x|\lambda^i)$ should be close to that of $g(x|\lambda^j)$ if the weight vectors λ^i and λ^j are close to each other. Therefore, any information of these $g's$ with weight vectors close to λ^i should be helpful in optimizing $g(x|\lambda^i)$. This is the major motivation in MOEA/D. The neighboring relationships between subproblems and weight vectors are illustrated in Figure 3.1.

The general algorithmic framework of MOEA/D is presented in Figure 3.2. In MOEA/D, a population of N solutions $x^1, \ldots, x^N$ are maintained, where x^i is the best solution found so far for the ith subproblem. MOEA/D optimizes N subproblems simultaneously by evolving $x^1, \ldots, x^N$. In the rest of this chapter, the main issues in MOEA/D, such as neighboring subproblems, diversity maintenance, mating restriction, elitist strategies and problem-specific heuristics/local search, are introduced. More details of MOEA/D can be found later in the following three chapters.

Input: MOP (1.1); a stopping criterion; N : the number of the subproblems; $\lambda^1, \ldots, \lambda^N$: a set of N weight vectors; T : the number of weight vectors in the neighborhood of each weight vector.

Output: EP or $\{F(x^1), \ldots, F(x^N)\}$.

Step 1: Initialization

Step 1.1 Set EP$= \emptyset$.

Step 1.2 Compute the Euclidean distances between any two weight vectors and then work out the T closest weight vectors to each weight vector. For each $i = 1, \ldots, N$, set $B(i) = \{i_1, \ldots, i_T\}$ where $\lambda^{i_1}, \ldots, \lambda^{i_T}$ are the T closest weight vectors to λ^i;

Step 1.3 Generate the initial population $x^1, \ldots, x^N$ randomly or by a problem-specific heuristic or local search method;

Step 2: Reproduction and Update

For each subproblem $i \in \{1, \ldots, N\}$

Step 2.1 Select mating parents from $B(i)$ and generate an offspring y by using genetic operators;

Step 2.2 Apply a problem-specific heuristic or local search method on y to produce y';

Step 2.3 For each $j \in B(i)$, set $x^j = y'$ if x^j is worse than y' regarding $g(x|\lambda^j)$;

Step 2.4 Update EP.

Step 3: Termination If the stopping criteria is satisfied, then return EP or $\{F(x^1), \ldots, F(x^N)\}$; otherwise, go to **Reproduction and Update**.

Figure 3.2: The algorithmic framework of MOEA/D

3.2.2 Neighboring Subproblems

The objective of MOEA/D is to optimize N scalar subproblems simultaneously. MOEA/D spends about the same amount of effort on each of the N aggregation functions, while MOGLS randomly generates a weight vector at each iteration, aiming at optimizing all the possible aggregation functions [44]. Recall that a decision maker only needs a finite number of evenly distributed Pareto solutions, optimizing a finite set of selected scalar subproblems is not only realistic but also appropriate. Since the computational resource is always limited, optimizing all the possible aggregation functions would not be very practical, and thus may waste some computational effort.

In Step 1.2 of MOEA/D, a neighborhood of weight vector λ^i is defined as a set of T closest weight vectors in $\{\lambda^1, \ldots, \lambda^N\}$, i.e., $B(i) = \{i_1, \ldots, i_T\}$, where $\lambda^{i_1}, \ldots, \lambda^{i_T}$ are the T closest weight vectors to λ^i. The closeness between weight vectors is measured by using Euclidean distance of vectors in $\mathbb{R}^m$. Correspondingly, the neighborhood of the ith subproblem consists of all the subproblems with the weight vectors in the neighborhood of λ^i. The population is composed of the best solution found so far for each subproblem. In general, only the current solutions to its neighboring subproblems are exploited for optimizing subproblem in MOEA/D, that is, the selection of mating parents and the replacement of solutions in the current population are only performed among the solutions of neighboring subproblems.

3.2.3 Mating Restriction

In the EA community, mating restriction is used to reduce the recombination operations between dissimilar parents, which has low probability to produce good offspring solutions [29]. Mating restriction has also attracted some attention in MOEA community [22][40][102][90][39]. However, this technique has not been efficiently used in the majority of state-of-the-art MOEAs for solving continuous MOPs [90].

In Step 2.1 of MOEA/D, only current solutions to the T closest neighbors of a subproblem are used for optimizing it. This means that solutions have a chance to mate only if they are neighbors. Attention should be paid to the setting of T. If T is too small, the mating parents chosen for genetic operators may be very similar since they are for the similar subproblems. Consequently, the offspring produced could be very close to the mating parents. Therefore, the algorithm lacks the ability of exploring new areas in the search space. On the other hand, if T is too large, the mating parents may be poor for the subproblem under consideration, and so are their offspring. Therefore, the exploitation ability of the algorithm is weakened. Moreover, a too large T will increase the computational overhead.

3.2.4 Diversity Maintenance

A MOEA needs to maintain diversity in its population for producing a set of representative solutions. Most, if not all, of Pareto-based MOEAs such as NSGA-II and SPEA-II use crowding distance among the solutions in their selection to maintain diversity. However, it is not always easy to generate a uniform distribution Pareto-optimal objective vectors in these algorithms.

In MOEA/D, a MOP is decomposed into N scalar optimization subproblems. Different solutions are associated with different subproblems. The ”diversity” among these subproblems will naturally lead to diversity in the population. When the decomposition method and weight vectors are properly chosen, and thus the optimal solutions to the resultant subproblems are evenly distributed along the PF. MOEA/D will have a good chance of producing a uniform distribution of Pareto solutions if it optimizes all subproblems very well. Compared with crowding distance used in NSGA-II and SPEA-II, no extra computational overhead is needed in the diversity maintenance since it is adaptive.

3.2.5 Elitist Strategies

In MOEA/D, a second population EP is maintained and updated in Step 1.1 and Step 2.4. All nondominated solutions found during the search can be stored in the second population. One can limit the size of the second population to avoid any possible memory overflow problem since the number of Pareto solutions of MOPs could be infinite. Of course, some other sophisticated strategies [50] for updating can be easily adopted in the framework of MOEA/D.

In MOEA/D, using the external population is only an option. Alternatively, the final internal population can also be returned as the approximation of PF. If MOEA/D solves all scalar optimization subproblems well, the final internal population should consists of a set of evenly distributed Pareto solutions.

3.2.6 Problem-specific Heuristics/Local Search

The combination of EAs with problem-specific heuristics or local search, also called memetic algorithms or genetic local search algorithms, has proved to be very promising for solving hard optimization problems [63][86][54]. On the one hand, EAs can locate the promising search area by evolving a population of solutions (exploration); on the other hand, high-quality local optima in a promising area can be examined by using local search techniques (exploitation). Some multiobjective memetic algorithms have also been developed [38][43][48][51]. As shown in Figure 3.2, problem-specific heuristics and local search can be easily used in Step 1.3 and Step 2.2 of MOEA/D since each solution is associated with a subproblem.

3.3 Similarity and Difference Between MOEA/D and cMOGA

The cellar multiobjective genetic algorithm (cMOGA) of Murata et al. [67] can also be regarded as a variant of MOEA/D. The cells of cMOGA correspond to the subproblems of MOEA/D. In essence the same neighborhood relationship is defined by measuring the distances between weight vectors in both algorithm for mating restriction. That is, mating parents are only selected from the neighboring subproblems. cMOGA mainly differs from MOEA/D in two aspects - selecting solutions for genetic operators and updating the internal population. First, cMOGA chooses mating parents by performing proportional selection to the solutions of the neighboring subproblems while MOEA/D chooses mating parents from the neighborhood randomly (see step 2.1 in Figure 3.2). Second, there is no mechanism for keeping the best solutions found so far to each subproblem in cMOGA. The current solution of each subproblem is always replaced by the offspring solution generated for it without comparing their function values. To deal with the nonconvex PFs, cMOGA has to insert solutions from its external population to its internal population since it uses the weighted sum approach for its fitness assignment.

3.4 Summary

The general framework of MOEA/D was first presented in this chapter. Based on this framework, three variants of MOEA/D will be developed to solve benchmark continuous MOPs, multiobjective knapsack problem and continuous MOPs with nontrivial PS in the following three chapters. The major issues including neighboring subproblems, diversity maintenance, elitist strategies and problem-specific heuristics/local search were discussed. The similarity and difference between MOEA/D and cMOGA were explained.

Chapter 4

MOEA/D for Benchmark Continuous MOPs

4.1 Introduction

In multiobjective evolutionary computation, many different test problems with known sets of Pareto-optimal solutions have been used by researchers for testing the performance of MOEAs. In this chapter, we consider the most well-known and commonly-used benchmark continuous test problems - ZDT and DTLZ test problems for comparing the performance of MOEA/D with NSGA-II [16] and NBI [13] and investigating the suitability of decomposition methods in MOEA/D.

The ZDT test problems can be created by using Deb's tunable 2-objective toolkit [14], which can be formulated as:

$$\begin{aligned} \text{minimize} \quad & f_1(x) = f_1(x_I), \\ \text{minimize} \quad & f_2(x) = g(x_{II}) \times h(f_1(x_I), g(x_{II})). \end{aligned} \tag{4.1}$$

where

- $x = (x_1, \dots, x_n)^T \in \Omega \subset \mathbb{R}^n$, $x_I = (x_1, \dots, x_k)^T$ and $x_{II} = (x_{k+1}, \dots, x_n)^T$ are the subvectors of x;
- f_1 is the function of x_I, g is the function of x_{II}, and h is the function of f_1 and g.

If the functions f_1, h and g are appropriately chosen, some specific features of the MOP in (4.1), such as convexity, discontinuity, multimodality, deception and nonuniformatliy, can be generated. The function f_1 can affect the difficulty in diversifying the nondominated solutions along the PF (nonuniformaltiy) while the function g can affect the difficulties in converging to the PF (multimodality and deception). The function h determines the shape of the PF (convexity and discontinuity). Generally, the global Pareto-optimal set corresponds to the global minimum of the function g. In the ZDT test problems, the Pareto-optimal solutions satisfy $x_i = 0, i = 2, \dots, n$.

Deb et. al further suggested the DTLZ test problems in [19], which are scalable to the number of objectives. The first six DTLZ test problems (i.e., DTLZ1-DTLZ6) can be written as:

$$\begin{aligned} f_1(x) &= s_1(x_I) \times (1 + g(x_{II})) \\ &\dots \\ f_m(x) &= s_m(x_I) \times (1 + g(x_{II})) \end{aligned} \tag{4.2}$$

where

- $x = (x_1, \dots, x_n)^T \in \Omega \subset \mathbb{R}^n$, $x_I = (x_1, \dots, x_k)^T$ and $x_{II} = (x_{k+1}, \dots, x_n)^T$ are the subvectors of x;
- $s_i, i = 1, \dots, m$ is the function of x_I, g is the function of x_{II}.

The functions $s_i, i = 1, \dots, m$ determine the shape of the PF and the function g can cause difficulty in converging to the PF. The Pareto-optimal solutions of the DTLZ problems also correspond to the global minimum of the function g.

In this chapter, we develop a simple version of MOEA/D with the weighted Tchebycheff approach for continuous MOPs. Its main features include the following.

- The objective function of each subproblem is defined by the weighted Tchebycheff approach, which is capable of dealing with both convex and nonconvex PFs. A set of weight vectors needed in MOEA/D are uniformly-distributed;
- No external population is employed to store all nondominated solutions found during search. Instead, the final solutions of subproblems are returned;
- The conventional real-valued reproduction operators, such as simulated binary crossover (SBX) and polynomial mutation, are used to generate offspring solutions.

The rest of this chapter is organized as follows. Section 4.2 presents the proposed MOEA/D with Tchebycheff approach. The comparison of MOEA/D with NSGA-II is provided in Section 4.3. The performance of MOEA/D with objective normalization, the PBI approach and the weighted sum approach is studied in Section 4.4. The following section compares MOEA/D with the NBI approach. Section 4.6 investigates the small population, sensitivity and scalability in MOEA/D. The final section summarizes this chapter.

4.2 MOEA/D with Tchebycheff approach

In this section, a MOEA/D with the Tchebycheff approach is presented by following the general framework in Chapter 3. At each generation, MOEA/D maintains:

- a population of N points $x^1, x^2, \ldots, x^N$, where x^i is the current solution to the i^{th} subproblem;

```
Input  MOP (1.1); a stopping criterion; $N$ : the number of the subproblems; $\lambda^1, \ldots, \lambda^N$: a set of $N$ weight vectors; $T$ : the number of weight vectors in the neighborhood of each weight vector.
Output  Approximation to the PF: $F(x^1), \ldots, F(x^N)$
Step 1  Initialization
    Step 1.1 Compute the Euclidean distances between any two weight vectors and then work out the $T$ closest weight vectors to each weight vector. For each $i = 1, \ldots, N$, set $B(i) = \{i_1, \ldots, i_T\}$ where $\lambda^{i_1}, \ldots, \lambda^{i_T}$ are the $T$ closest weight vectors to $\lambda^i$.
    Step 1.2 Generate an initial population $x^1, \ldots, x^N$ by uniformly randomly sampling from $\Omega$. Set $FV^i = F(x^i)$.
    Step 1.3 Initialize $z = (z_1, \ldots, z_m)$ by setting $z_j = \min_{1 \leq i \leq N} f_j(x^i)$.
Step 2  Update
    For each $i \in \{1, \ldots, N\}$
    Step 2.1 Reproduction: Randomly select two distinct indexes $p$ and $q$ from $B(i)$ and generate a new solution $y$ from $x^p$ and $x^q$ by using genetic operators.
    Step 2.2 Update of $z$: For each $j = 1, \ldots, m$, if $z_j > f_j(y)$, then set $z_j = f_j(y)$.
    Step 2.3 Update of Neighboring Solutions: For each index $j \in B(i)$, if $g^{te}(y|\lambda^j, z) \leq g^{te}(x^j|\lambda^j, z)$, then set $x^j = y$ and $FV^j = F(y)$.
Step 3 Stopping Criteria  If the stopping criteria is satisfied, then stop and output $F(x^1), \ldots, F(x^N)$. Otherwise, go to Step 2.
```

Figure 4.1: MOEA/D with Tchebycheff approach

- $FV^1, \dots, FV^N$, where FV^i is the function values of x^i, i.e., $FV^i = F(x^i)$, for each $i = 1, \dots, N$;
- a vector $z = (z_1, \dots, z_m)^T$, where z_i is the best value found so far for objective f_i.

The details of MOEA/D with the Tchebycheff approach are given in Figure 4.1. In Step 1.1, the neighborhood of each subproblem is determined by calculating the Euclidean distances among weight vectors. Step 1.2 initializes the population randomly. In Step 1.3 and Step 2.2, we use any newly-generated solutions to update the reference point $z = (z_1, \dots, z_m)^T$. In Step 2.1, any commonly-used real-valued genetic operators in single objective EAs can be used here. In Step 2.3, the current solutions of neighboring subproblems are replaced by the offspring y if y is better regarding the related objective function. The final population is returned as an approximation to the PF in Step 3.

4.3 Comparison with NSGA-II

This section starts by the description of NSGA-II. The comparison of MOEA/D with NSGA-II on the space and computational complexity is then provided. Afterwards, the experimental results on the comparison of MOEA/D with NSGA-II on a set of benchmark continuous MOPs are presented.

4.3.1 NSGA-II

NSGA-II proposed in [16] is one of the most successful MOEAs using Pareto dominance. In this algorithm, the entire population is classified by using a nondominated sorting procedure. The basic idea is to find the nondominated solutions in the population. These solutions form the first nondominated front with the highest rank and are removed temporarily. Another set of nondominated solutions regarding the remaining population

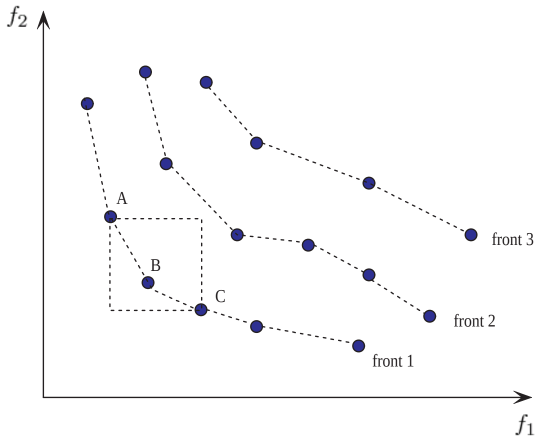


Figure 4.2: Nondomination ranking and crowding distance estimation

form the second nondominated front with the next highest rank. This process is repeated until all individuals in the population are ranked. Let us consider the population in Figure 4.2, all population members are classified into three distinct nondominated fronts. The computational complexity of this nondominated sorting is about $O(m \cdot N^2)$.

To maintain the diversity of the solutions in a certain nondominated front, NSGA-II employs crowding distance to estimate the density value of each individual in the objective space, which equals the average distance of two individuals on either side of this individual along each of the objectives. As shown in Figure 4.2, the crowding distance of individual B in the front "1" is the average side length of the rectangle around it. The calculation of the crowding distance values has $O(mNlogN)$ computational complexity.

The basic framework of NSGA-II is illustrated in Figure 4.3. As we can see, the major computational costs of NSGA-II are involved in Step 2.1 and Step 2.2.

```
Input   MOP (1.1); a stopping criterion; $N$: population size
Output  $F(x^1), \ldots, F(x^N)$
Step 1 Initialization Generate an initial population $P_0 = \{x^1, \ldots, x^N\}$ randomly and evaluate
    them. Set $t := 0$.
Step 2 Reproduction and Update
    2.1 Reproduction: Create an offspring population $Q_t$ with $N$ offspring solutions by using
    selection, crossover, and mutation from $P_t$;
    2.2 Update: Chose $N$ best solutions from $P_t \bigcup Q_t$ to form $P_{t+1}$ by using nondominated
    sorting and crowding density estimation.
Step 3 Stopping Criteria If the stopping criteria is satisfied, then stop and output
    $F(x^1), \ldots, F(x^N)$. Otherwise, $t := t + 1$ and go to Step 2.
```

Figure 4.3: The algorithmic framework of NSGA-II

4.3.2 Analysis of Space and Computational Complexities

In MOEA/D, the major computational costs are in Step 2. Step 1.2 in MOEA/D generates N trial solutions, so does NSGA-II at each generation. Step 2.1 just randomly picks two solutions for genetic operators, Step 2.2 performs $O(m)$ comparisons and assignments, and Step 2.3 needs $O(mT)$ basic operations since its major costs are to compute the values of g^{te} for T solutions and the computation of one of such values requires $O(m)$ basic operations. Therefore, the computational complexity of Step 2 in the above variant of MOEA/D is $O(mNT)$ since it has N passes. If both MOEA/D and NSGA-II use the same population size, the ratio between their computational complexities at each generation is

$$\frac{O(mNT)}{O(mN^2)} = \frac{O(T)}{O(N)}.$$

Since T is smaller than N, the variant of MOEA/D has lower computational complexity than NSGA-II at each generation.

4.3.3 Experimental Setting

We use five widely-used 2-objective ZDT test instances [103] and two 3-objective instances [19] in comparing MOEA/D with NSGA-II. The objectives of all these test instances are to be minimized (see the details of these test instances in **Appendix 1**).

In our experimental studies, the implementation of NSGA-II follows [16]. The population size N in both NSGA-II and MOEA/D is set to 100 for all the 2-objective test instances and 300 for two 3-objective test instances. Both algorithms stop after 250 generations. Both MOEA/D and NSGA-II were implemented in C++ and have been run 30 times independently for each test instance on identical computers (Pentium (R) 3.2 GHZ).

In both algorithms, initial populations are generated by uniformly randomly sampling from the feasible search space. z_i in MOEA/D is initialized as the lowest value of f_i found in the initial population. The simulated binary crossover (SBX) and polynomial mutation are used in both NSGA-II and MOEA/D. More precisely, in the Step 2.1 of MOEA/D, the crossover operator generates one offspring, which is then modified by the mutation operator. The setting of the control parameters in these two operators is the same in both algorithms. Following the practice in [16], the distribution indexes in both SBX and the polynomial mutation are set to be 20. The crossover rate is 1.00 while the mutation rate is $1/n$ where n is the number of decision variables.

In MOEA/D, T is set to 20 for all the test instances. The weight vectors $\lambda^1, \dots, \lambda^N$ in MOEA/D are controlled by a parameter H. More precisely, $\lambda^1, \dots, \lambda^N$ are all the weight vectors in which each individual weight takes a value from:

$$\{\frac{0}{H}, \frac{1}{H}, \dots, \frac{H}{H}\}.$$

Therefore, the total number of feasible weight vectors is: $N = C_{H+m-1}^{m-1}$. In our experiments, we use $H = 99$ for five 2-objective test instances ($N = C_{100}^{1} = 100$) and $H = 23$ for two

3-objective test instances ($N = C_{25}^2 = 300$).

4.3.4 Performance metrics

As mentioned in the first chapter, no metric can reliably evaluate the quality of solutions found by MOEAs both in convergence and in diversity. In this chapter, we consider the following two metrics.

- **Set Coverage (C-metric):** Let A and B be two approximations to the PF of a MOP, $C(A, B)$ is defined as the percentage of the solutions in B that are dominated by at least one solution in A, i.e.,

$$C(A, B) = \frac{|\{u \in B | \exists v \in A : v \text{ dominates } u\}|}{|B|}$$

  $C(A, B)$ is not necessarily equal to $1 - C(B, A)$. $C(A, B) = 1$ means that all solutions in B are dominated by some solutions in A, while $C(A, B) = 0$ implies that no solution in B is dominated by a solution in A.

- **Distance from Representatives in the PF (D-metric):** Let P^* be a set of uniformly distributed points along the PF. Let A be an approximation to the PF, the average distance from P^* to A is defined as:

$$D(A, P^*) = \frac{\sum_{v \in P^*} d(v, A)}{|P^*|}$$

  where $d(v, A)$ is the minimum Euclidean distance between v and the points in A. If $|P^*|$ is large enough to represent the PF very well, $D(A, P^*)$ could measure both the diversity and convergence of A in a sense. To have a low value of $D(A, P^*)$, the set A must be very close to the PF and cannot miss any part of the whole PF.

Table 4.1: Average CPU time (in second) used by NSGA-II and MOEA/D with the Tchebycheff approach.

Instance	NSGA-II	MOEA/D
ZDT1	1.03	0.60
ZDT2	1.00	0.47
ZDT3	1.03	0.57
ZDT4	0.77	0.33
ZDT6	0.73	0.27
DTLZ1	10.27	1.20
DTLZ2	8.37	1.10

4.3.5 Experimental Results

Table 4.1 presents the average CPU time used by each algorithm for each test instance. It is clear from Table 4.1 that, on average, MOEA/D runs about twice as fast as NSGA-II with the same number of function evaluations for the 2-objective test instances, and more than 8 times for the 3-objective test instances. This observation is consistent with our complexity analysis in Section 4.3.2.

Table 4.2: Average set coverage between MOEA/D with the Tchebycheff approach (A) and NSGA-II (B).

Instance	C(A,B)	C(B,A)
ZDT1	15.88	1.64
ZDT2	15.53	5.37
ZDT3	14.61	2.92
ZDT4	10.74	23.09
ZDT6	99.56	0
DTLZ1	7.84	0.49
DTLZ2	9.91	0

We use both the C-metric and D-metric to compare the performances of these two algorithms. To compute D-metric values, P^* is chosen to be a set of 500 uniformly

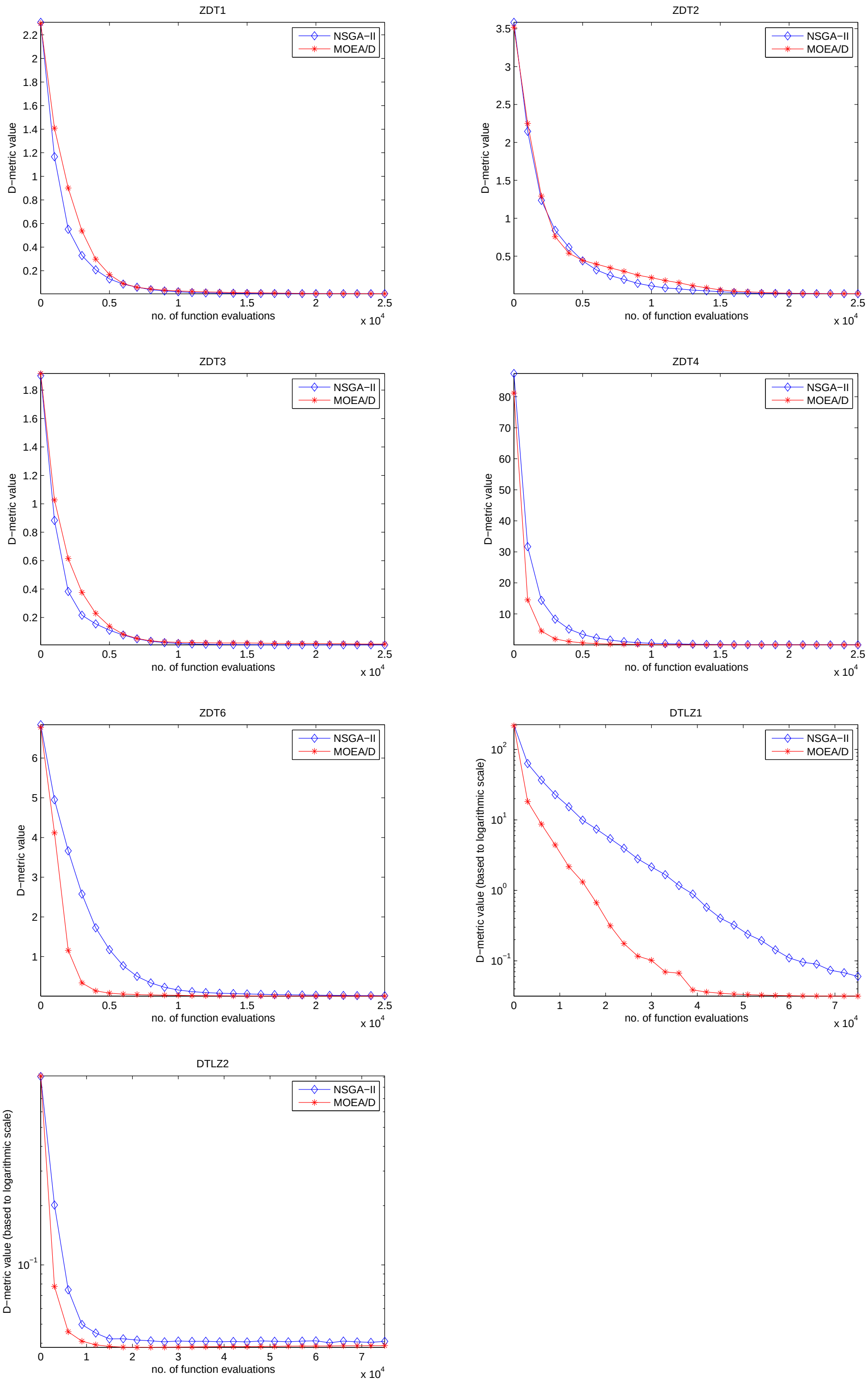


Figure 4.4: The evolution of D-metric values in both MOEA/D with the Tchebycheff approach and NSGA-II for each test instance.

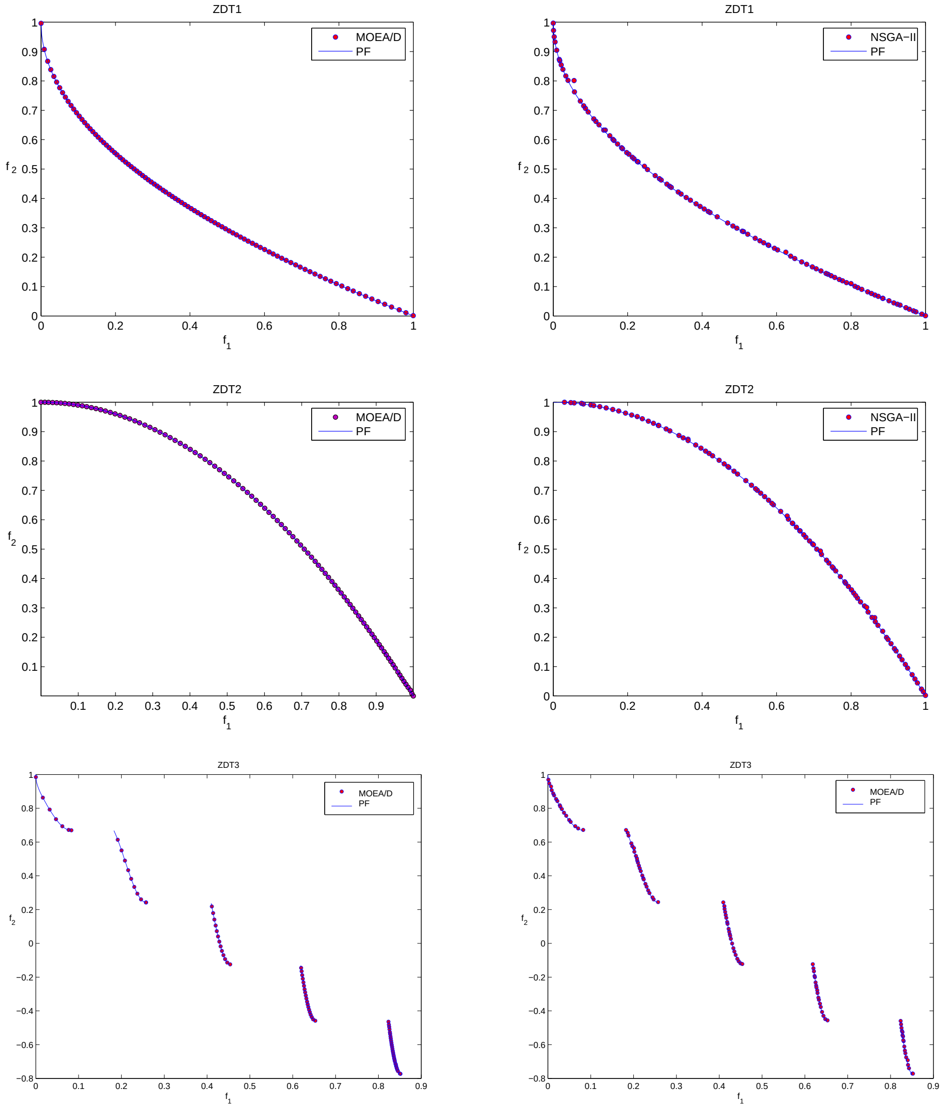


Figure 4.5: Plot of the nondominated fronts with the lowest D-metric value found by MOEA/D with the Tchebycheff approach (left) and NSGA-II (right) for three 2-objective test instances with 30 variables.

Table 4.3: *D*-metric values of the solutions found by MOEA/D with the Tchebycheff approach and NSGA-II. The numbers in parentheses represent the standard deviation.

Instance	NSGA-II	MOEA/D with Tchebycheff
ZDT1	0.0050 (0.0002)	0.0055 (0.0039)
ZDT2	0.0049 (0.0002)	0.0079 (0.0109)
ZDT3	0.0065 (0.0054)	0.0143 (0.0091)
ZDT4	0.0182 (0.0237)	0.0076 (0.0023)
ZDT6	0.0169 (0.0028)	0.0042 (0.0003)
DTLZ1	0.0648 (0.1015)	0.0317 (0.0005)
DTLZ2	0.0417 (0.0013)	0.0389 (0.0001)

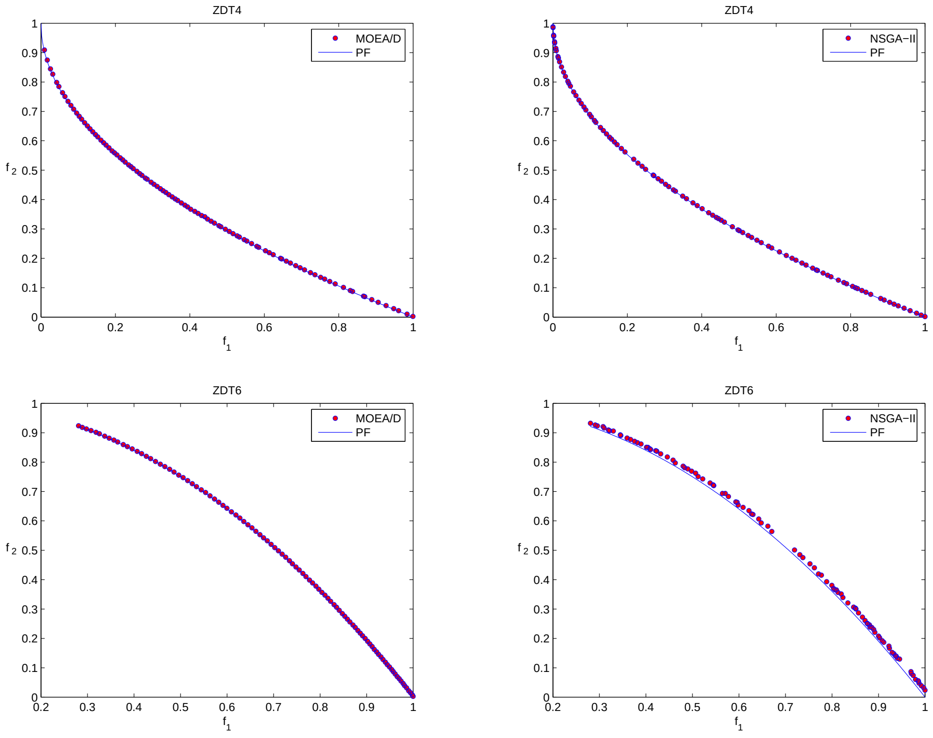


Figure 4.6: Plot of the nondominated fronts with the lowest D-metric value found by MOEA/D with the Tchebycheff approach (left) and NSGA-II (right) for two 2-objective test instances with 10 variables.

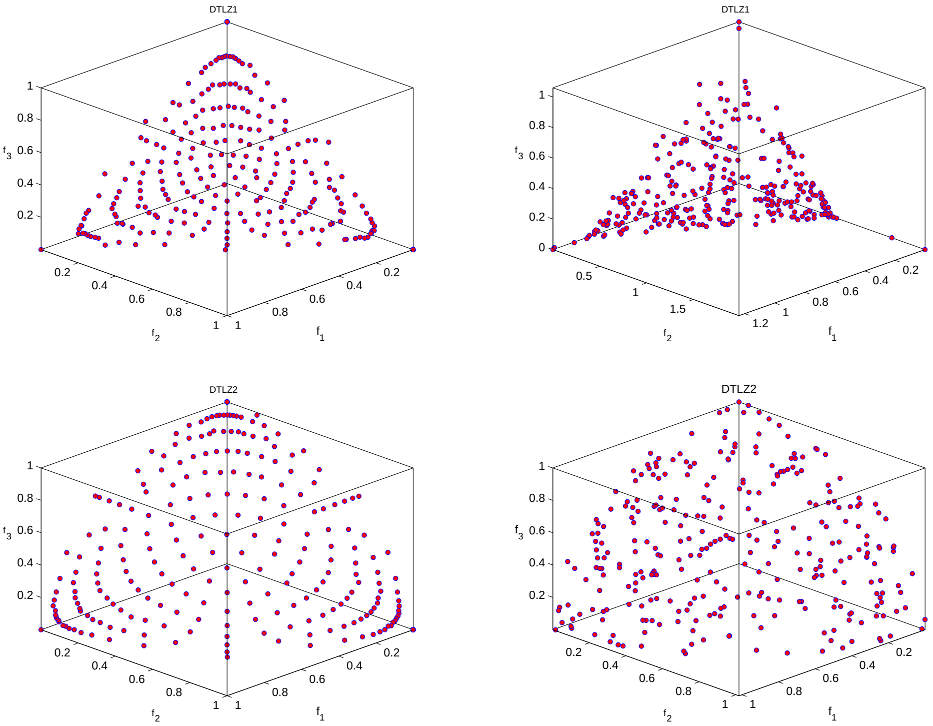


Figure 4.7: Plot of the nondominated fronts with the lowest D-metric value found by MOEA/D with the Tchebycheff approach (left) and NSGA-II (right) for each 3-objective test instance.

distributed points in the PF for all the 2-objective test instances, and 990 points for 3-objective instances.

The number of function evaluations matters much when the objective functions are very costly to evaluate. Figure 4.4 presents the evolution of the average D-metric value of the current population to P^* with the number of function evaluations in each algorithm for each test instance. These results indicate that MOEA/D converges, in terms of the number of the function evaluations, much faster than NSGA-II in minimizing the D-metric value for ZDT4, ZDT6, DTLZ1 and DTLZ2, and at about the same speed as or a bit slower than NSGA-II for the other three test instances.

Table 4.2 shows that, in terms of the C-metric, the final solutions obtained by MOEA/D are better than those obtained by NSGA-II for all the test instances but ZDT4. Table 4.3 presents the mean and standard deviation of the D-metric value of the final solutions obtained by each algorithm for each test instance. This table reveals that, in terms of the D-metric, the final solutions obtained by MOEA/D are better than NSGA-II for ZDT4, ZDT6 and the two 3-objective instances, and slightly worse for other three instances.

Figures 4.5-4.7 show, in the objective space, the distribution of the final solutions obtained in the run with the lowest D-value of each algorithm for each test instance. It is evident that, as to the uniformness of final solutions, MOEA/D is better than NSGA-II for ZDT1, ZDT2, ZDT6 and the two 3-objective test instances, is about the same as NSGA-II for ZDT4, and worse than NSGA-II for ZDT3, on which MOEA/D finds fewer solutions than NSGA-II in the two left segments of the PF in ZDT3. The poor performance of MOEA/D on ZDT3 could be attributed to the fact that the objectives in ZDT3 are disparately scaled.

We can conclude from the above results that MOEA/D with the Tchebycheff approach needs less CPU time than NSGA-II with the same number of function evaluations for these test instances. In terms of the solution quality, these two algorithms have about the same performance on the 2-objective test instances, but MOEA/D is better than NSGA-II on the two 3-objective instances.

4.4 MOEA/D with Other Decomposition Techniques

In the above experiments, we used a very naive implementation of MOEA/D. The decomposition method is the classic Tchebycheff approach. We did not perform any objective normalization.

As mentioned in Chapter 2, the Tchebycheff approach may perform worse, in terms of solution uniformness, than BI approaches, particularly when the number of objectives is more than two. It is also well known that objective normalization is very useful for increasing the solution uniformness when the objectives are disparately scaled.

Revisiting Figures 4.5 and 4.7, one may not be quite satisfied with the uniformness in the solutions found by the above implementation of MOEA/D for ZDT3, DTLZ1 and DTLZ2. Note that the two objectives in ZDT3 are disparately scaled, and DTLZ1 and DTLZ2 have 3 objectives, two questions naturally arise:

- Can MOEA/D with objective normalization perform better in the case of disparately-scaled objectives as in ZDT3?
- Can MOEA/D with other advanced decomposition methods such as the PBI approach find more evenly distributed solutions for 3-objective test instances like DTLZ1 and 2?

We have performed some experiments to study these two issues. In the following, we report our experimental results.

4.4.1 MOEA/D with Objective Normalization

We still use the Tchebycheff approach and do not change the parameter settings in the following experiments. But we incorporate a simple objective normalization technique into MOEA/D to study if the performance can be improved in the case of disparately-scaled objectives.

A lot of effort has been made on the issue of objective normalization in the communities of both mathematical programming and evolutionary computation [27][66][74][83]. A very simple normalization method is to replace objective f_i $(i = 1, \dots, m)$ by:

$$\bar{f^i} = \frac{f_i - z_i^*}{z_i^{nad} - z_i^*} \tag{4.3}$$

where $z^* = (z_1^*, \dots, z_m^*)^T$, as in Chapter 2, is the reference point, $z^{nad} = (z_1^{nad}, \dots, z_m^{nad})^T$ is the nadir point in the objective space, i.e., $z_i^{nad} = max\{f_i(x)|x \in PS\}$. In other words, z^{nad} defines the upper bound of PF, the Pareto front. In such a way, the range of each objective in the PF becomes $[0, 1]$.

It is not easy or necessary to compute z^{nad} and z^* beforehand. In our implementation, we replace z^* by z,[1] and each z_i^{nad} by $\tilde{z}_i^{nad}$, the largest value of f_i in the current population. Therefore, in the step 2.3 of MOEA/D with the Tchebycheff approach for continuous MOPs, g^{te} is replaced by:

$$\max_{1 \le i \le m} \{\lambda_i | \frac{f_i - z_i}{\tilde{z}_i^{nad} - z_i} |\}. \tag{4.4}$$

We have tested MOEA/D with the Tchebycheff approach and the above normalization technique on ZDT3 and a modified version of ZDT1 in which f_2 is replaced by $10f_2$ such that the two objectives are disparately scaled. Figure 4.8 plots the nondominated solutions obtained by a single run of MOEA/D with and without normalization for the modified

[1] Please refer to the description of MOEA/D for the details of initialization and update of z.

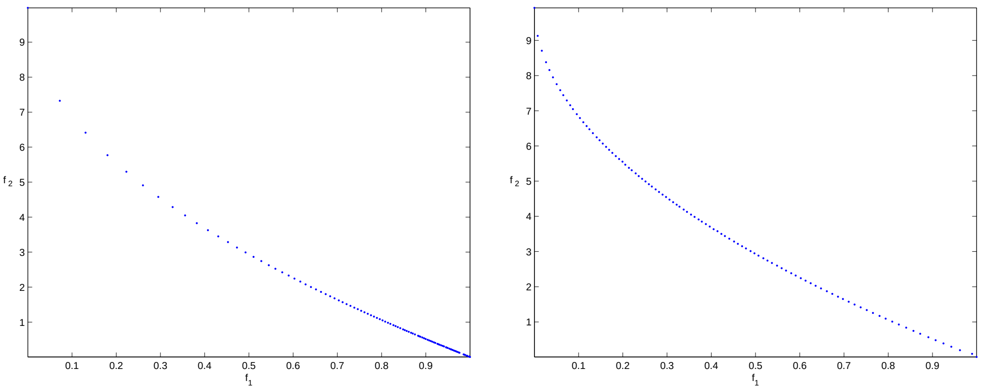


Figure 4.8: Plot of the nondominated fronts found by MOEA/D without(left) and with (right) normalization for the modified version of ZDT1 in which f_2 is replaced by $10f_2$.

version of ZDT1. Figure 4.9 shows the nondominated solutions obtained by a single run of MOEA/D with normalization for ZDT3.

Figures 4.8 and 4.9, together with the plots of the solutions found by NSGA-II and MOEA/D without normalization for ZDT3, clearly indicate that normalization does improve MOEA/D significantly, in terms of uniformness, in the case of disparately-scaled objectives.

4.4.2 MOEA/D with the PBI approach

We have tested MOEA/D with the PBI approach on all seven test instances. In our experiment, the penalty factor in g^{bip} is set to be 5 and the other settings are exactly the same as in the implementation of MOEA/D with the Tchebycheff approach in the above subsection.

Table 4.4 compares the average D-metric values of NSGA-II and MOEA/D with two different decomposition approaches. Figure 4.10 shows the distributions of the nondominated fronts with the lowest D-metric values in 30 independent runs of MOEA/D with the PBI approach for two 3-objective test instances. It is very clear from these results and

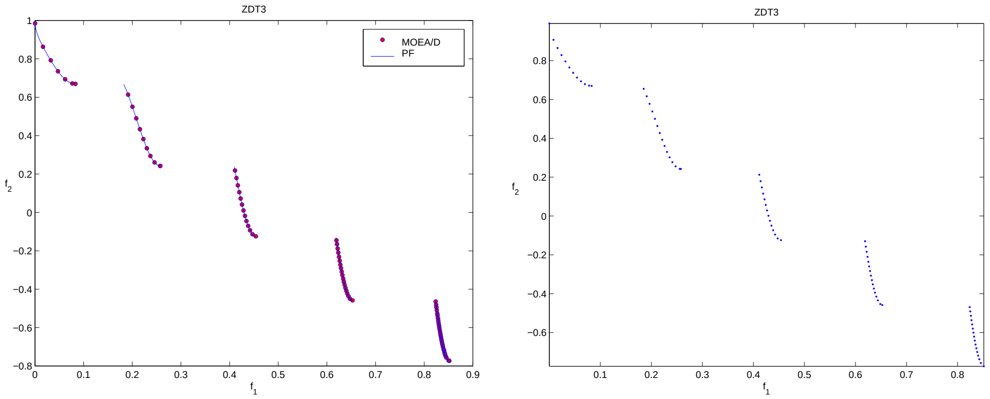


Figure 4.9: Plot of the nondominated fronts found by MOEA/D without (left) and with (right) normalization for ZDT3.

Table 4.4: Comparison of Average *D*-metric values of the solutions found by NSGA-II, and MOEA/D with the Tchebycheff and PBI approaches. The numbers in parentheses represent the standard deviation.

Instance	NSGA-II	MOEA/D with Tchebycheff	MOEA/D with PBI
ZDT1	0.0050 (0.0002)	0.0055 (0.0039)	0.0211 (0.0204)
ZDT2	0.0049 (0.0002)	0.0079 (0.0109)	0.0376 (0.0396)
ZDT3	0.0065 (0.0054)	0.0143 (0.0091)	0.0249 (0.0281)
ZDT4	0.0182 (0.0237)	0.0076 (0.0023)	0.0265 (0.0239)
ZDT6	0.0169 (0.0028)	0.0042 (0.0003)	0.0165 (0.0040)
DTLZ1	0.0648 (0.1015)	0.0317 (0.0005)	0.0232 (0.0018)
DTLZ2	0.0417 (0.0013)	0.0389 (0.0001)	0.0280 (4.7034e-006)

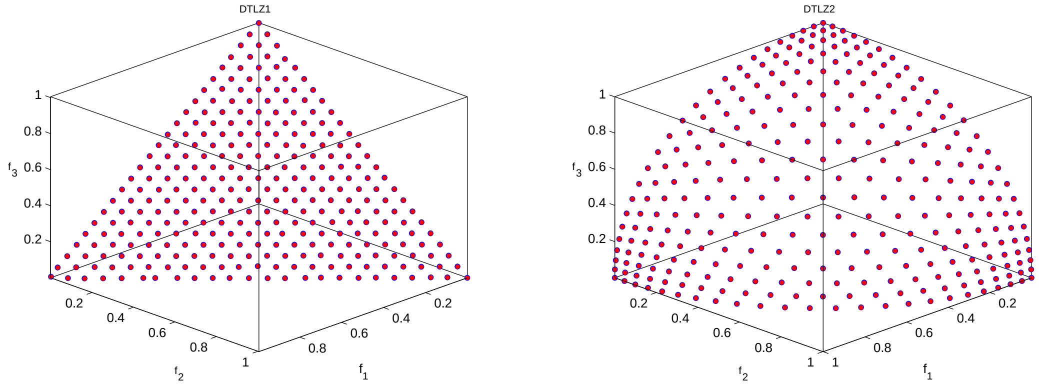


Figure 4.10: Plot of the nondominated fronts with the lowest D-metric values in MOEA/D with the PBI approach for DTLZ1 and DTLZ2.

Figure 4.7 that MOEA/D with the PBI approach performs much better than NSGA-II and MOEA/D with the Tchebycheff approach on these two 3-objective instances. These results suggest that incorporating other advanced decomposition approaches into MOEA/D would be worthwhile studying for solving MOPs with more than 2 objectives.

From the results in Table 4.4, it is clear that MOEA/D with PBI approach performs slightly worse than the other two algorithms for five 2-objective test instances except ZDT6. For ZDT1-ZDT3, the standard deviations of D-metric values found by MOEA/D with the PBI approach are larger than those of NSGA-II and MOEA/D with Tchebycheff approach. This indicates that MOEA/D with the PBI approach is not stable for these test instances.

The distributions of the nondominated fronts with the lowest D-metric values in 30 independent runs of MOEA/D with the PBI approach for five 2-objective test instances are also visualized in Figure 4.11. These results suggest that MOEA/D with the PBI approach can still find a good approximation of PFs for these test instances in some runs. It can also be observed from Figure 4.11 that some solutions found by MOEA/D with the PBI approach for ZDT3 is not Pareto-optimal. The reason is that the optimal solutions of some

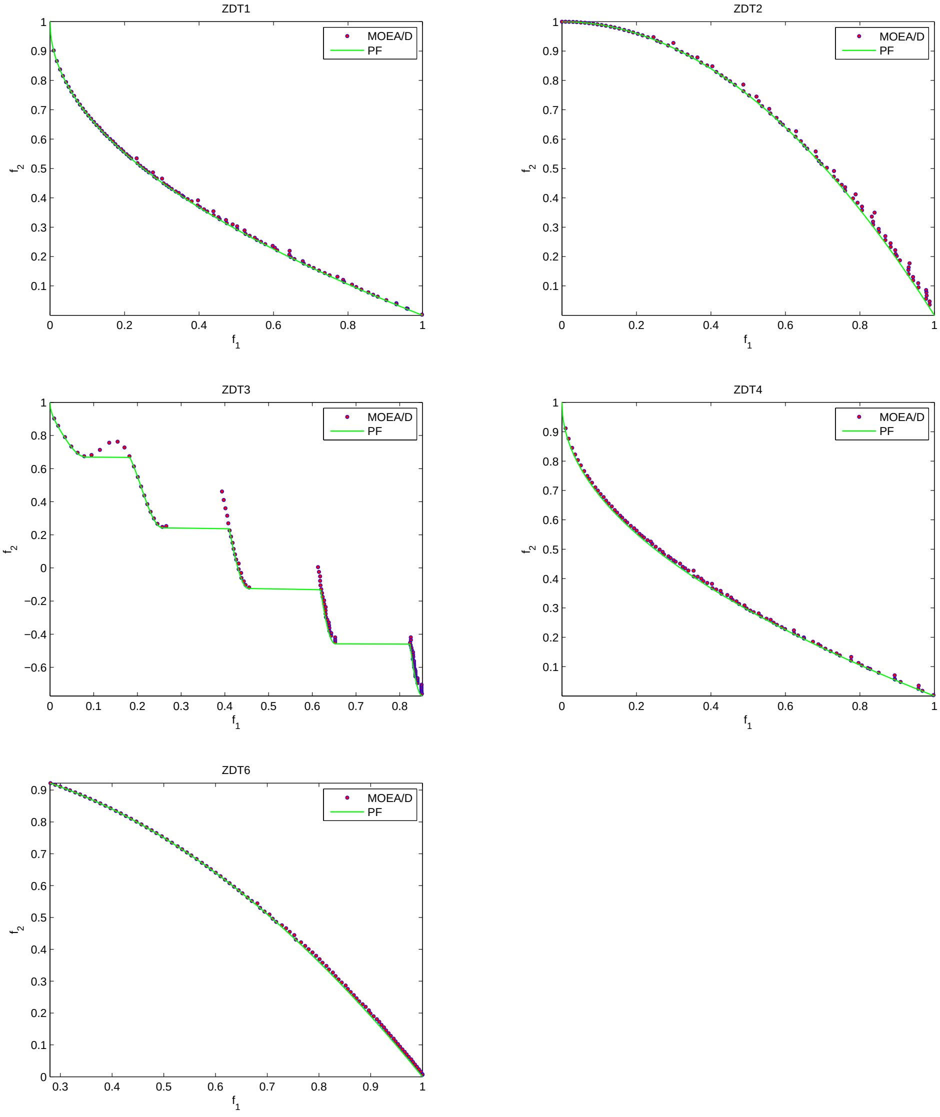


Figure 4.11: Plot of the nondominated fronts with the lowest D-metric value found by MOEA/D with the PBI approach for five 2-objective test instances.

scalar subproblems in PBI approach are the intersection points between the line segments passing the reference point and the boundary of feasible objective region, which are not Pareto-optimal to the MOP under consideration.

We would like to point out that MOEA/D with advanced decomposition approaches may require a bit more human effort, for example, to set the value of the penalty factor in the PBI approach.

In summary, we have very positive answers to the two questions raised at the outset of this subsection. Using other sophisticated decomposition and normalization techniques in MOEA/D will be our future research topics.

4.4.3 MOEA/D with the Weighted Sum Approach

In this section, we have tried the version of MOEA/D with the weighted sum approach for ZDT1, ZDT2 and ZDT3. All parameter settings remain the same as those in the previous sections.

Figure 4.12 shows that distribution of the final solutions found by MOEA/D with the weighted sum method. It is clear that the nondominated solutions found by this version of MOEA/D are not uniform along the PF for ZDT1. This suggests that MOEA/D with the weighted sum approach may not work well in diversifying the nondominated solutions even if the PF is convex and all weight vectors are uniformly distributed. The results in Figure 4.12 also reveal that MOEA/D with the weighted sum approach fails to find the whole PFs of ZDT2 and ZDT3. This is because the weighted sum approach cannot solve the MOPs with nonconvex PFs.

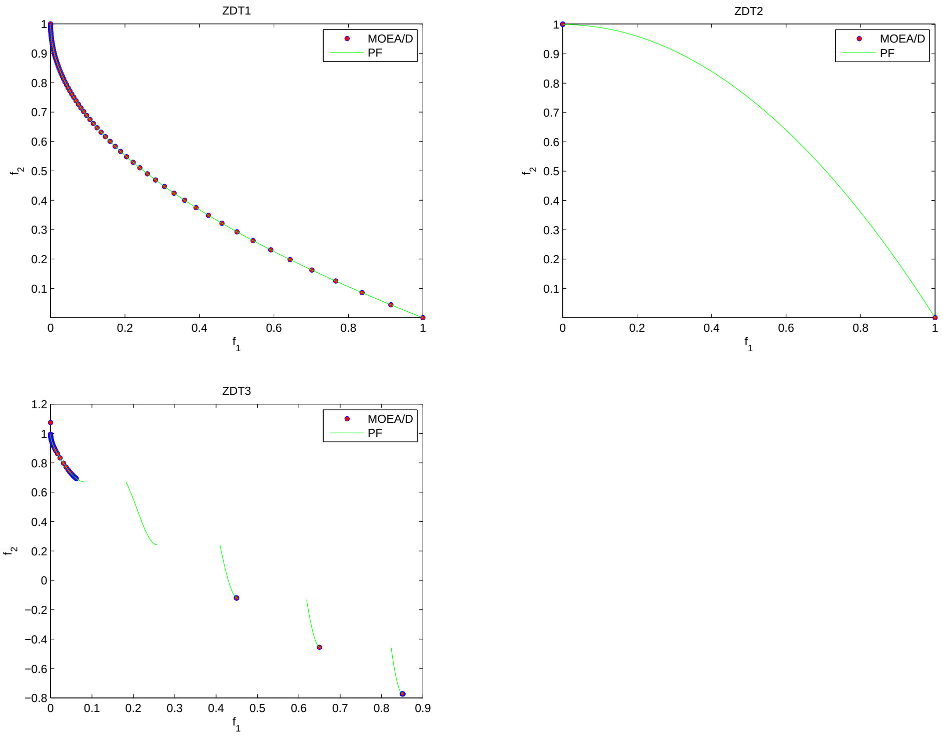


Figure 4.12: Plots of the nondominated fronts found by MOEA/D with the weighted sum approach on ZDT1, ZDT2 and ZDT3

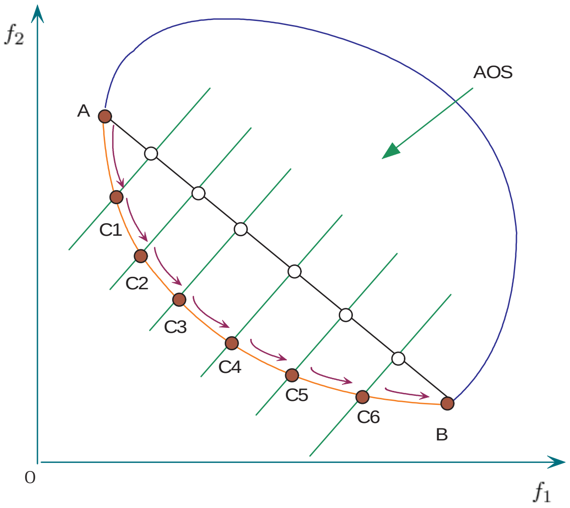


Figure 4.13: Subproblems solved by single objective optimizers sequentially. Points A and B are the individual minimal solutions of f_1 and f_2 respectively. $C1 - C6$ are the optimal solutions of 6 subproblems in NBI.

4.5 Comparison with NBI

4.5.1 Local Search Based NBI

Normal Boundary Intersection (NBI) [13] is an efficient multiobjective search methods based on decomposition. It solves a number of single objective subproblems sequentially by single objective optimizers, such as local search or EA. In our work, we consider the sequential quadratic programming optimizer (SQP) to perform local search for solving each subproblem in NBI. The similar idea was also employed in a two phase local search for multiobjective travelling salesman problem [73].

In NBI, each subproblem corresponds to a reference point, which is associated with a weight vector. In fact, the subproblems with similar weight vectors should have close optimal solutions in the decision space. For this reason, the starting point of a local search method for the current subproblem can be set as the solutions previously found for its neighboring subproblems. In such a way, the local search method is expected to converge in relatively few number of function evaluations. Figure 4.13 illustrates the NBI for solving subproblems in a sequential way.

4.5.2 Experimental Results

In our experiments, we have implemented NBI in matlab. To have a fair comparison with MOEA/D, the number of subproblems is set to 100, which is the same as the population size in MOEA/D. The weight vectors of subproblems are generated in the same way as that in MOEA/D. For each subproblem, the maximal number of function evaluations used in the SQP optimizer is set to 250. We have tested NBI for five 2-objective test instances.

Figure 4.14 plots the final solutions in the run with the lowest D-metric value

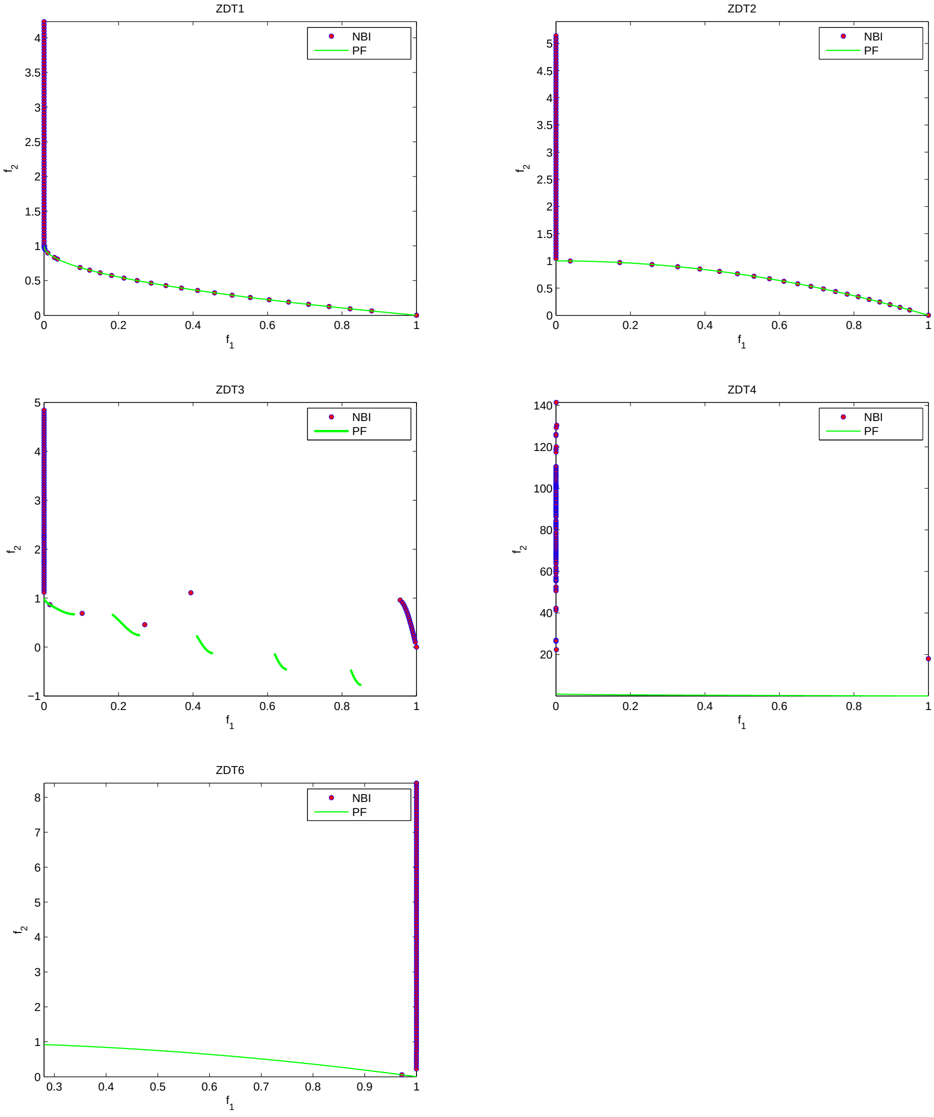


Figure 4.14: Plots of the final solutions in the run with the lowest D-metric value found by NBI for five 2-objective test instances

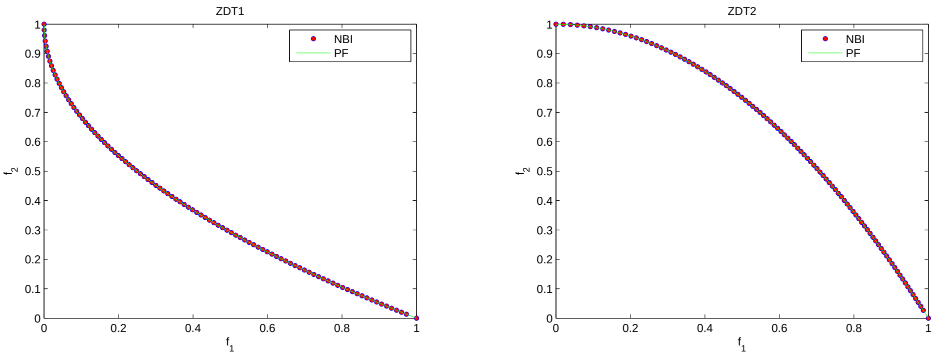


Figure 4.15: Plots of the final solutions in the run with the lowest D-metric value found by NBI for ZDT1 and ZDT2

found by NBI for all five test instances. It can be seen from this figure that not only Pareto-optimal solutions but also some weakly Pareto-optimal solutions are found by NBI for ZDT1 and ZDT2. The main reason is that the optimal solutions of many subproblems are weakly Pareto-optimal since the individual minima found by the SQP optimizer do not belong to the PF.

We can also observe from Figure 4.14 that NBI performs very poorly in finding good approximation of PFs for ZDT3, ZDT4 and ZDT6. This can be explained by the fact that the objective functions of these test instances have local optima. Therefore, the SQP optimizer in NBI gets trapped in the local optima of subproblems.

Compared with the results in Figures 4.5 and 4.6, it is clear that MOEA/D performs much better than NBI for all test instances. This result indiciates that the SQP optimizer in NBI is not suitable for solving the MOPs with multimodal objective functions. In contrast, MOEA/D solves multiple subproblems by using an EA and has the potential to find the global optima of multimodal objective functions.

Figure 4.15 shows the nondominated fronts found by NBI for ZDT1 and ZDT2 when the individual minima of all objectives in the PF are known in advance. It is clear that NBI works pretty well both in diversity and in convergence. This is due to two reasons: 1) NBI starts from a known Pareto-optimal solution in the PF and 2) the Pareto-optimal solutions of ZDT1 and ZDT2 are connected both in the objective space and in the decision space. As pointed out in [46], the success of local search can benefit from the connectedness of the Pareto-optimal solutions. In this case, both MOEA/D and NBI perform similarly.

4.6 Small Population, Scalability and Sensitivity in MOEA/D

4.6.1 MOEA/D Using Small Population

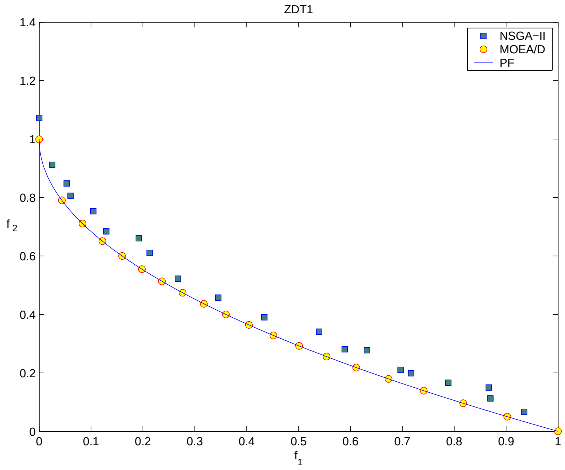


Figure 4.16: Plot of the nondominated fronts found by MOEA/D and NSGA-II using small population ($N = 20$).

A decision maker may not want to have a huge number of Pareto optimal solutions at high computational cost. They are often interested in obtaining a small number of evenly distributed solutions at low computational cost. In the following, we will show that MOEA/D using small population could serve for this purpose. We take ZDT1 as an example and use MOEA/D with the Tchebycheff approach. All the parameter settings are the same as in Section 4.3.3 except the population size $N = 20$. For comparison, we also run NSGA-II with $N = 20$ on ZDT1. Both algorithms stop after 250 generations as in Section 4.3.3.

Figure 4.16 plots the final solutions obtained in a single run of NSGA-II and MOEA/D with $N = 20$. It is evident that MOEA/D found 20 very evenly distributed Pareto solutions while NSGA-II failed in reaching the PF within the given number of generations. Clearly, this advantage of MOEA/D comes from its decomposition strategy.

4.6.2 Sensitivity of Neighborhood Size in MOEA/D

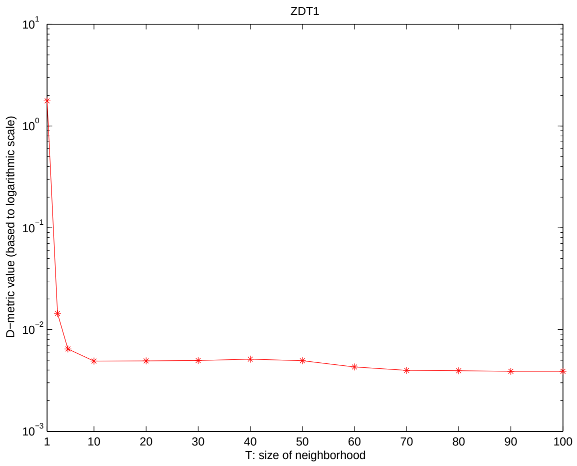


Figure 4.17: The average D-metric value vs. the value of T in ZDT1.

T is a major control parameter in MOEA/D. To study the sensitivity of the performance to T in MOEA/D for both continuous MOPs, we have tested different settings of T in the implementation of MOEA/D with the Tchebycheff approach for ZDT1. All the parameter settings are the same as in Section 4.3.3, except the settings of T.

Figure 4.17 reveals that MOEA/D does not work well on both instances when T is very small. This could be due to that, as mentioned in Chapter 3, MOEA/D with too small T is poor at exploration. The reason that MOEA/D with large T performs well on ZDT1 could be because that even if two weight vectors are far different, the optimal solutions to their associated subproblems are very similar, in fact, they are the same in the all the components except the first one.

4.6.3 Scalability

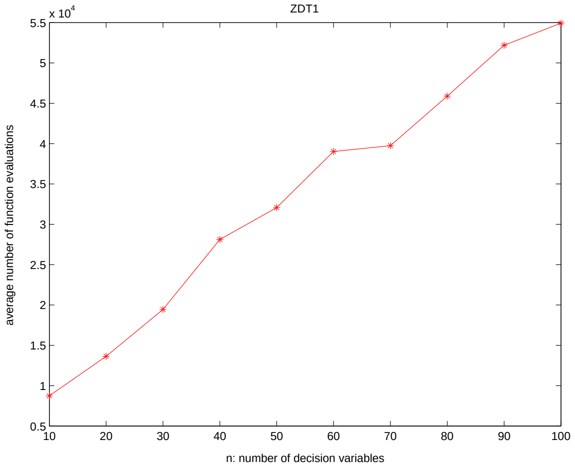


Figure 4.18: The average numbers of function evaluations used for reducing the D-metric value under 0.005 for ZDT1 with different numbers of decision variables.

To study how the computational cost, in terms of the number of function evaluations, increase as the number of decision variable increases, we have tried, on ZDT1 with different numbers of decision variables, the variant of MOEA/D with the same parameter settings except the maximal number of generations is $1,000$ (i.e., the maximal number of function evaluations is $100 \times 1,000 = 10^5$). We have found that in every run among 30 independent runs for each number of decision variables, MOEA/D has lowered the D-metric value below 0.005, therefore the final solutions are a very good approximation to the PF by our experience in the previous two sections. Figure 4.18 gives the average number of function evaluations used for reducing the D-metric value under 0.005 for ZDT1 with different numbers of decision variables. It is evident that the average number of function evaluations scales up linearly as the number of decision variables increases.

The linear scalability of MOEA/D in this test instance should be due to two facts: 1) the number of scalar optimization subproblems in MOEA/D is fixed to be 100 no matter how many the decision variables there are. One does not need to increase the number of subproblems as the number of decision variables increases since the intrinsic dimensionality of a MOP is determined by its number of objectives, but not its number of decision variables [20][66]. 2) The complexity of each scalar optimization could scale up linearly with the number of decision variables.

4.7 Summary

In this chapter, the MOEA/D with the Tchebycheff approach was proposed for continuous MOPs, in which the SBX operator and polynomial mutation were applied to produce offspring solutions. The best solutions of all scalar subproblems were maintained and returned as the approximation of PFs in MOEA/D. MOEA/D with the weighted Tchebycheff approach was then compared with two other well-known multiobjective algorithms - NSGA-II and NBI on a benchmark set of continuous test instances. Besides, two other decomposition techniques - PBI and objective normalization were also studied in MOEA/D for finding the good distribution of the nondominated solutions in the final population. The weighted sum approach was also tested in MOEA/D. Additionally, small population, scalability and sensitivity in MOEA/D were investigated. The following are the major conclusions from our experimental results.

- MOEA/D performs better than NSGA-II in terms of computational time for all test instances. For five 2-objective test instances, MOEA/D performs similarly to NSGA-II in terms of C-metric and D-metric. The distribution of nondominated solutions in MOEA/D is better than those in NSGA-II for two 3-objective test instances - DTLZ1 and DTLZ2.

- The performance of MOEA/D in finding a diverse set of nondominated solutions can be improved by objective normalization for the test instances with disparately scaled objectives and by the PBI for two 3-objective test instances - DTLZ1 and DTLZ2.

- MOEA/D clearly outperforms the SQP-based NBI for five 2-objective test instances since MOEA/D has the ability to deal with the multimodality of objective functions and updates the reference point adaptively.

- MOEA/D performs better than NSGA-II when small population is used. Due to the special features of the PS in ZDT test instances (i.e., all Pareto-optimal solutions have the same values in all its variables except the first one), MOEA/D is not sensitive to the setting of the neighborhood size. As the dimensionality of test instances increases, the number of function evaluations needed by MOEA/D to find a good approximation of PF linearly scales up.

Chapter 5

MOEA/D for Multiobjective Knapsack Problem

5.1 Introduction

Given a set of n items and a set of m knapsacks, the multiobjective 0-1 knapsack problem (MOKP) can be stated as:

$$\begin{aligned} \text{maximize} \quad & f_i(x) = \sum_{j=1}^{n} p_{ij} x_j, \ i = 1, \dots, m && (5.1) \\ \text{subject to} \quad & \sum_{j=1}^{n} w_{ij} x_j \leq c_i, \ i = 1, \dots, m && (5.2) \\ & x = (x_1, \dots, x_n)^T \in \{0, 1\}^n \end{aligned}$$

where $p_{ij} \geq 0$ is the profit of item j in knapsack i, $w_{ij} \geq 0$ is the weight of item j in knapsack i, and c_i is the capacity of knapsack i. $x_i = 1$ means that item i is selected and put in all the knapsacks. The MOKP is $\mathcal{NP}$-hard and can model a variety of applications in resource allocation. In [106], Zitzler suggested a set of test instances for multiobjective 0/1 knapsack problems, which have been widely used in comparing the performance MOEAs

[49][44][60][37].

As mentioned in Chapter 3, the combinations of EAs with local search or problem-specific heuristics, called memetic algorithms, are very promising for solving hard optimization problem. In this chapter, we develop the version of MOEA/D with local heuristics for solving multiobjective 0/1 knapsack problem. The following are the main features in the new version of MOEA/D.

- An advanced repair heuristic is applied to maintain the feasibility of each child solution. The quality of child solutions can also be preserved locally since the repair heuristic is guided by a scalar function.
- An external population is maintained to store all nondominated solutions found during the search.

The rest of this chapter is organized as follows. The next section describes the version of MOEA/D with local heuristics for MOKP. In the following section the comparison of MOEA/D with MOGLS is provided. The sensitivity of neighborhood size in MOEA/D is then analyzed. The final section summarizes this chapter.

5.2 MOEA/D with Local Heuristics

5.2.1 Heuristics for Constraints Handing

EA operators, such as crossover and mutation, are very likely to generate infeasible solutions during the evolutionary process due to the constraints on the capacities of knapsacks. In this case, repair heuristics are often needed to maintain the feasibility of solutions. The basic idea of repair heuristics is to remove some items from the knapsacks in a specific order without losing too much quality of solutions.

In [44], Jaszkiewicz proposed a heuristic repair method for dealing with constraints of MOKP according to the proposition of Michalewicz and Arabas [65], where the item's profit is replaced by the weighted sum of item's profits. The item k is determined by:

$$k = \arg\min_{j \in J} \frac{\sum_{i=1}^{m} \lambda_i p_{ij}}{\sum_{i=1}^{m} w_{ij}}. \tag{5.3}$$

where $J = \{j | x_j = 1 \text{ and } 1 \leq j \leq n\}$. In other words, the items with higher weights and lower cost are more likely to be removed. This procedure locally decreases the weights in the knapsacks at the lowest cost of current scalarizing function.

Recently, Jaszkiewicz proposed and used the following greedy repair method in the latest version of his implementation, which can be downloaded from his website[1] and is slightly better than that used in his earlier paper [44]. The item k for removal at each iteration is obtained by:

$$k = \arg\min_{j \in J} \frac{g(x) - g(x^{j-})}{\sum_{i \in I} w_{ij}} \tag{5.4}$$

where $J = \{j | x_j = 1 \text{ and } 1 \leq j \leq n\}$, $g : \{0,1\}^n \rightarrow R$ is the objective function to be maximized, $I = \{i | 1 \leq i \leq m \text{ and } \sum_{j=1}^{n} w_{ij} x_j > c_i\}$ is the set of dimensions violating constraints, x^{j-} is different from x only in position j, i.e., $x_i^{j-} = x_i$ for all $i \neq j$ and $x_j^{j-} = 0$. In this approach, items are removed one by one from y until y becomes feasible. An item with the heavy weights (i.e., $\sum_{i \in I} w_{ij}$) in the overfilled knapsacks and little contribution to $g(x)$ (i.e., $g(y) - g(y^{j-})$) is more likely to be removed.

5.2.2 Algorithm

In this section, we describe the version of MOEA/D with local heuristics for MOKP. Both the weighted sum approach and the weighted Tchebycheff approach are used for decomposing MOP into N subproblems. At each generation t, MOEA/D maintains:

[1] http://www-idss.cs.put.poznan.pl/ jaszkiewicz/

```
Input  MOP in (5.1); stopping criteria; $N$: the number of the subproblems; $\lambda^1, \dots, \lambda^N$: $N$ weight
    vectors; $T$: the number of the weight vectors in the neighborhood of each weight vector.
Output  $EP$ : external population
Step 1  Initialization
    1.1 Set $EP = \emptyset$.
    1.2 Compute the Euclidean distances between any two weight vectors and then work out the $T$
    closest weight vectors to each weight vector. For each $i = 1, \dots, N$, set $B(i) = \{i_1, \dots, i_T\}$
    where $\lambda^{i_1}, \dots, \lambda^{i_T}$ are the $T$ closest weight vectors to $\lambda^i$.
    1.3 Generate an initial population $x^1, \dots, x^N$ randomly or by a problem-specific method.
    Set $FV^i = F(x^i)$
    1.4 Initialize $z = (z_1, \dots, z_m)^T$ by a problem-specific method.
Step 2  Update
    For each $i \in \{1, \dots, N\}$
    2.1 Reproduction: Randomly select two indices $p, q$ from $B(i)$, and then generate a new
    solutions $y$ from $x^p$ and $x^q$ by using genetic operators.
    2.2 Improvement: Apply a problem-specific repair/improvement heuristics on $y$ to generate
    $y'$.
    2.3 Update of $z$: For each $j = 1, \dots, m$, if $z_j < f_j(y')$, then set $z_j = f_j(y')$.
    2.4 Update of Neighboring Solutions: For each index $j \in B(i)$, if $g^{te}(y'|\lambda^j, z) \leq$
    $g^{te}(x^j|\lambda^j, z)$, then set $x^j = y'$ and $FV^j = F(y')$.
    2.5 Update of $EP$:
    Remove from $EP$ all the vectors dominated by $F(y')$.
    Add $F(y')$ to $EP$ if no vectors in $EP$ dominate $F(y')$.
Step 3  Stopping Criteria If the stopping criteria are satisfied, then stop and output $EP$. Other-
    wise, go to Step 2.
```

Figure 5.1: MOEA/D with Local Heuristics

- a population $x^1, x^2, \dots, x^N$ points, where x^i is the current solution of the i^{th} subproblem;
- the vector $z = (z_1, \dots, z_m)^T$, where z^i the best value found so far for objective f_i;
- an external population EP, which is used to store nondominated solutions found during the search.

Assuming that the weighted Tchebycheff approach is employed, MOEA/D with local heuristics is illustrated in Figure 5.1. Initially, the external population is set to be empty. For each subproblem i, its neighborhood $B(i)$ is defined in the same way as that in Chapter 4. In Steps 1.4 and 2.2, the problem-specific heuristics are applied to improve the quality of solutions. The external population EP is maintained in Step 1.1 and Step 2.5 and returned in Step 3.

5.3 Comparison with MOGLS

In the following, we first introduce MOGLS and then analyze the complexity of MOGLS and MOEA/D. We also compare the performances of these two algorithms on a set of test instances of the multiobjective 0/1 knapsack problem. The major reasons for choosing MOGLS for comparison are that it is also based on decomposition and performs better than a number of popular algorithms on the multiobjective 0/1 knapsack problem [44].

5.3.1 MOGLS

MOGLS was first proposed by Ishibuchi and Murata in [38], and further improved by Jaszkiewicz [44]. The basic idea is to reformulate the MOP (5.1) as simultaneous op-

Input MOP in (5.1); stopping criteria; K : the size of temporary elite population; S : the size of initial population.

Output EP : external population

Step 1 Initialization

1.1 Generate S initial solutions $x^1, \dots, x^S$ randomly or by a problem-specific method. Then CS is initialized to be $\{x^1, \dots, x^S\}$;

1.2 Initialize $z = (z_1, \dots, z_m)^T$ by a problem-specific method;

1.3 EP is initialized to be the set of the F-values of all the nondominated solutions in CS;

Step 2 Update

2.1 Reproduction:

Generate a random weight vector λ uniformly;

From CS select the K best solutions, with regard to Tchebycheff aggregation function g^{te} with the weight vector λ, to form a temporary elite population (TEP).

Draw at random two solutions from TEP, and then generate a new solution y from these two solutions by using genetic operators.

2.2 Improvement: Apply a problem-specific repair/improvement heuristics on y to generate y'.

2.3 Update of z: For each $j = 1, \dots, m$, if $z_j < f_j(y')$, then set $z_j = f_j(y')$.

2.4 Update of Solutions in TEP:

If y' is better than the worst solution in TEP with regard to g^{te} with the weight vector λ and different from solutions in TEP with regard to F-values, then add it to the set CS.

If the size of CS is larger than $K \times S$, delete the oldest solution in CS.

2.5 Update of EP:

Remove from EP all the vectors dominated by $F(y')$.

Add $F(y')$ to EP if no vectors in EP dominate $F(y')$.

Step 3 Stopping Criteria If the stopping criteria are satisfied, then stop and output EP. Otherwise, go to **Step 2**.

Figure 5.2: MOGLS

timization of all weighted Tchebycheff functions or all weighted sum functions. In the following, we give a brief description of Jaszkiewicz's version of MOGLS.

At each iteration, MOGLS maintains: 1) a set of current solutions CS, and the F-values of these solutions; 2) an external population EP, which is used to store nondominated solutions. 3) $z = (z_1, \ldots, z_m)^T$, where z_i is the largest value found so far for objective f_i. MOGLS needs two control parameters K and S. K is the size of its temporary elite population and S is the initial size of CS. MOGLS works as follows:

As in MOEA/D with the Tchebycheff approach, z is used as a substitute for z^* in g^{te}. In the case of MOGLS with the weighted sum approach, g^{te} should be replaced by g^{ws} and there is no need to store z. Therefore, Step 2.3 should be removed. MOGLS needs to keep the F-values of all the current solutions since these F-values will be used in Step 2.4 for computing the values of g^{te}.

5.3.2 Comparison of complexity of MOEA/D and MOGLS

Space complexity

During the search, MOEA/D needs to maintain its internal population of N solutions, and external population EP, while MOGLS stores the set of current solutions CS and its external population EP. The size of CS increases until it reaches its upper bound, which is suggested to be set as $K \times S$ in [44]. Therefore, if $K \times S$ in MOGLS is much larger than N in MOEA/D and both algorithms produce about the same number of nondominated solutions, then the space complexity of MOEA/D is lower than that of MOGLS.

Computational complexity

The major computational costs in both MOEA/D and MOGLS are involved in their Step 2. Both a single pass of Step 2 in MOEA/D and Step 2 in MOGLS generate one new trial solution y. Let us compare the computational complexity in a single pass of Step 2 in MOEA/D and Step 2 in MOGLS:

- Step 2.1 in MOEA/D vs. Step 2.1 in MOGLS: MOGLS has to generate TEP. One has to compute the g^{te}-values of all the points in CS, which needs $O(m \times |CS|)$ basic operations. It also needs $O(K \times |CS|)$ basic operations for selecting TEP if a naive selection method is employed, while Step 2.1 in MOEA/D only needs to randomly pick two solutions for genetic operators. Note that $|CS|$, the size of CS, could be very large (e.g, it was set from $3,000$ to $7,000$ in [44]), the computational cost of Step 2.1 in MOGLS is much higher than that of Step 2.1 in MOEA/D.
- Steps 2.2 and 2.3 in MOEA/D are the same as Steps 2.2 and 2.3 in MOGLS, respectively.
- Step 2.4 in MOEA/D vs Step 2.4 in MOGLS: Step 2.4 in MOEA/D needs $O(T)$ basic operations while Step 2.4 in MOGLS needs $O(K)$ basic operations. In the case when T and K are close as in our experiments, there is no significant difference in computational costs between them.

Therefore, we can conclude that each pass of Step 2 in MOEA/D involves less computational cost than Step 2 in MOGLS does.

5.3.3 Implementations of MOEA/D and MOGLS for MOKP

Implementation of MOGLS

To have a fair comparison, we directly use Jaszkiewicz's latest implementation of MOGLS for the MOKP, the details of MOGLS with the Tchebycheff approach are given as follows:

- **Initialization of $z = (z_1, \dots, z_m)^T$:** Taking each f_i as the objective function, apply the repair method on a randomly generated point and produce a feasible solution. Set z_i to be the f_i value of the resultant point.
- **Initialization of EP and CS:**

  Set $EP = \emptyset$ and $CS = \emptyset$. Then **Repeat** S times:

  1. Randomly generate a weight vector λ by using the sampling method described in [44].
  2. Randomly generate a solution $x = (x_1, \dots, x_n)^T \in \{0, 1\}^n$, where the probability of $x_i = 1$ equals to 0.5.
  3. Taking $-g^{te}(x; \lambda, z)$ as the objective function, apply the repair method to x and obtain a feasible solution x'.
  4. Add x' to CS. Remove from EP all the vectors dominated by $F(y')$, then add $F(y')$ to EP if no vectors in EP dominate $F(y')$.
- **Genetic operators in Step 2.1:** The genetic operators used are the one-point crossover operator and the standard mutation operator. The one-point crossover is first applied to the two solutions and generates one child solution, then the standard mutation operator mutates it to produce a new solution y. The mutation mutates each position of the child solution independently with probability 0.01.
- **Heuristic in Step 2.2:** The repair method described in this section is used.

Implementation of MOEA/D

We use the same genetic operators in Step 2.1 and the repair method in Step 2.2 as in the implementation of MOGLS. The initialization of z is also the same as in MOGLS. Initialization of x^i (the initial solution to the $i-$th subproblem) is performed as follows:

- **Initialization of x^i in Step 1.3:** Taking $-g^{te}(x|\lambda^i, z)$ as the objective function, apply the repair method to a randomly generated solution. Set x^i to be the resultant solution.

Tchebycheff aggregation function g^{te} is used in the above implementations of MOEA/D and MOGLS.

The implementations of these two algorithms with the weighted sum approach in our experimental studies are the same as their counterparts with the Tchebycheff approach, except that they use g^{ws} as the objective in the repair method and do not maintain z.

5.3.4 Parameter Setting

The setting of S and K in MOGLS is the same as in [44]. K is set to 20 for all the instances. The values of S for different instances are given in Table 5.1. T in MOEA/D is set to 10 for all the test instances. N weight vectors $\lambda^1, \ldots, \lambda^N$ in MOEA/D are generated by using the same scheme in Chapter 4, in which the parameter H takes a positive integer value.

Table 5.1 lists the value of N and H in MOEA/D for each test instance. For the instances with 2 objectives, the value of N in MOEA/D is the same as that of S in MOGLS. For all the instances with three objectives, $H = 25$ and therefore $N = 351$. For all the instances with four objectives, $H = 12$ and then $N = 455$. Both of the algorithms stop after $500 \times S$ calls of the repair method.

In our experimental studies, both g^{ws} and g^{te} have been used in the repair method. In the following, W-MOEA/D (W-MOGLS) stands for MOEA/D (MOGLS) in which g^{ws} is used, while T-MOEA/D (T-MOGLS) represents MOEA/D (MOGLS) in which g^{te} is used.

Table 5.1: Parameter setting of MOEA/D and MOGLS for the test instances of the 0/1 knapsack problem. m is the number of knapsacks and n is the number of items.

Instance		S	N (H)
m	n	in MOGLS	in MOEA/D
2	250	150	150 (149)
2	500	200	200 (199)
2	750	200	250 (249)
3	250	200	351 (25)
3	500	250	351 (25)
3	750	300	351 (25)
4	250	250	455 (12)
4	500	300	455 (12)
4	750	350	455 (12)

5.3.5 Experimental Results

Both MOGLS and MOEA/D have been independently run for 30 times for each test instance on identical computers (Pentium(R) 3.2GHZ, 1.00 GB). Due to the nature of MOPs, multiple performance indices should be used for comparing the performances of different algorithms [15] [107]. In our experiments, set coverage and D-metric are used.

In the case when we don't know the actual PF, we can set P^* to be an upper approximation of the PF. Jaszkiewicz has produced a very good upper approximation to each 0/1 knapsack test instance by solving the linear programming relaxed version of weighted Tchebycheff approach in (2.3) with a number of uniformly distributed λ's [44]. The number of the points in the upper approximation is 202 for each of the bi-objective instances, 1326 for the 3-objective instances, and 3276 for the 4-objective instances. In our experiments, P^* is set as such an approximation.

Table 5.2 gives the average CPU time used by each algorithm for each instance. Table 5.3 presents the means of the C-metric values of the final approximations obtained

Table 5.2: Average CPU time (in second) used by MOEA/D and MOGLS. m is the number of knapsacks and n is the number of items.

Instance		Tchebycheff		Weighted Sum	
m	n	MOEA/D	MOGLS	MOEA/D	MOGLS
2	250	3.93	26.47	3.70	29.37
2	500	10.40	70.80	9.40	72.97
2	750	20.13	127.30	17.87	129.60
3	250	7.00	81.70	6.53	78.17
3	500	17.50	147.70	15.30	137.13
3	750	31.90	219.30	26.73	202.47
4	250	17.33	188.80	19.60	186.17
4	500	42.60	292.10	45.17	287.60
4	750	75.17	425.33	70.93	396.20

Table 5.3: Average set coverage between MOEA/D (A) and MOGLS (B). m is the number of knapsacks and n is the number of items.

Instance		Tchebycheff		Weighted Sum	
m	n	$C(A,B)$	$C(B,A)$	$C(A,B)$	$C(B,A)$
2	250	93.17	3.99	45.06	40.81
2	500	95.45	3.54	55.93	29.47
2	750	97.22	2.21	73.98	15.85
3	250	78.83	7.88	45.22	21.76
3	500	90.97	2.42	61.74	10.09
3	750	98.03	0.39	78.01	4.33
4	250	29.01	26.48	19.34	26.11
4	500	35.98	14.44	25.53	16.41
4	750	52.70	6.49	41.40	8.52

Table 5.4: D-metric values of the solutions found by MOEA/D and MOGLS. The numbers in parentheses represent the standard deviation. m is the number of knapsacks and n is the number of items.

Instance		Tchebycheff		Weighted Sum	
m	n	MOEA/D	MOGLS	MOEA/D	MOGLS
2	250	54.10 (4.57)	96.38 (6.03)	37.17(2.98)	38.18(3.21)
2	500	184.85 (11.88)	328.00 (19.85)	79.07(5.41)	98.63 (9.16)
2	750	437.07 (21.19)	765.66 (44.95)	166.04(13.63)	274.20 (22.70)
3	250	158.71 (6.58)	217.25 (8.45)	97.75(7.23)	141.21 (12.25)
3	500	489.27 (15.91)	701.70(27.40)	270.31 (11.92)	419.15 (23.86)
3	750	960.17 (23.62)	1378.64 (54.63)	446.12(19.12)	768.30 (31.86)
4	250	253.23 (7.17)	301.32 (6.33)	176.52 (7.25)	266.17 (9.27)
4	500	763.96 (14.81)	964.41 (24.85)	431.94(11.59)	725.16 (24.57))
4	750	1546.44 (27.78)	1994.78 (72.84)	761.57 (17.29)	1246.54 (29.16)

by the two algorithms with two different repair methods. Table 5.4 shows the mean and standard deviation of the D-metric values in MOEA/D and MOGLS for each instance.

Figures 5.3-5.5 show the evolution of the average D-metric value of EP from P^* in 30 runs with the number of the calls of the repair method in each algorithm for each test instance. Since the ranges of D-metric values are large, we use the logarithmic scale for the axes of the average D-metric value in Figures 5.3-5.5. Figure 5.6 plots the distributions of EP with the lowest D-metric value found in each algorithm for the three bi-objective instances.

We can make the following remarks:

- With the same number of the calls of a repair method (i.e., the same number of trial solutions), it is evident from Table 5.2 that MOEA/D needs less computational time than MOGLS does. On average, MOEA/D requires about 14% of the CPU time that MOGLS needs. In other words, MOEA/D is seven times as fast as MOGLS. This observation agrees with our analysis of the computational complexity of MOEA/D

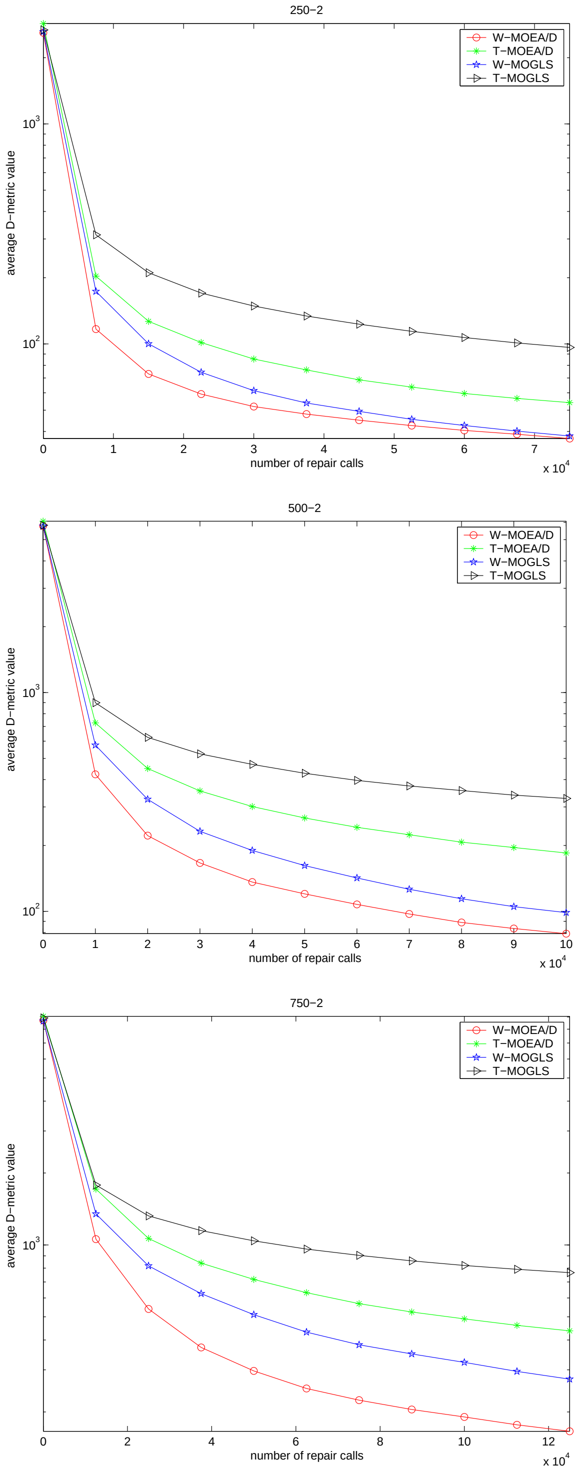


Figure 5.3: The evolution of the average D-metric in MOEA/D and MOGLS for bi-objective three instances. The base 10 logarithmic scale is used for the y-axis in all these figures.

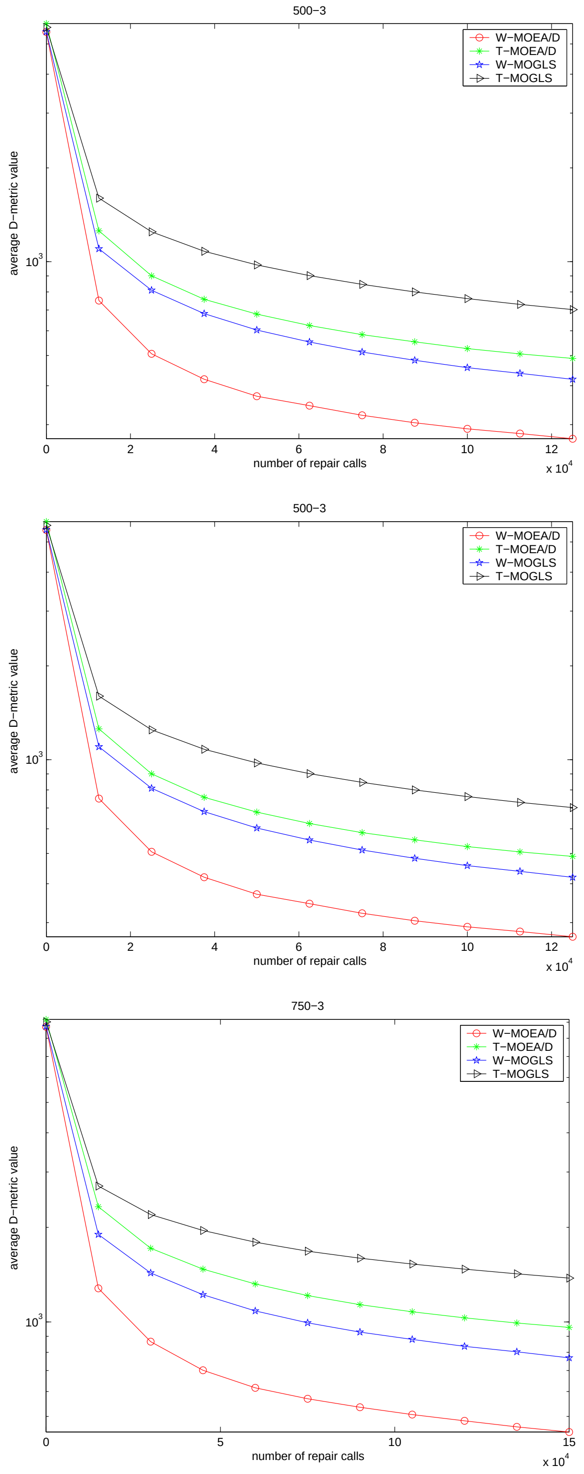


Figure 5.4: The evolution of the average D-metric in MOEA/D and MOGLS for three instances with three objectives. The base 10 logarithmic scale is used for the y-axis in all these figures.

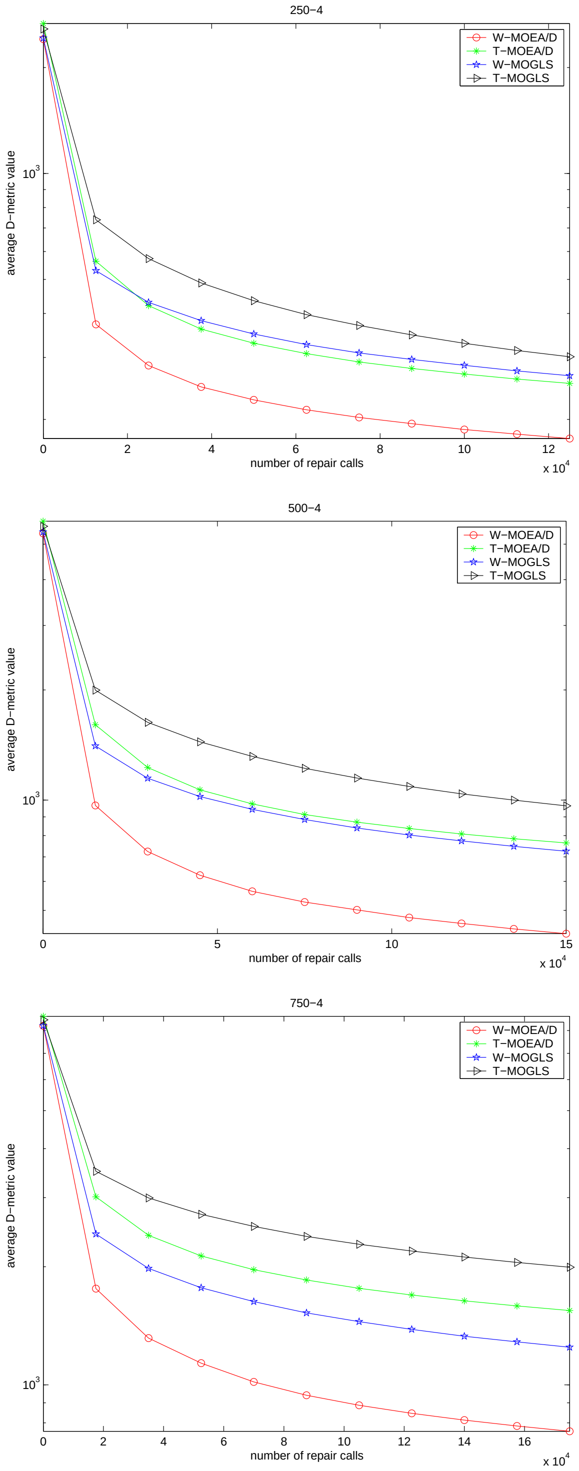


Figure 5.5: The evolution of the average D-metric in MOEA/D and MOGLS for nine instances. The base 10 logarithmic scale is used for the y-axis in all these figures.

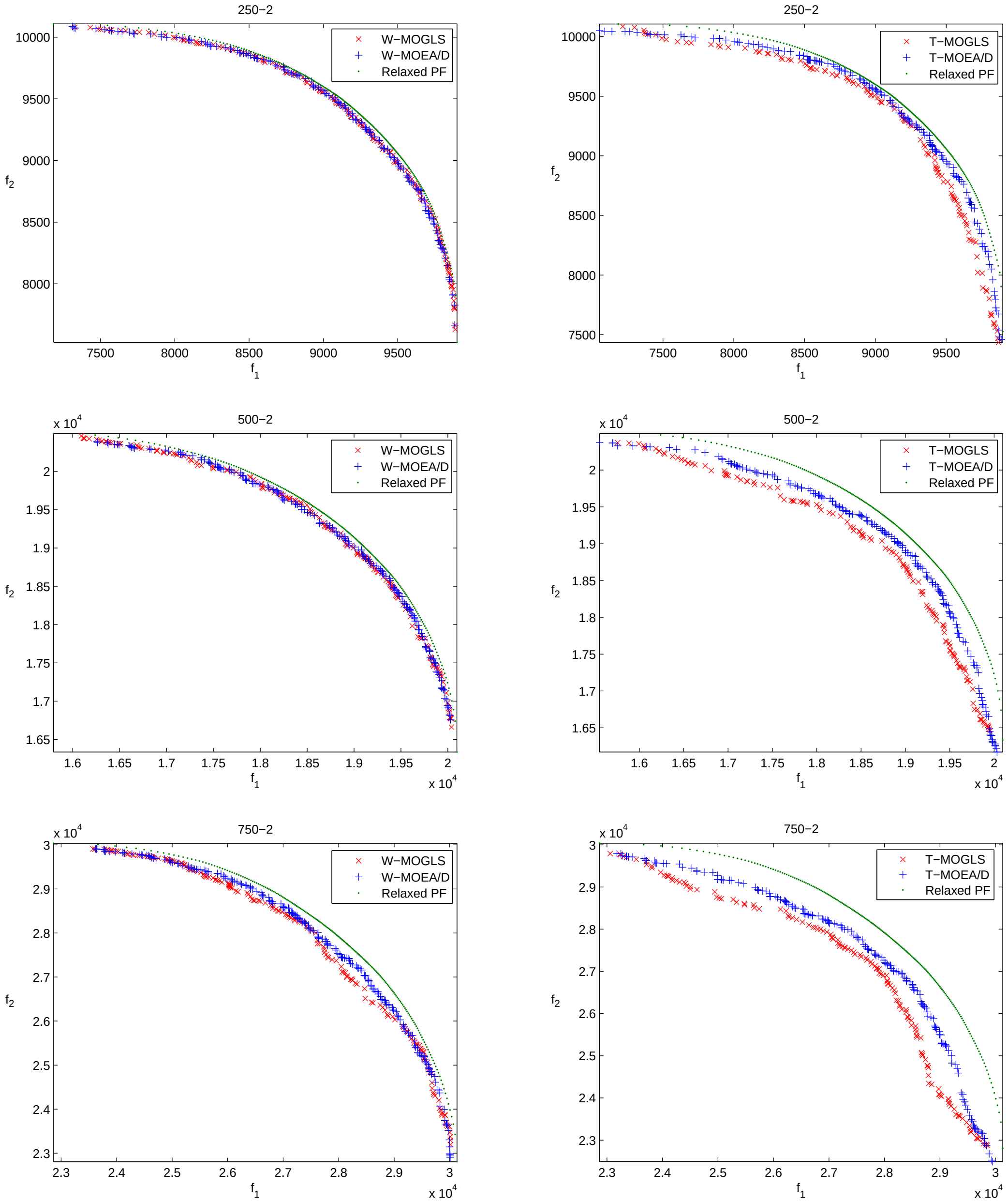


Figure 5.6: Plots of the nondominated solutions with the lowest D-metric in 30 runs of MOEA/D and MOGLS with the weighted sum approach (left) and the Tchebycheff approach for all the 2-objective MOKP test instances.

and MOGLS in Section 5.3.2.

- Figures 5.3-5.5 clearly indicate that for all the test instances, W-MOEA/D (T-MOEA/D) needs fewer calls of a repair method than MOGLS for minimizing the D-metric value, which suggests that MOEA/D is more efficient and effective than MOGLS for the MOKP.

- Tables 5.3 and 5.4 show that the final EP obtained by W-MOEA/D (T-MOEA/D) is better than that obtained by W-MOGLS (T-MOGLS), in terms of both D-metric and C-metric, for all the test instances except instance 250-4 in which W-MOEA/D is slightly worse than W-MOGLS in C-metric. Taking instance 500-3 as an example, on average, 90.97% of the final solutions generated by T-MOGLS are dominated by those generated by M-MOEA/D, and only 2.42% vice versa. The difference between the approximations by T-MOEA/D and T-MOGLS on instances 250-2, 500-2 and 750-2 can be visually detected from Figure 5.6, while difference between W-MOEA/D and W-MOGLS in middle part of the fronts on instance $750-2$ can be spotted from Figure 5.6.

- Table 5.4 also shows that the standard deviation of D-metric in W-MOEA/D (T-MOEA/D) is smaller than that in W-MOGLS (T-MOGLS) for all the instances, which implies that MOEA/D is more stable than MOGLS.

- Table 5.4 and Figure 5.6 reveal that the weighted sum approach outperforms the Tchebycheff approach in both MOEA/D and MOGLS, which suggests that different decomposition approaches in MOEA/D and MOGLS can have different performances.

Overall, we can claim that MOEA/D is computationally much cheaper and can produce better approximations than MOGLS on these MOKP test instances.

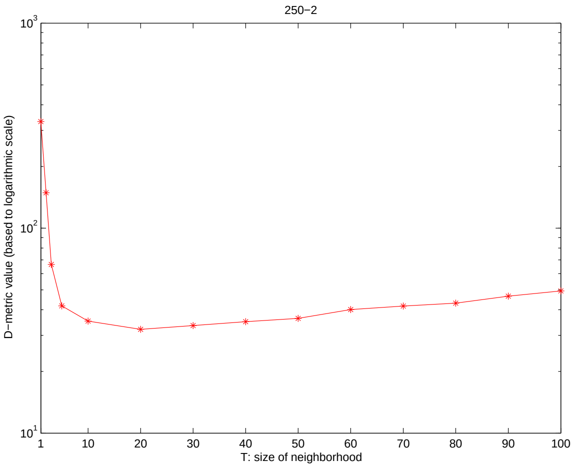


Figure 5.7: The average D-metric value vs. the value of T in MOEA/D for knapsack instance 2-250.

5.4 Sensitivity of Neighborhood Size in MOEA/D

T is a major control parameter in MOEA/D. According to the the discussion in Chapter 4, this parameter is not very sensitive to the performance of MOEA/D for continuous ZDT problem since the Pareto-optimal solutions of these test problems differs only in the values of the first variable. To study the sensitivity of the performance to T in MOEA/D for discrete MOPs, we have tested different settings of T in the implementation of MOEA/D with the weighted sum approach for knapsack instance 250-2. All the parameter settings are the same as in Section 5.3.4, except the settings of T. As clearly shown in Figure 5.7, MOEA/D performs very well with the small T from 10 to 50 on knapsack instance 250-2 and worse with the large T.

Figure 5.7 also reveals that MOEA/D does not work well on both instances when T is very small. This could be due to that, as mentioned in chapter three, MOEA/D with too small T is poor at exploration. The fact that MOEA/D with large T works poorly on knapsack instance 250-2 could be explained by our analysis in Chapter 3 since the solutions to two subproblems with very far different weight vectors are far different in this instance.

5.5 Summary

In this chapter, the version of MOEA/D with local heuristics for solving MOKP was suggested. An advanced repair heuristic was employed to maintain the feasibility of solutions and preserve the quality of resultant feasible solutions. An external population was needed to retain all nondominated solutions found by MOEA/D during the search. MOEA/D was compared with MOGLS on a set of nine knapsack instances. In both algorithms, the weighted sum approach and the weighted Tchebycheff approach were applied. The sensitivity of performance to neighborhood size T was also investigated in MOEA/D.

The experimental results showed that MOEA/D needs much less computational time to find the good approximation of PFs than MOGLS for all test instances. In terms of C-metric and D-metric, the nondominated solutions found by MOEA/D are better than those of MOGLS for all test instances except the instance 250-4. It was also revealed that the weighted sum approach performs better than the weighted Tchebycheff approach in both MOEA/D and MOGLS for all test instances.

The analysis on the sensitivity of neighborhood size T showed that MOEA/D with either too large or too small T performs poorly for the MOKP instance. This result demonstrated that mating restriction does improve the performance of MOEA/D for MOKP with the discrete search space.

Chapter 6

MOEA/D for Continuous MOPs with Nontrivial Pareto Sets

6.1 Introduction

Due largely to the nature of MOEAs, their behaviors and performances are mainly investigated experimentally. Continuous multiobjective test problems are most widely used for this purpose since they are easy to describe and understand. It has been well-understood (as reviewed in [14][33]) that the geometry of the PF, among other characteristics of a MOP, could affect the performance of MOEAs. In fact, a range of PFs such as convex, concave and mixed PFs can be found in commonly-used continuous test problems, which can be used for studying how MOEAs performs with different PF geometries. However, the PS geometry of most existing test problems are often strikingly simple: the PSs of continuous ZDT problems [103] with two objectives are the subsets of line segment defined by

$$\{x|0 \leq x_1 \leq 1, x_2 = x_3 = \ldots = x_n = 0\}$$

and the PSs of the DTLZ test problems [19] with three objectives are the subsets of

$$\{x|0 \leq x_1, x_2 \leq 1, x_3 = x_4 = \ldots = x_n = 0.5\}.$$

There is no reason why a real world problem should have such a simple PS. Observing these oversimplified PSs in existing test problems, Okabe *et al.* first argued the necessity of constructing test problems with complicated PSs and provided a method for controlling the PS shape [69]. They have constructed several test instances with complicated PSs. However, their test instances have two objectives and two decision variables, and they have not generalized their method to the case of many decision variables. Very recently, Deb *et al.* [18] and Huband *et al.* [33] emphasized that variable linkages (i.e., parameter dependencies) should be considered in constructing test problems and proposed using variable transformations for introducing variable linkages. Variable linkages could often complicate PS shapes. However, the PS shapes in the test problems constructed in [18] and [33] are not easy to be directly controlled and described. Moreover, variable linkages are PS shapes are different aspects of MOPs. PS shapes are not the focus of [18] and [33]. It is possible that a continuous MOP with highly variable linkages has a very simple PS. Partially due to lack of test problems, the influence of the PS shapes over the performance of MOEAs has attracted little attention in the evolutionary computation community.

Inspired by the strategies for constructing test problems in [69][18][33] , we have recently proposed several continuous test problems with predefined and nontrivial PS shapes [59], which have also been used in [100] for comparing RM-MEDA with several other MOEAs. The experiments indicate that the complicated PS shapes could cause difficulties for MOEAs. One of the major purposes of this chapter is to extend our previous work and develop a general class of continuous test problems with arbitrary predefined PS shapes for facilitating the study of the ability of MOEAs to deal with the complication of PSs.

The design of efficient MOEAs for the continuous MOPs with nontrivial PS shapes has been rarely studied. Over the past few years, differential evolution algorithms (DEs) have proved to be promising in dealing with the MOPs with variable linkages [77][18]. In this chapter, we develop an efficient version of MOEA/D with DE operators, called DE-MOEA/D. The major difference between DE-MOEA/D and other DE-based MOEAs is that DE-MOEA/D uses decomposition techniques for fitness assignment and mating restriction for sampling child solutions. To encourage the diversity of population, the following new features are taken into consideration in DE-MOEA/D.

- Any individuals in the population can be mated without restriction on their similarity with a low probability;
- The maximal number of solutions replaced by each child solution is limited in order to avoid the loss of the diversity of population;
- Child solutions are sampled by using DE operators and then mutated by polynomial mutation.

The remainder of this chapter is organized as follows. Section 6.2 presents a general formulation for building test problems with prescribed Pareto sets. DE-MOEA/D is then presented in Section 6.3. In the following section, the experimental results on comparing the performance of DE-MOEA/D with two other MOEAs are provided. In section 6.5, the sensitivity of parameters and the effect of reproduction operators in DE-MOEA/D are experimentally investigated. The final section summarizes the chapter.

6.2 Multiobjective Test Problem with Prescribed Pareto Set

In this section, the generic test problems with prescribed Pareto set are presented. The Pareto optimality of the PF and PS of these test problems is then proved. Afterwards, the instantiation of the test problems with prescribed Pareto set is also given.

6.2.1 Generic Test Problem

In our proposed generic continuous test problem, the search space

$$\Omega = \prod_{i=1}^{n}[a_i, b_i] \subset \mathbb{R}^n, \tag{6.1}$$

where $-\infty < a_i < b_i < +\infty$ for all $i = 1, \ldots, n$. Its m objectives to be minimized take the following form:

$$\begin{aligned} f_1(x) &= \alpha_1(x_I) + \beta_1(x_{II} - g(x_I)) \\ &\vdots \\ f_m(x) &= \alpha_m(x_I) + \beta_m(x_{II} - g(x_I)) \end{aligned} \tag{6.2}$$

where

- $x = (x_1, \ldots, x_n)^T \in \Omega$, $x_I = (x_1, \ldots, x_{m-1})^T$ and $x_{II} = (x_m, \ldots, x_n)^T$ are two subvectors of x;
- α_i $(i = 1, \ldots, m)$ are functions from $\prod_{i=1}^{m-1}[a_i, b_i]$ to $\mathbb{R}$;
- β_i $(i = 1, \ldots, m)$ are functions from $\mathbb{R}^{n-m+1}$ to $\mathbb{R}^+$;
- g is a function from $\prod_{i=1}^{m-1}[a_i, b_i]$ to $\prod_{i=m}^{n}[a_i, b_i]$

The following theorem is about the PS and PF of the generic test problem.

Theorem 1 Suppose that

[**i**] $\beta_i(x_I) = 0$ for all $i = 1, \ldots, m$ if and only if $x_I = 0$;

[**ii**] The PS of the following m-objective optimization problem

$$\begin{aligned} &\text{minimize } (\alpha_1(x_I), \ldots, \alpha_m(x_I))^T \\ &\text{subject to } x_I \in \textstyle\prod_{i=1}^{m-1}[a_i, b_i] \end{aligned} \tag{6.3}$$

is $D \in \prod_{i=1}^{m-1}[a_i, b_i]$.

Then the PS of the generic continuous test problem defined by (6.1) and (6.2) is

$$\{(x_I, g(x_I))|x_I \in D\},$$

and its PF is the same as that of (6.3), i.e.

$$\{(\alpha_1(x_I), \dots, \alpha_m(x_I))^T|x_I \in D\}.$$

Proof: It suffices to show that $x = (x_I, x_{II})$ is Pareto-optimal to the test problem defined by (6.1) and (6.2) if and only if $x_I \in D$ and $x_{II} = g(x_I)$. We first prove the "if" part. Let $x = (x_I, x_{II}), y = (y_I, y_{II}) \in \prod_{i=1}^{n}[a_i, b_i]$. If $x_I \in D$ and $x_{II} = g(x_I)$, then, by [i]

$$F(x) = (\alpha_1(x_I), \dots, \alpha_m(x_I))^T$$

It is from [ii] that $F(x)$ cannot be dominated by $(\alpha_1(y_I), \dots, \alpha_m(y_I))$, the latter dominates $F(y)$ since $\beta_i \geq 0$ for all $i = 1, \dots, m$. Therefore, $F(x)$ cannot be dominated by $F(y)$, which implies that x is Pareto-optimal.

Now we prove the "only if" part. Let $x = (x_I, x_{II})$. If $x_I \notin D$, then, by [ii], x_I is not Pareto-optimal to (6.3). Therefore, there exists y_I such that $(\alpha_1(y_I), \dots, \alpha_m(y_I))$ dominates $(\alpha_1(x_I), \dots, \alpha_m(x_I))$. Noting that the latter dominates $F(x)$ and

$$F(y_I, g(y_I)) = (\alpha_1(y_I), \dots, \alpha_m(y_I))^T,$$

we have that $F(y_I, g(y_I))$ dominates $F(x)$. Thus x is not Pareto-optimal.

If $x_{II} \neq g(x_I)$, then, by [i] and negativeness of β_i, $F(x)$ is dominated by $F(x_I, g(x_{II})$. Therefore, x is not Pareto-optimal. This completes the proof of "only if" part.

In the following, we would like to make several comments on the generic test problem defined by (6.1) and (6.2).

- If the dimensionality of D is $m - 1$ as in all the test instances constructed in this paper (in fact, $D = \prod_{i=1}^{n}[a_i, b_i] \subset R^n$ in most of them), the PS and PF of the test problem will be $(m-1)$-D and then exhibit the regularity property.

- The PF and PS of the test problem is determined by α_i and g, respectively. In principle, one can obtain arbitrary $(m-1)$-D PSs and PFs by setting appropriate α_i and g.

- Functions β_i control the difficulty of convergence. If $\sum_{i=1}^{m} \beta_i(x_I)$ has many local minima, then the test problem may have many local Pareto-optimal solutions.

Like DTLZ and WFG test problems [19][33], the proposed test problem uses component functions for defining its PF and introducing multimodality. Its major advantage over others is that the PS can be easily prescribed. A set of fourteen test instances (i.e., F1-F14) generated from (6.1) and (6.2) can be found in **Appendix 2**. These test instances have nontrivial PS shapes, which can be simple or complicated. The instances with simple PS shapes are F1, F8, F10 and F12 while the rest of other instances have complicated PS shapes. Among these test instances, F10, F11 and F12 also have many local nondominated fronts.

6.2.2 Instantiation of Test Instances

In the following, taking two test instances in **Appendix 2** for example, we explain how the generic test problem could be instantiated.

F2 - 2-objective test instance

- The decision space $\Omega = [0,1] \times [-1,1]^{29}$, $x_I = x_1$ and $x_{II} = (x_2, \ldots, x_{30})^T$;

- $\alpha_1(x_1) = x_1$ and $\alpha_2(x_1) = 1 - \sqrt{x_1}$;

- $p(x_1) = (g_2(x_1), \ldots, g_{30}(x_1))^T$ and

$$g_i(x_1) = \sin(6\pi x_1 + \frac{i\pi}{30}), i = 2, \ldots, 30;$$

- β_1 and β_2 are functions from $\mathbb{R}^{29}$ to $\mathbb{R}^+$. Let $y = (y_2, \dots, y_{30})^T \in \mathbb{R}^{29}$,

$$\beta_i(y) = \frac{2.0}{|J_i|} \sum_{j \in J_i} y_j^2, i = 1, 2$$

  where $J_1 = \{j | j \text{ is even and } 2 \leq j \leq 30\}$ and $J_2 = \{j | j \text{ is odd and } 2 \leq j \leq 30\}$.

By Theorem 1, the PF of F2 is:

$$\{(\alpha_1(x_1), \alpha_2(x_1))^T \in \mathbb{R}^2 | 0 \leq x_1 \leq 1\}$$

and its PS:

$$\{(x_1, p(x_1)) \in \mathbb{R}^{30} | 0 \leq x_1 \leq 1\}$$

The projection of the PS in the space of x_1-x_2-x_3 is plotted in Figure 6.1(a).

F9 - test instance with three objectives

- The decision space $\Omega = [0, 1]^2 \times [-2, 2]^8$, $x_I = (x_1, x_2)^T$ and $x_{II} = (x_3, \dots, x_{10})^T$;

- $\alpha_1(x_I) = \cos(0.5x_1\pi)\cos(0.5x_2\pi)$, $\alpha_2(x_I) = \cos(0.5x_1\pi)\sin(0.5x_2\pi)$, and $\alpha_3(x_I) = \sin(0.5x_1\pi)$;

- $p(x_I) = (g_3(x_I), \dots, g_{10}(x_I))^T$ and

$$g_i(x_I) = 2x_2 \sin(2x_1\pi + \frac{i\pi}{10}), i = 3, \dots, 10.$$

- β_1, β_2, and β_3 are functions from $\mathbb{R}^8$ to $\mathbb{R}^+$. Let $y = (y_3, \dots, y_{10})^T \in \mathbb{R}^8$,

$$\beta_i(y) = \frac{2.0}{|J_i|} \sum_{j \in J_i} y_j^2, i = 1, 2, 3$$

  where $J_1 = \{4, 7, 10\}$ $J_2 = \{5, 8\}$ $J_3 = \{3, 6, 9\}$

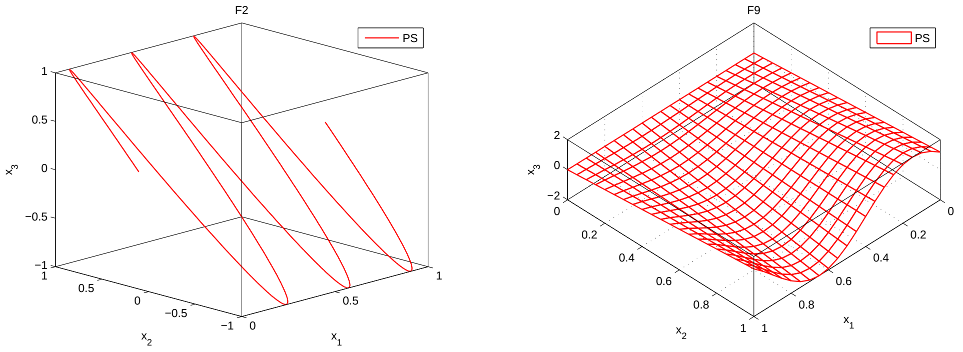
By Theorem 1, the PF of F9 is:

$$\{(\alpha_1(x_I), \alpha_2(x_I), \alpha_3(x_I))^T \in \mathbb{R}^3 | 0 \leq x_1 \leq 1 \text{ and } 0 \leq x_2 \leq 1\}$$

and its PS:

$$\{(x_I, p(x_I)) \in \mathbb{R}^{10} | 0 \leq x_1 \leq 1 \text{ and } 0 \leq x_2 \leq 1\}$$

The projection of the PS in the space of x_1-x_2-x_3 is plotted in Figure 6.1(b).



(a) (b)

Figure 6.1: The Pareto sets of (a) F2 and (b) F9

6.3 DE-MOEA/D

In this section, a DE-based reproduction operator is first introduced. Two new strategies for encouraging diversity in MOEA/D are then suggested. Afterwards, DE-MOEA/D is presented.

6.3.1 DE-based Reproduction Operator

Differential Evolution (DE) developed by R. Storn and K. Price [84] is a simple and efficient population-based approach for global optimization over continuous domains. It has been successfully applied to solve many optimization problems [11][85][68]. Over the past few years, DEs have also received increasing interest for solving MOPs [1][62][93][77][55]. Unlike some other real-valued evolutionary algorithms, DEs use directional information between mating parents to sample child solutions. For example, the reproduction operator in the scheme "DE/rand/1/bin" consists of the following two steps:

- Step 1: randomly select three distinct solutions x^{r_1}, x^{r_2} and x^{r_3}. Set

  $$y = x^{r_1} + F \cdot (x^{r_2} - x^{r_3}) \tag{6.4}$$

  where F is a parameter in $[0, 2]$, which controls the scale of the difference $x^{r_2} - x^{r_3}$.

- Step 2: generate a child solution z by

  $$z_j = \begin{cases} y_j & \text{if } rand < CR \text{ or } j = ID \\ x_j^{r_1} & \text{otherwise} \end{cases}, j = 1, \dots, n \tag{6.5}$$

  where $rand$ is a random number generated from $[0, 1]$ uniformly. ID is a randomly selected index in $\{1, \dots, n\}$. CR is the crossover factor, which controls the probability that the child solution z will come from y.

As mentioned in [76], the length and direction of the difference $x^{r_2} - x^{r_3}$ automatically adapt to the objective function landscape. This is one of the main advantages of DE algorithms.

6.3.2 Diversity Strategies

Like traditional optimization methods in mathematical programming, EAs may have a tendency to converge towards local optima for some problems. To increase the possibility of finding global optima, it is very important to maintain a diverse population of solutions in EAs. In the following, we suggest two new strategies to preserve good diversity of the population in MOEA/D.

- **Strategy 1**: Recombining the individuals without mating restriction with a low probability.

  As mentioned in Chapter 3, the exploration and exploitation in MOEA/D can be controlled by the neighborhood size of subproblems. The larger the neighborhood size is, the more dissimilar the mating parents could be. The better diversity of the population could be provided by using large neighborhood size in MOEA/D. However, it is not easy to tune the neighborhood size dynamically during the evolutionary process of MOEA/D.

  Here, we allow the recombination between the individuals chosen from the whole population without restriction on their similarity with a low probability. In this way, the diverse offspring solutions can be generated due to the dissimilarity between mating parents. To exploit the search space efficiently, more computational efforts still focus on the search with mating restriction in MOEA/D.

- **Strategy 2**: Limiting the maximal number of solutions in the neighborhood replaced by each child solution.

  In the previous two chapters, the maximal number of neighboring solutions replaced by the child solution generated for each subproblem (see step 2.3 in Figure 4.1 and step 2.4 in Figure 5.1) are not limited in MOEA/D. In this case, the majority of current solutions of neighboring subproblems, if not all, might be replaced by the child solution for the current subproblem if it is of high quality. If this occurs, many subproblems may share the same solution. As a result, recombination between the identical mating parents cannot produce diverse child solutions. For this reason, MOEA/D could get trapped in local PFs or converge towards only a part of the PFs. We also practically observed this phenomenon from our experiments on MOEA/D for the ZDT test problems in Chapter 4.

Based on the above analysis, we limit the maximal number of neighboring solutions replaced by each child solution in order to maintain the diversity of mating parents in MOEA/D.

6.3.3 Algorithmic Framework

Taking the above-mentioned two new strategies into consideration, the version of MOEA/D with DE operator, called DE-MOEA/D, is presented. At each generation, DE-MOEA/D maintains:

- a population of N points $x^1, \ldots, x^N \in \Omega$, where x^i is the current solution to the i-th subproblem;
- $FV^1, \ldots, FV^N$, where FV^i is the F-value of x^i, i.e., $FV^i = F(x^i)$ for each $i = 1, \ldots, N$;
- $z = (z_1, \ldots, z_m)^T$, where z_i is the best value found so far for objective f_i;

The details of DE-MOEA/D with the Tchebycheff approach are illustrated in Figure 6.2. Steps 1.1-1.3 are the same as those of MOEA/D in Chapter 4. In step 2.1, the mating parents are selected from the neighborhood of current subproblem with the probability ζ and from the whole population with the probability $(1 - \zeta)$. In step 2.2, child solutions are produced by using DE operators. In step 2.4, each child solution updates at most n_r neighboring solutions during the replacement.

Input: MOP (6.2); a stopping criterion; N : the number of the subproblems; $\lambda^1, \dots, \lambda^N$: a set of N weight vectors; T : the number of weight vectors in the neighborhood of each weight vector; ζ : the probability that parent solutions are selected from the neighborhood of the current solution; n_r : the maximal number of solutions replaced by each child solution.

Output:

Approximation to the PS: $\{x1, \dots, x^N\}$ and approximation to the PF: $\{FV^1, \dots, FV^N\}$.

Step 1 Initialization

Step 1.1 Compute the Euclidean distances between any two weight vectors and then work out the T closest weight vectors to each weight vector. For each $i = 1, \dots, N$, set $B(i) = \{i_1, \dots, i_T\}$ where $\lambda^{i_1}, \dots, \lambda^{i_T}$ are the T closest weight vectors to λ^i.

Step 1.2 Generate an initial population $x^1, \dots, x^N$ by uniformly randomly sampling from Ω. Set $FV^i = F(x^i)$.

Step 1.3 Initialize $z = (z_1, \dots, z_m)$ by setting $z_j = \min_{1 \le i \le N} f_j(x^i)$.

Step 2 Update

For $i = 1, \dots, N$, do

Step 2.1 Selection of Mating/Update Range: Uniformly randomly generate a number a from $(0, 1)$.

$$P = \begin{cases} B(i) & \text{if } a < \zeta, \\ \{1, \dots, N\} & \text{otherwise.} \end{cases}$$

Step 2.2 Reproduction: Randomly select two indexes k, l from P, and then generate a new solution y from x^k and x^l by using DE operator and polynomial mutation.

Step 2.3 Update of z: For each $j = 1, \dots, m$, if $z_j > f_j(y)$, then set $z_j = f_j(y)$.

Step 2.4 Update of Solutions : Set $c = 0$ and then do the following:

(1) If $c = n_r$ or P is empty, go to Step 3. Otherwise, randomly pick an index j from P.

(2) if $g^{te}(y|\lambda^j, z) \le g^{te}(x^j|\lambda^j, z)$, then set $x^j = y$, $FV^j = F(y)$ and $c = c + 1$.

(3) Remove j from P and go to (1).

Step 3 Stopping Criteria If stopping criteria is satisfied, then stop and output $\{x^1, \dots, x^N\}$ and $\{FV^1, \dots, FV^N\}$. Otherwise go to **Step 2**.

Figure 6.2: The algorithmic framework of DE-MOEA/D

6.4 Computational Experiments

6.4.1 Algorithms in Comparison

In this section, we choose two other efficient algorithms: DE-based NSGA-II and RM-MEDA-II (the improved version of RM-MEDA [100]) in the comparison with DE-MOEA/D for the test instances in **Appendix 2**. These two algorithms are briefly described as follows.

DE-NSGA-II

The combinations of NSGA-II with DE algorithms have been studied by several researchers [36][35]. In this chapter, we consider the modified version of DE-based NSGA-II proposed in [36], where new solutions are sampled by using both DE reproduction operator and polynomial mutation. We call this algorithm DE-NSGA-II in our work. In DE-NSGA-II, the nondominated sorting procedure and crowding density estimation are applied to rank all solutions in the union of offspring population and current population, which are the same as those in NSGA-II [16]. $(\mu + \lambda)$-replacement strategy is used to determine the solutions in the next generation.

RM-MEDA-II

In [100], a regularity model-based multiobjective estimation of distribution algorithm, called RM-MEDA, is proposed. Unlike many other MOEAs, it samples new solutions according to the probability distribution of promising solutions, whose centroid is a (m-1)-D manifold. The model is built by using the local principal component algorithm. RM-MEDA has shown promising performance on a set of multiobjective test problems with variable linkages.

Very recently, the authors of RM-MEDA have further developed an improved version of RM-MEDA [101], which has not been officially published. In this algorithm, a new selection operator for fitness assignment is suggested by considering the regularity of MOPs in the objective space. Local models are built for sampling child solutions. The details of the improved RM-MEDA can be referred to the technical report [101]. This algorithm is called RM-MEDA-II in this chapter. After testing both versions of RM-MEDA on the test instances suggested in this chapter, we have found that RM-MEDA-II performed better than its previous version.

6.4.2 Experimental Settings

We have implemented both DE-MOEA/D and DE-NSGA-II in C++. The experimental results of RM-MEDA-II reported in this chapter were provided by the authors of RM-MEDA. The experimental setting is as follows.

- **Dimensionality of variables** The number of variables in F1-F7, F13 and F14 is set to 30 while that in F8-F12 is set to 10.
- **Population Size:** The population size N is set to 300 for all 2-objective test instances and 595 for all 3-objective test instances. The weight vectors in MOEA/D are generated in the same way as that in Chapter 4. The parameter H for generating weight vectors is set to 299 for 2-objective test instances and 33 for 3-objective test instances.
- **Reproduction Operators:** The DE operators in (6.4) and (6.5) and polynomial mutation are used to generate offspring solutions in both DE-MOEA/D and DE-NSGA-II. The scaling factor F in DE operator is 0.5 and the crossover rate CR is 1.0 for all test instances. The distribution index in polynomial mutation is 20 and the

mutation rate is $1.0/n$.

- **Maximal number of generations:** In all three algorithms, the maximal number of generations is set to 250 for F1 and F8, and 500 for the rest of test instances.
- **Number of runs:** Each algorithm is run 20 times independently for each test instance.
- **Parameters of DE-MOEA/D:** The neighborhood size T, the maximal number of solutions replaced by each child solution n_r, and the probability of selecting solution from neighborhood ζ, are set to 20, 2, and 90% respectively.
- **Parameters of RM-MEDA-II:** The number of clusters is set to 10 for F1 and 25 for the rest of test instances.
- **Performance metric:** Both D-metric and C-metric are used to compare the performance of three algorithms. To compute D-metric values, a set of 500 and 990 points in the PF with even spread are chosen as a reference set P^* for the test instances with two objectives and three objectives respectively.

6.4.3 Experimental Results

In this section, the comparison of DE-MOEA/D with DE-NSGA-II and RM-MEDA-II on a set of fourteen test instances F1-F14 is performed[1].

Test Instances with Simple PS Shapes

In this section, we compare DE-MOEA/D with DE-NSGA-II and RM-MEDA-II on F1 and F8, which have simple PS shapes in the decision space.

[1] Since the current version of RM-MEDA-II is unable to deal with the test instances with three objectives, we do not provide the comparison of MOEA/D with RM-MEDA-II for 3-objective test instances in this chapter.

Table 6.1: The D-metric values of the nondominated solutions found by DE-MOEA/D, RM-MEDA-II and DE-NSGA-II on F1 and F8

GD-value	DE-MOEA/D			RM-MEDA-II			DE-NSGA-II		
Instance	mean	min	std	mean	min	std	mean	min	std
F1	0.0015	0.0015	0.0000	0.0031	0.0022	0.0005	0.0044	0.0040	0.0002
F8	0.0271	0.0265	0.0003	N/A	N/A	N/A	0.0372	0.0362	0.0007

Table 6.2: Average set coverage among DE-MOEA/D(D), RM-MEDA-II(R) and DE-NSGA-II (N)

Instance	C(D, R)	C(R,D)	C(D, N)	C(N,D)	C(R, N)	C(N,R)
F1	0.24	77.44	0.02	94.88	4.93	0.37
F8	5.82	51.19	N/A	N/A	N/A	N/A

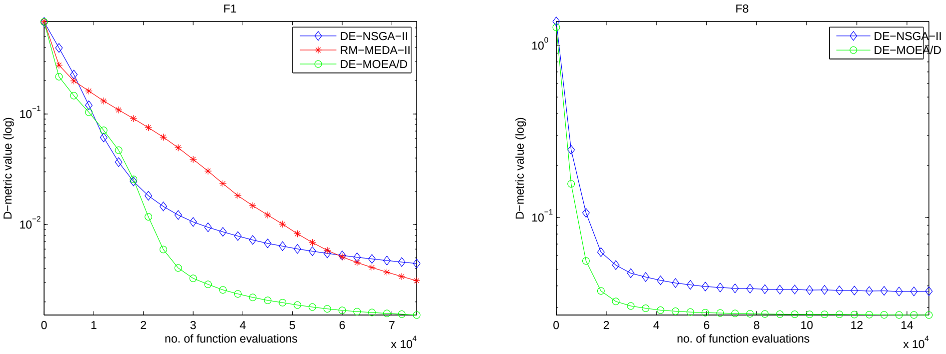
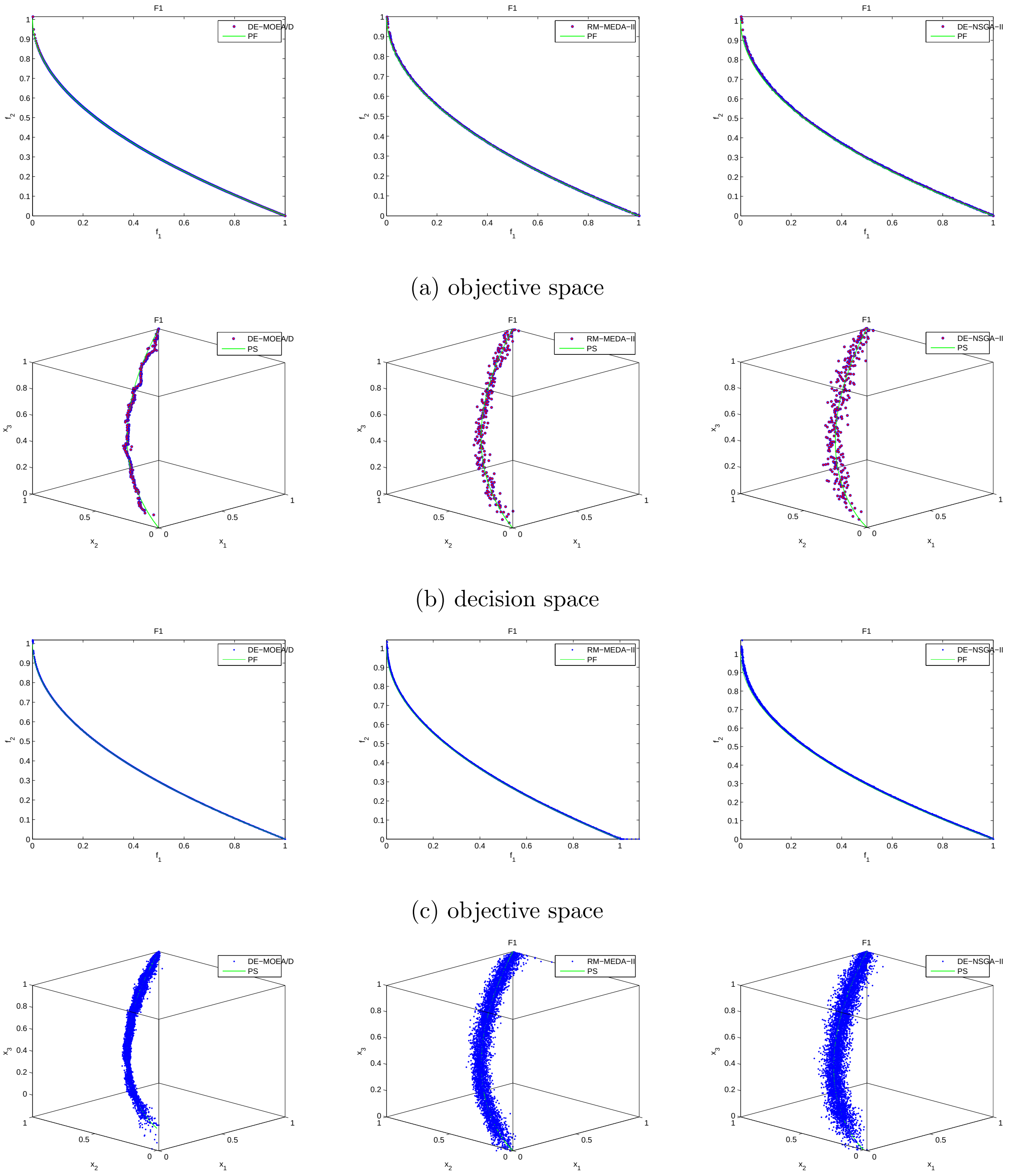


Figure 6.3: The evolution of D-metric values found by three algorithms on F1 and F8



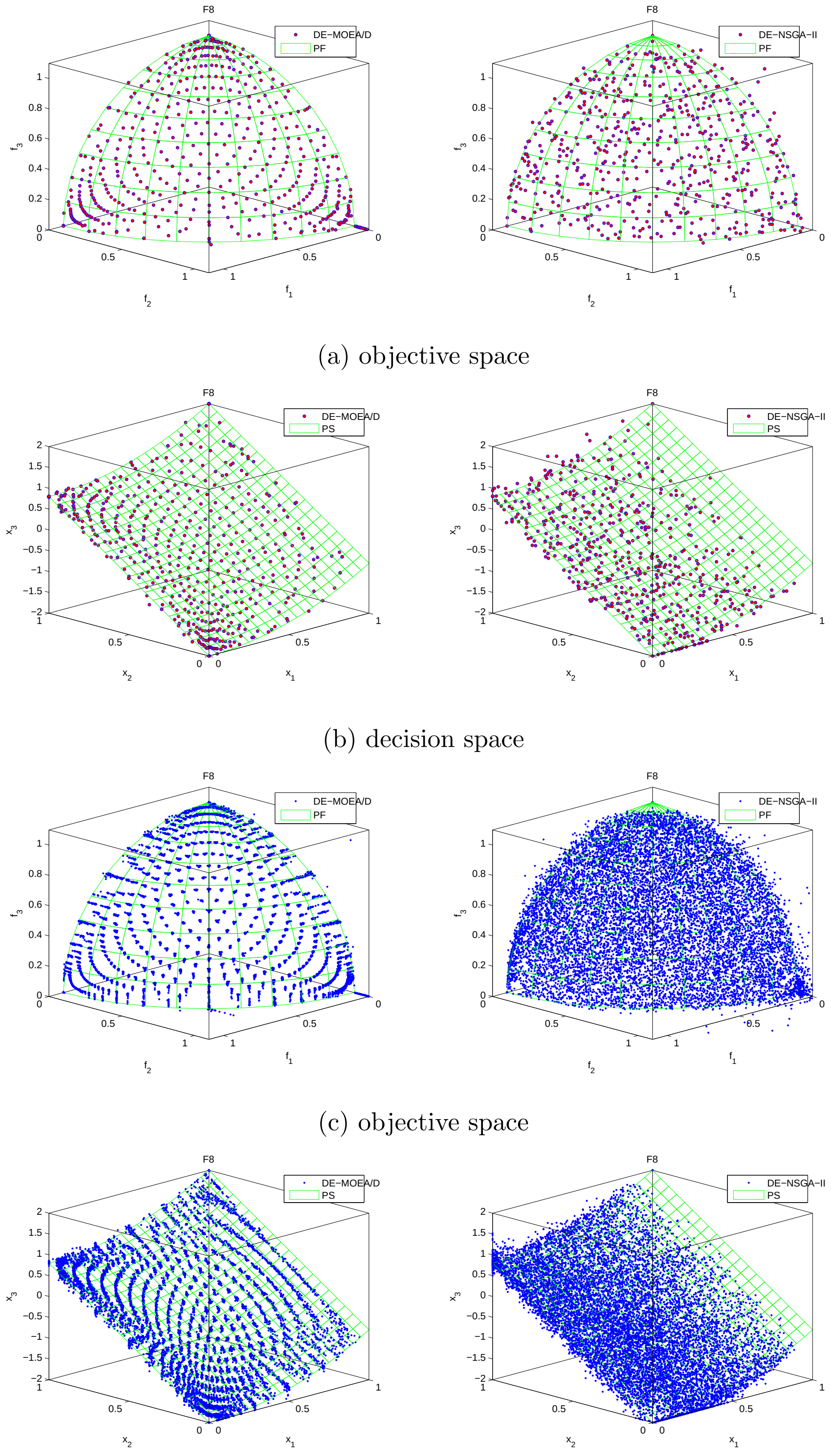
(a) objective space

(b) decision space

(c) objective space

(d) decision space

Figure 6.4: Plots of the solutions obtained by the run with the lowest D-metric values ((a) and (b)) and 20 runs ((c) and (d)) of each algorithm on F1.



(a) objective space

(b) decision space

(c) objective space

(d) decision space

Figure 6.5: Plots of the solutions obtained by the run with the lowest D-metric value ((a) and (b)) and 20 runs ((c) and (d)) of each algorithm on F8.

Table 6.1 gives that the minimal, mean and standard deviation of D-metric values in DE-MOEA/D, RM-MEDA-II and DE-NSGA-II for F1 and F8. Table 6.2 shows the mean of the C-metric values of the final nondominated fronts obtained by three algorithms. ”N/A” in both tables means not applicable.

From the experimental results in Tables 6.1 and 6.2, DE-MOEA/D clearly performs better than the other two algorithms on F1 and is superior to DE-NSGA-II on F8 in terms of both D-metric and C-metric values. It can also be seen from Table 6.1 that all the three algorithms find the mean of D-metric values below 0.005 on F1. Empirically, this result indicates that the final nondominated solutions found by all the three algorithms for F1 approximate the PF very well in a sense.

Figure 6.3 presents the evolution of the average D-metric values of the current population to P^* with the number of function evaluations in each algorithm on F1 and F8. From Figure 6.3, it is evident that DE-MOEA/D converges much faster than the other two algorithms on F1 and DE-NSGA-II on F8 in terms of D-metric values.

Figures 6.4 and 6.5 plot the nondominated solutions found by each algorithm on F1 and F8 in the run with the lowest D-metric value both in the objective space and in the decision space. To view the variance of the nondominated solutions found by each algorithm, all the 20 nondominated fronts found are also plotted in Figures 6.4 and 6.5.

It can be observed from Figures 6.4 and 6.5 that all the three algorithms find the good approximation of PF for F1 while both DE-MOEA/D and DE-NSGA-II approximate the PF of F8 well. Figure 6.4 also shows that the nondominated solutions found by DE-MOEA/D in the best run are closer to the PS of F1 than those of other two algorithms. As to the uniformness of the nondominated solutions in the objective space, DE-MOEA/D clearly performs better than DE-NSGA-II on F8.

As we can observe from Figures 6.4 and 6.5, the PS shapes of F1 and F8 are

Table 6.3: The D-metric values of the nondominated solutions found by DE-MOEA/D, RM-MEDA-II and DE-NSGA-II on F2-F7 and F9

GD-value	DE-MOEA/D			RM-MEDA-II			DE-NSGA-II		
Instance	mean	min	std	mean	min	std	mean	min	std
F2	0.0028	0.0023	0.0004	0.0318	0.0227	0.0060	0.0349	0.0203	0.0066
F3	0.0068	0.0022	0.0099	0.0142	0.0071	0.0131	0.0296	0.0228	0.0030
F4	0.0040	0.0025	0.0014	0.0059	0.0028	0.0026	0.0288	0.0251	0.0021
F5	0.0022	0.0019	0.0001	0.0092	0.0055	0.0040	0.0270	0.0227	0.0015
F6	0.0127	0.0073	0.0069	0.0169	0.0109	0.0103	0.0288	0.0244	0.0031
F7	0.0260	0.0060	0.0278	0.0811	0.0551	0.0343	0.0515	0.0344	0.0066
F9	0.0289	0.0276	0.0014	N/A	N/A	N/A	0.0680	0.0522	0.0072

simple curve or surface. As mentioned in [100], both DE-based MOEAs and RM-MEDA have the ability to solve continuous MOPs with nonlinear variable linkages, of which the PS geometries are also simple curves. The reason is that the child solutions produced by DE operators or regularity-based sampling method are very likely to be around the PS when mating parents are close to the PS.

Test Instances with Complicated PS Shapes

In this section, we compare MOEA/D with RM-MEDA-II and DE-NSGA-II on seven test instances - F2, F3, F4, F5, F6, F7 and F9, which have complicated PS shapes. The PFs of F2-F7 are the same as that of F1 while the PF of F9 is the same as that of F8.

Table 6.3 presents the minimal, mean and standard deviation of the D-metric values found by all three algorithms. It is evident from this table that the final solutions obtained by DE-MOEA/D are better than those in the other two algorithms for all 2-objective test instances in terms of the D-metric. For the 3-objective test instance F9, DE-MOEA/D also performs better than DE-NSGA-II in minimizing the D-metric values.

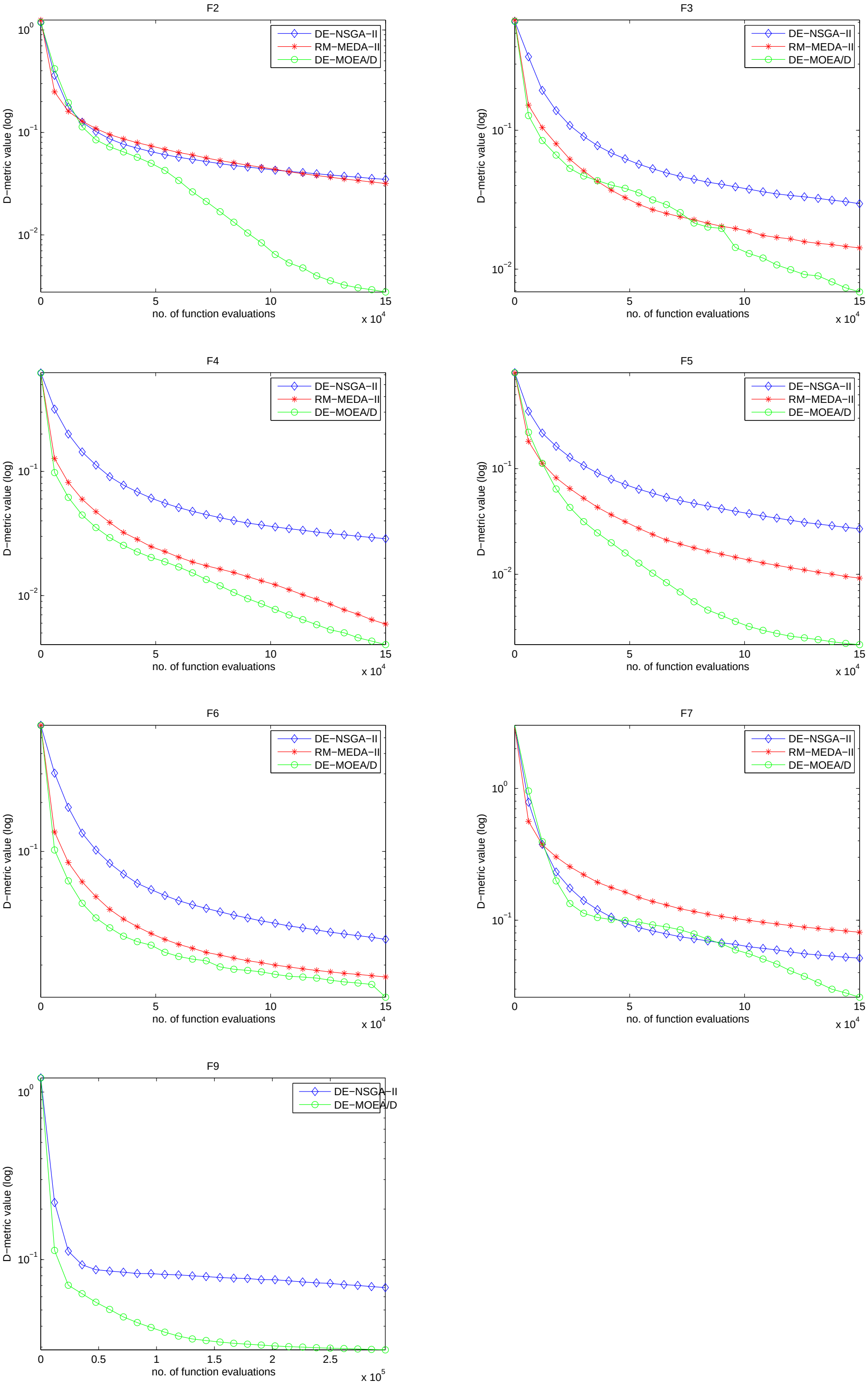
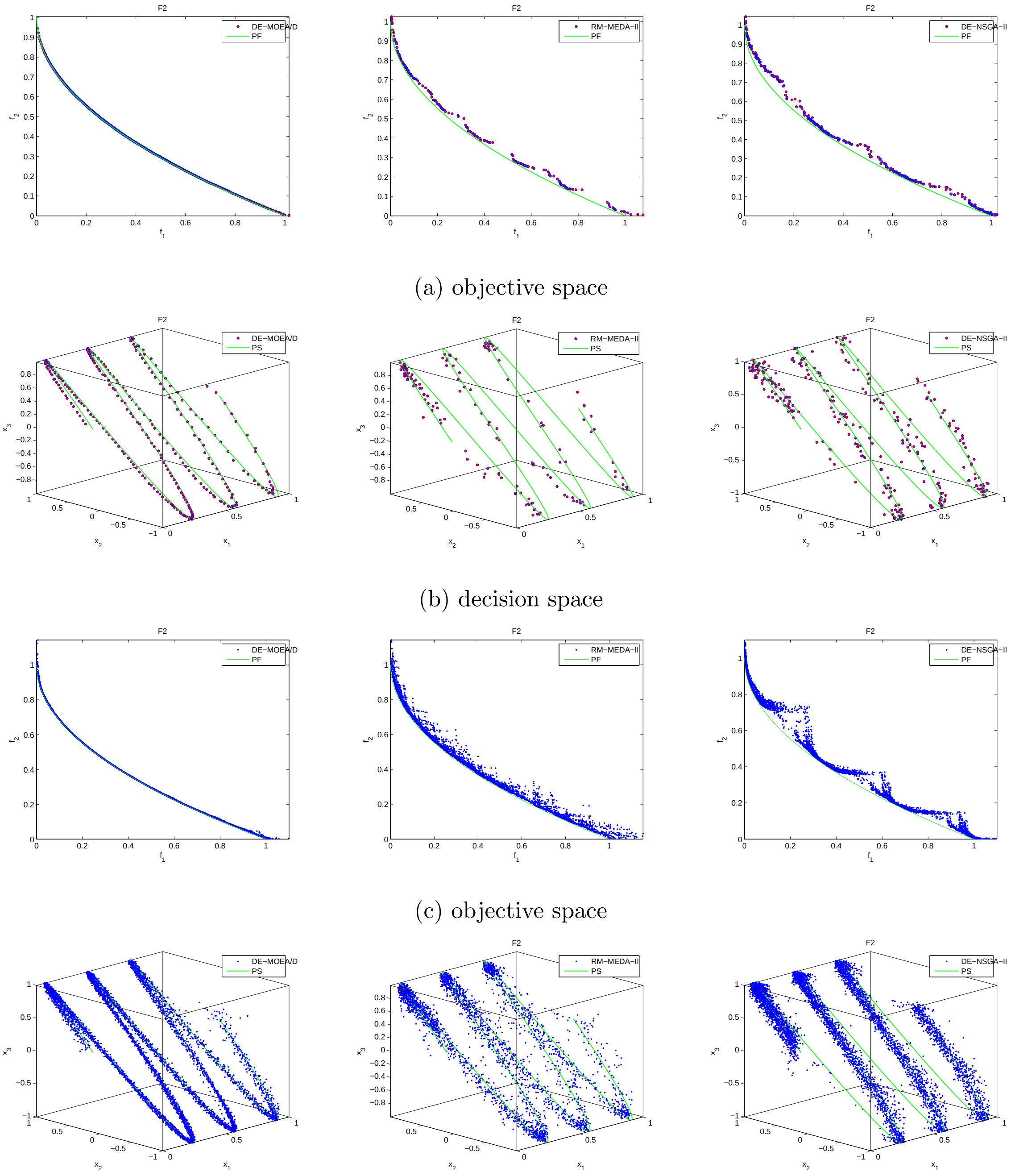


Figure 6.6: The evolution of D-metric values found by each algorithm on F2, F3, F4, F5, F6, F7 and F9



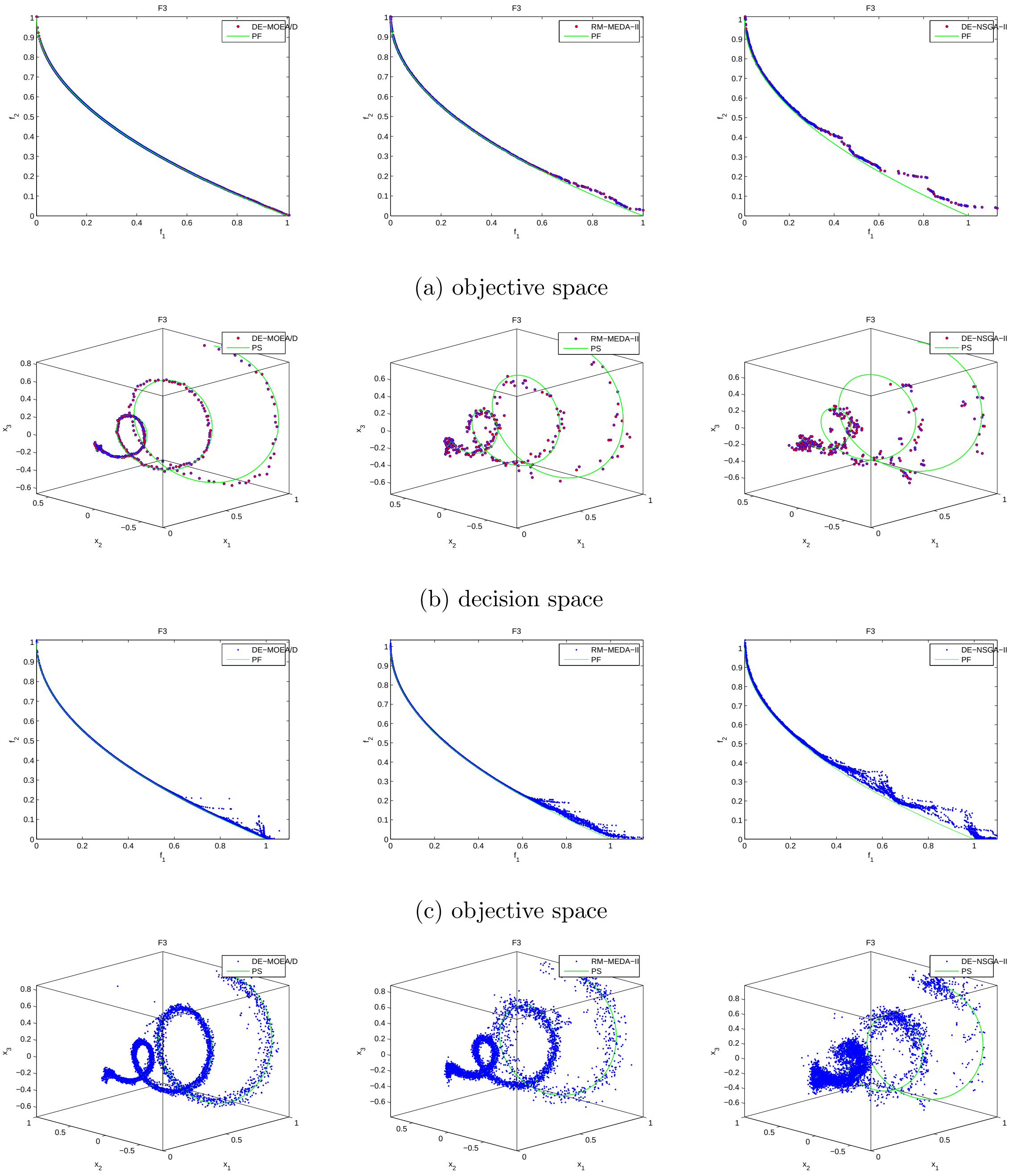
(a) objective space

(b) decision space

(c) objective space

(d) decision space

Figure 6.7: Plots of the solutions obtained by the run with the lowest D-metric value ((a) and (b)) and 20 runs ((c) and (d)) of each algorithm on F2.



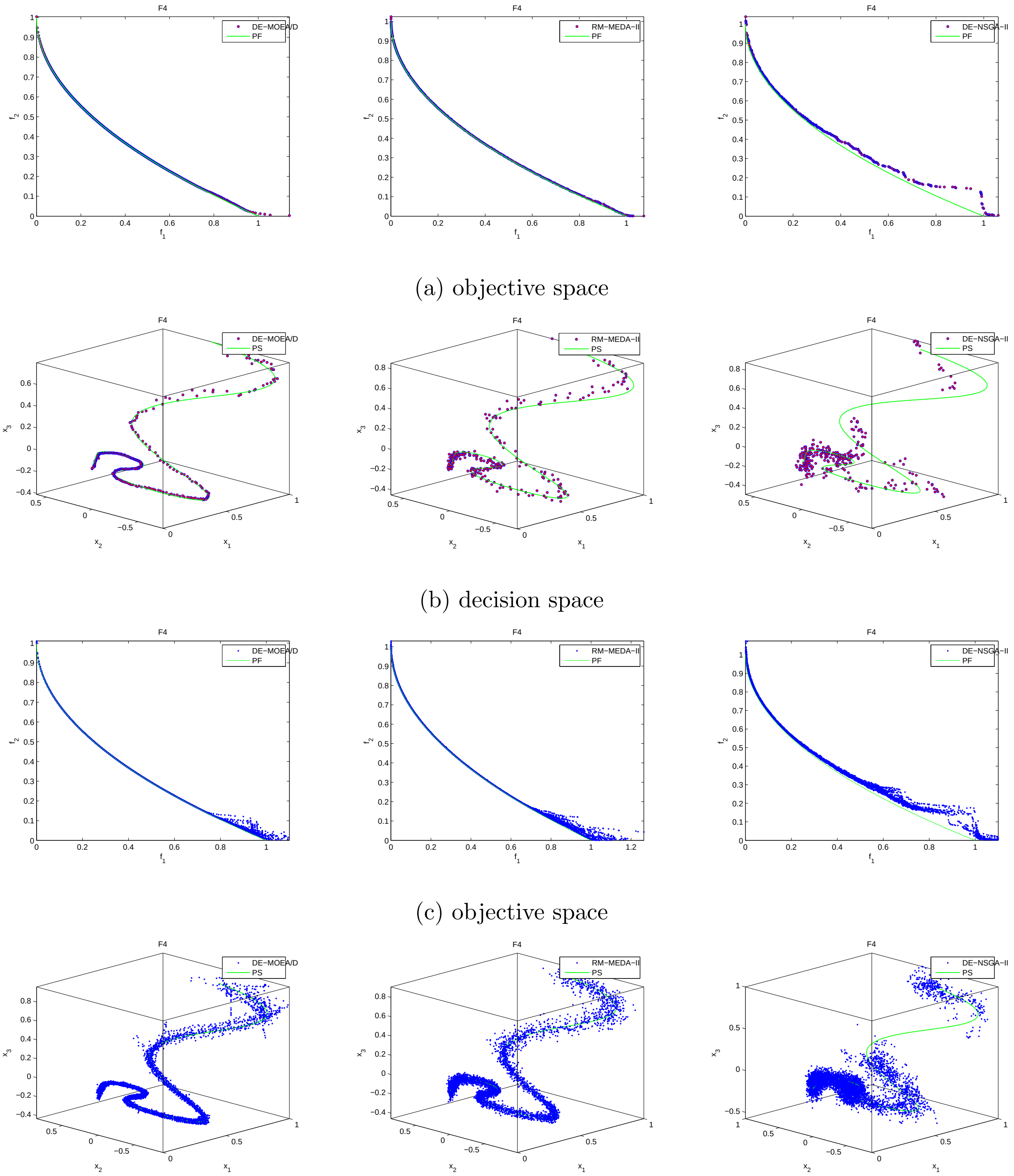
(a) objective space

(b) decision space

(c) objective space

(d) decision space

Figure 6.8: Plots of the solutions obtained by the run with the lowest D-metric value ((a) and (b)) and 20 runs ((c) and (d)) of each algorithm on F3.

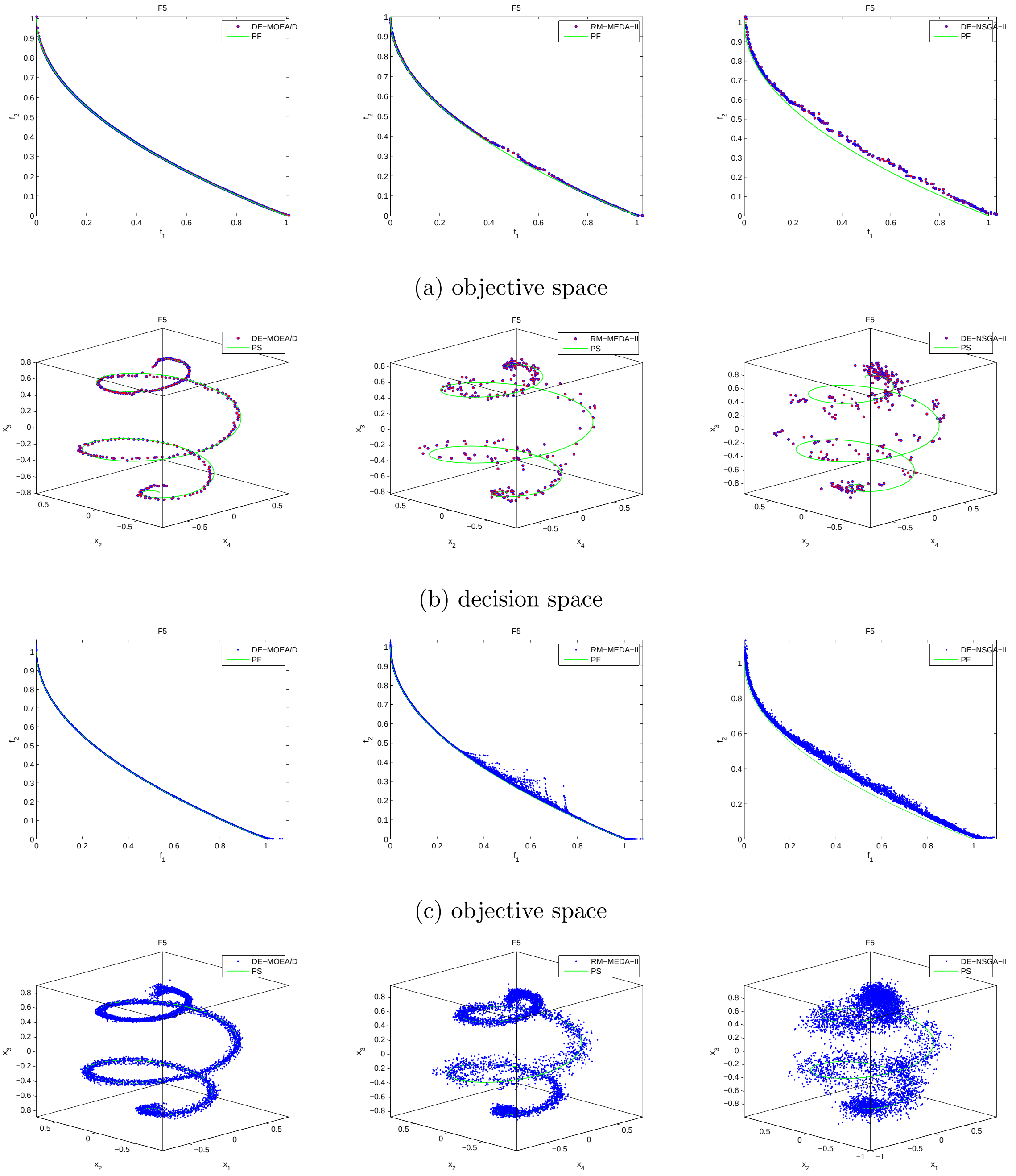

(a) objective space

(b) decision space

(c) objective space

(d) decision space

Figure 6.9: Plots of the solutions obtained by the run with the lowest D-metric value ((a) and (b)) and 20 runs ((c) and (d)) of each algorithm on F4.



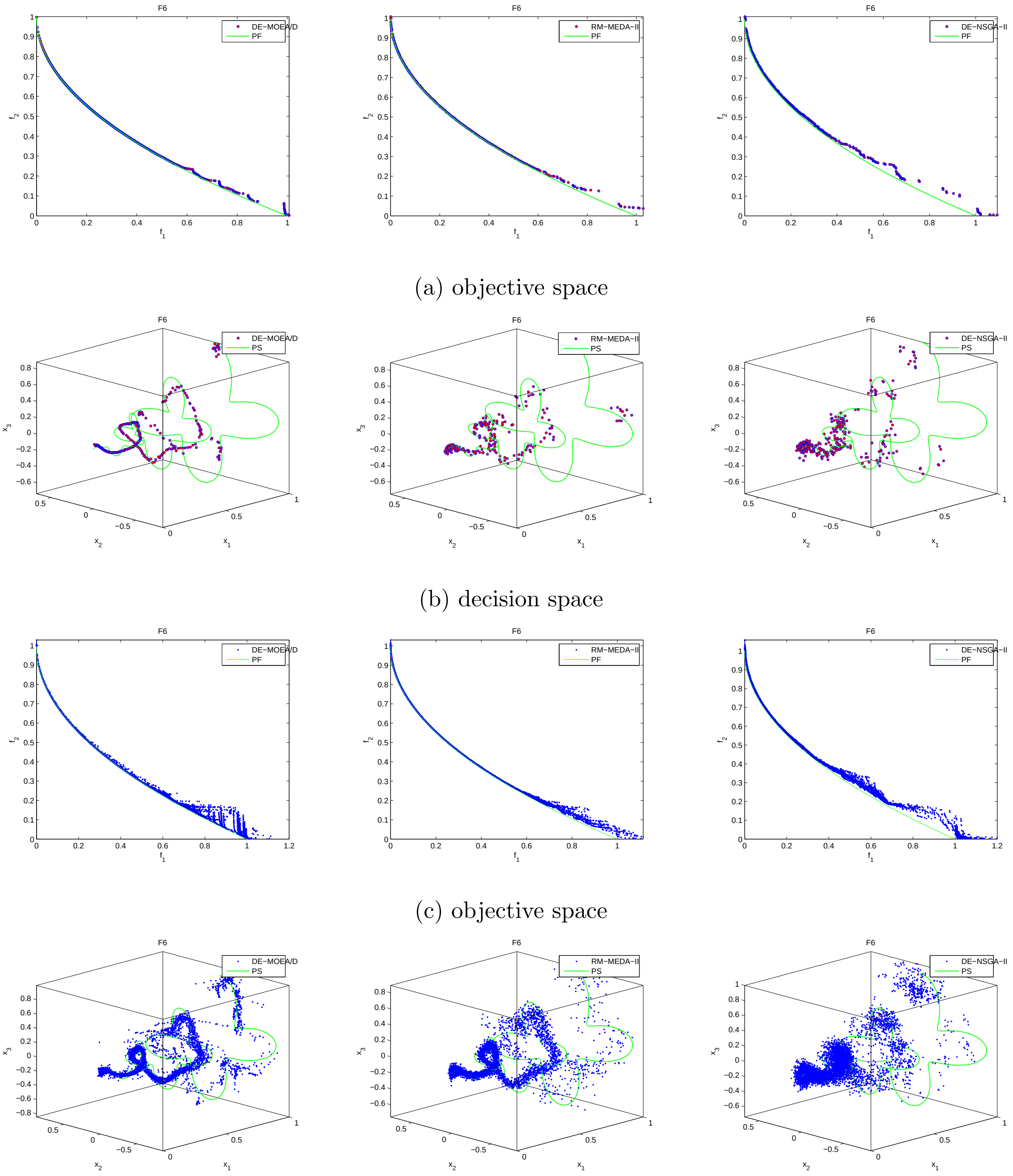
(a) objective space

(b) decision space

(c) objective space

(d) decision space

Figure 6.10: Plots of the solutions obtained by the run with the lowest D-metric value ((a) and (b)) and 20 runs ((c) and (d)) of each algorithm on F5.



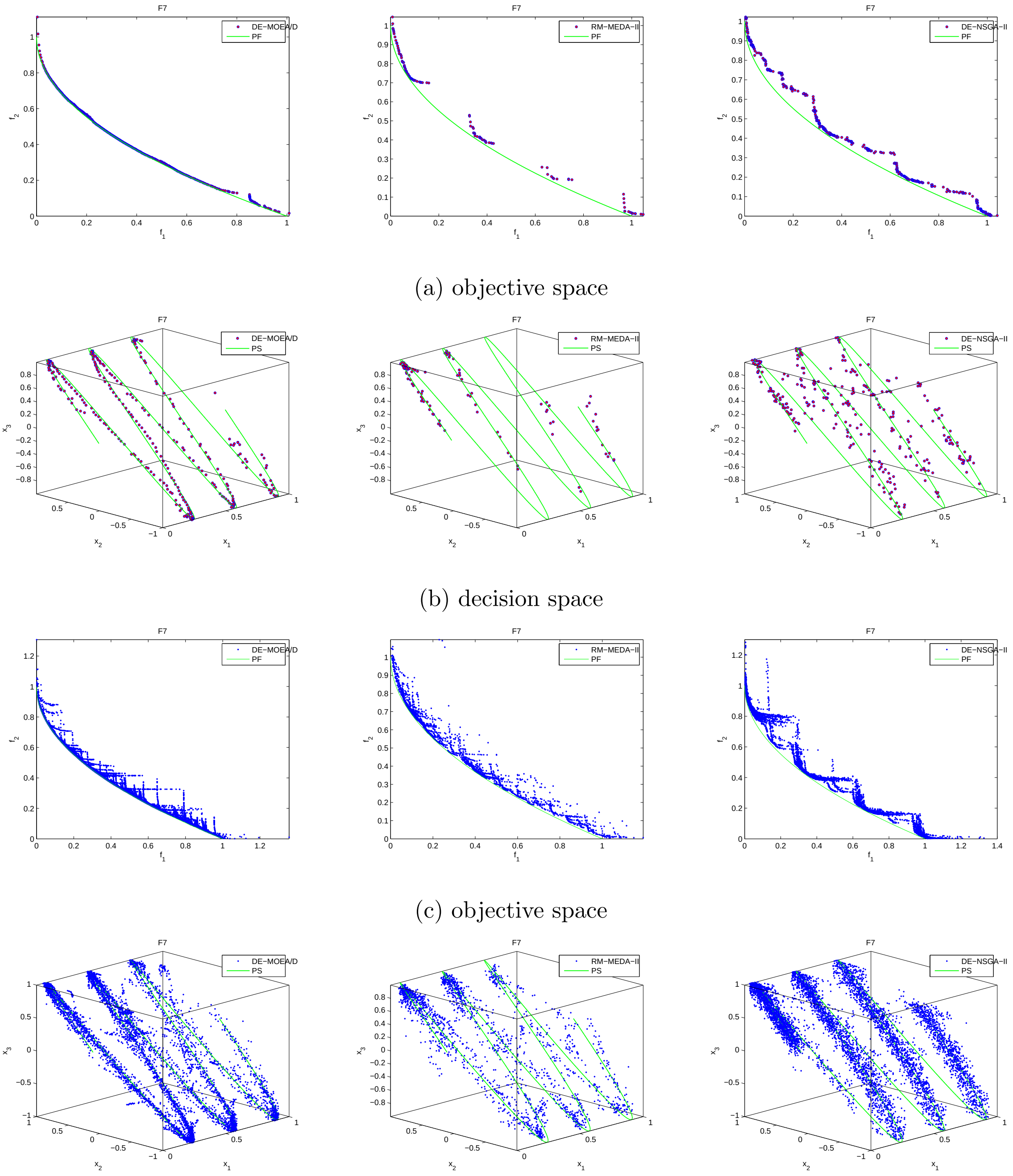
(a) objective space

(b) decision space

(c) objective space

(d) decision space

Figure 6.11: Plots of the solutions obtained by the run with the lowest D-metric value ((a) and (b)) and 20 runs ((c) and (d)) of each algorithm on F6.



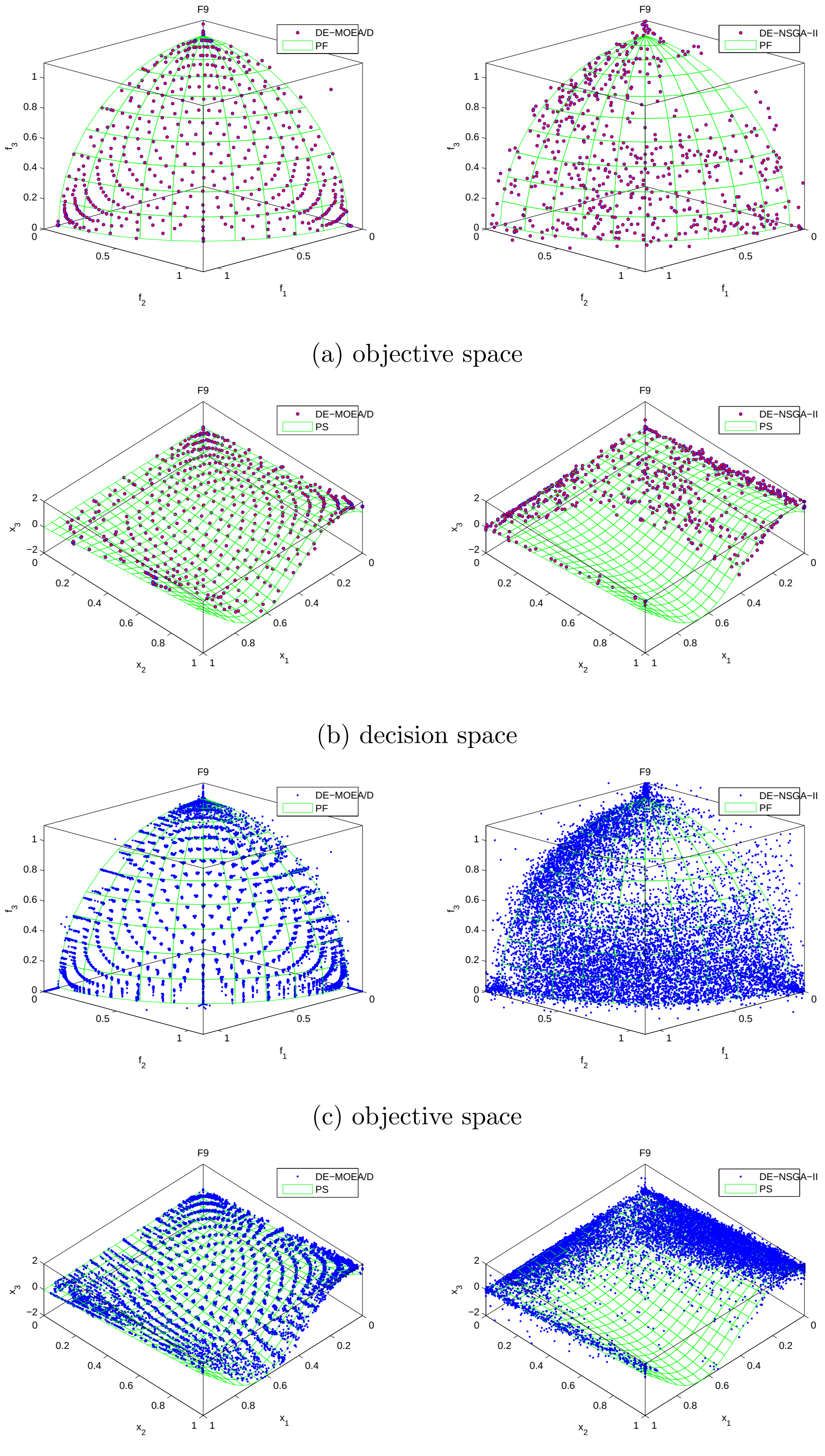
(a) objective space

(b) decision space

(c) objective space

(d) decision space

Figure 6.12: Plots of the solutions obtained by the run with the lowest D-metric value ((a) and (b)) and 20 runs ((c) and (d)) of each algorithm on F7.



(a) objective space

(b) decision space

(c) objective space

(d) decision space

Figure 6.13: Plots of the solutions obtained by the run with the lowest D-metric value ((a) and (b)) and 20 runs ((c) and (d)) of each algorithm on F9.

Table 6.4: Average set coverage among DE-MOEA/D(D), RM-MEDA-II(R), and DE-NSGA-II (N) on F2-F7 and F9

Instance	C(D, R)	C(R,D)	C(D, N)	C(N,D)	C(R, N)	C(N,R)
F2	0.15	98.45	1.04	91.38	33.51	0.47
F3	0.13	52.02	0.53	97.16	2.58	0.65
F4	0.16	51.22	1.34	97.13	1.91	0.73
F5	0.35	55.62	0.00	99.93	1.46	1.28
F6	0.15	41.50	2.14	95.33	1.49	0.60
F7	0.08	67.18	13.22	64.78	28.09	0.35
F9	0.13	62.40	N/A	N/A	N/A	N/A

For F4, both DE-MOEA/D and RM-MEDA-II perform similarly. It is also clear that DE-NSGA-II has the worst performance in minimizing D-metric values for all 2-objective test instances.

Table 6.4 shows that in terms of the C-metric, the nondominated solutions found by DE-MOEA/D are better than those obtained by the other two algorithms for all seven test instances. This table shows that more than 90% of the final solutions found by DE-NSGA-II are dominated by those of DE-MOEA/D except F7. On average, more than 50% of the final solutions obtained by RM-MEDA-II are dominated by those of DE-MOEA/D except F6. It can also be seen from Table 6.4 that RM-MEDA-II performs slightly worse than DE-NSGA-II on all test problems considered.

Figure 6.6 presents the evolution of the average D-metric values of the nondominated solutions in the current population. These experimental results show that DE-MOEA/D outperforms the other two algorithms in converging towards the PFs of F2-F7 and F9. That is, DE-MOEA/D converges much faster than the other two algorithms in terms of D-metric.

Figures 6.7-6.13 show the nondominated solutions found by each algorithm on each test instance. From these figures, it is very clear that DE-MOEA/D performs much better

Table 6.5: The D-metric values of the nondominated solutions found by DE-MOEA/D, RM-MEDA-II and DE-NSGA-II on F10, F11 and F12

GD-value	DE-MOEA/D			RM-MEDA-II			DE-NSGA-II		
Instance	mean	min	std	mean	min	std	mean	min	std
F10	0.0049	0.0015	0.0063	0.0187	0.0049	0.0186	0.1171	0.0270	0.0716
F11	0.1445	0.0596	0.0842	0.9390	0.8117	0.0847	0.2606	0.1599	0.0832
F12	0.0998	0.0487	0.0429	0.2717	0.0962	0.0860	0.1981	0.1191	0.0494

than the other two algorithms on F2 and F7. The experimental results in Figures 6.8-6.10 indicate that DE-MOEA/D and RM-MEDA-II perform similarly on F3, F4 and F5 while DE-NSGA-II performs the worst on these test instances. As seen from Figure 6.11, none of the three algorithms approximate the whole PF and PS of F6 well. Compared with other test instances, the PS of F6 is overcomplicated. Therefore, it is harder than all other test instances. Although both F2 and F7 have the same PF and PS, DE-MOEA/D has the worse performance on F7. The reason is that the scales of β functions in F7 are larger than those of F2. The larger the scales of β functions, the bigger the distance between the initial population and the true PF.

It can be seen from Figure 6.13 that DE-MOEA/D outperforms DE-NSGA-II in diversifying the nondominated solutions both in the objective space and in the decision space on the 3-objective test instance - F9. The distribution of nondominated solutions found by DE-NSGA-II plotted in Figure 6.13 shows that DE-NSGA-II fails to find the whole PS of F9. Compared with results in Figure 6.5, F9 is harder than F8 since the PS of F9 is a more complicated surface in the decision space.

Test Instances with Many Local Optima

In this section, we consider three multimodal 2-objective test instances - F10, F11 and F12. F10 and F12 have simple PS shapes while the PS shape of F11 is complicated.

Table 6.6: Average set coverage among DE-MOEA/D(D), RM-MEDA-II(R), and DE-NSGA-II (N) on F10, F11 and F12

Instance	C(D, R)	C(R,D)	C(D, N)	C(N,D)	C(R, N)	C(N,R)
F10	0.05	19.88	4.06	31.06	5.91	0.14
F11	0.50	99.11	8.31	75.38	99.80	27.19
F12	0.75	61.62	11.21	32.59	37.26	13.07

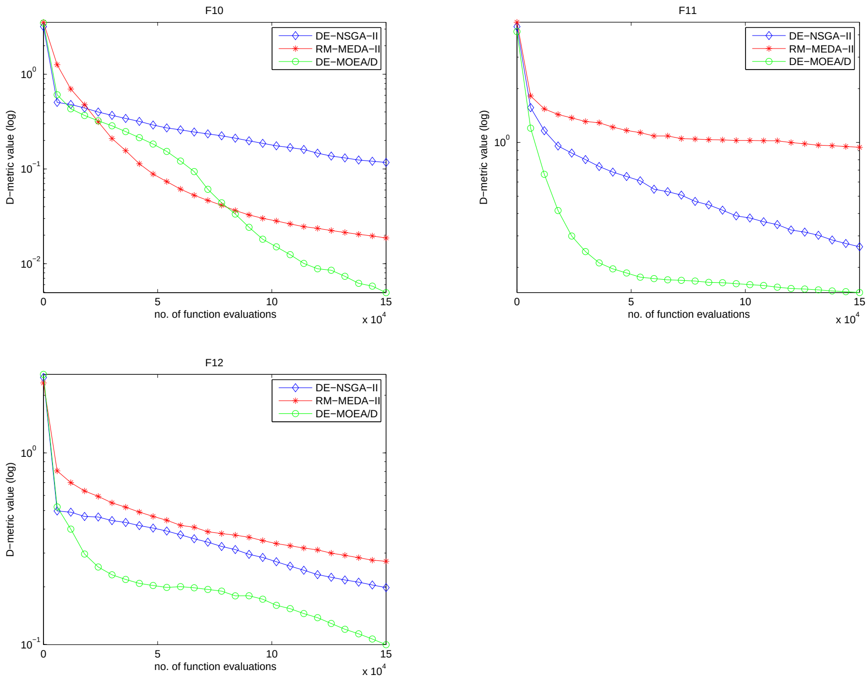
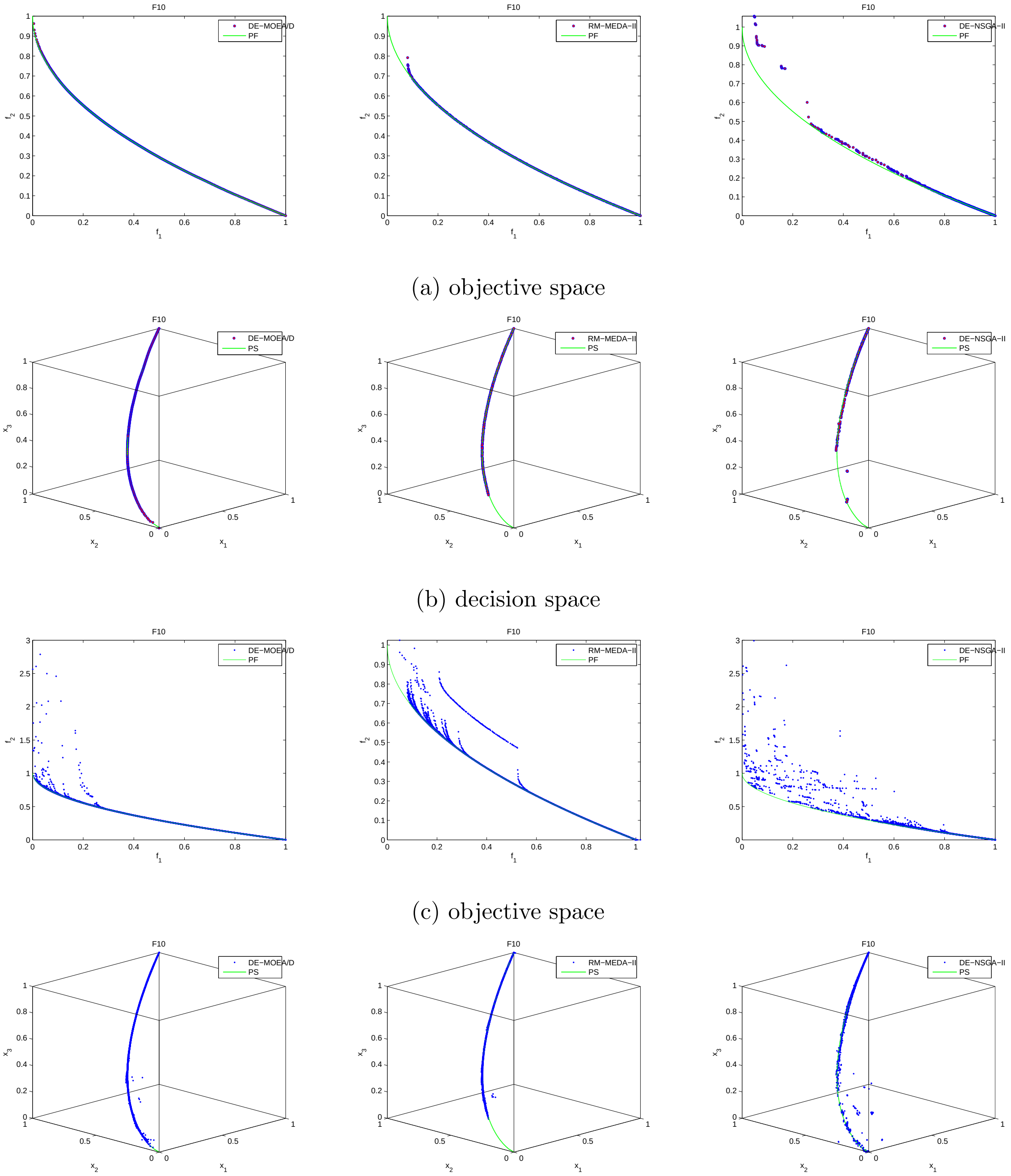


Figure 6.14: The evolution of D-metric values found by three algorithms on F10-F12



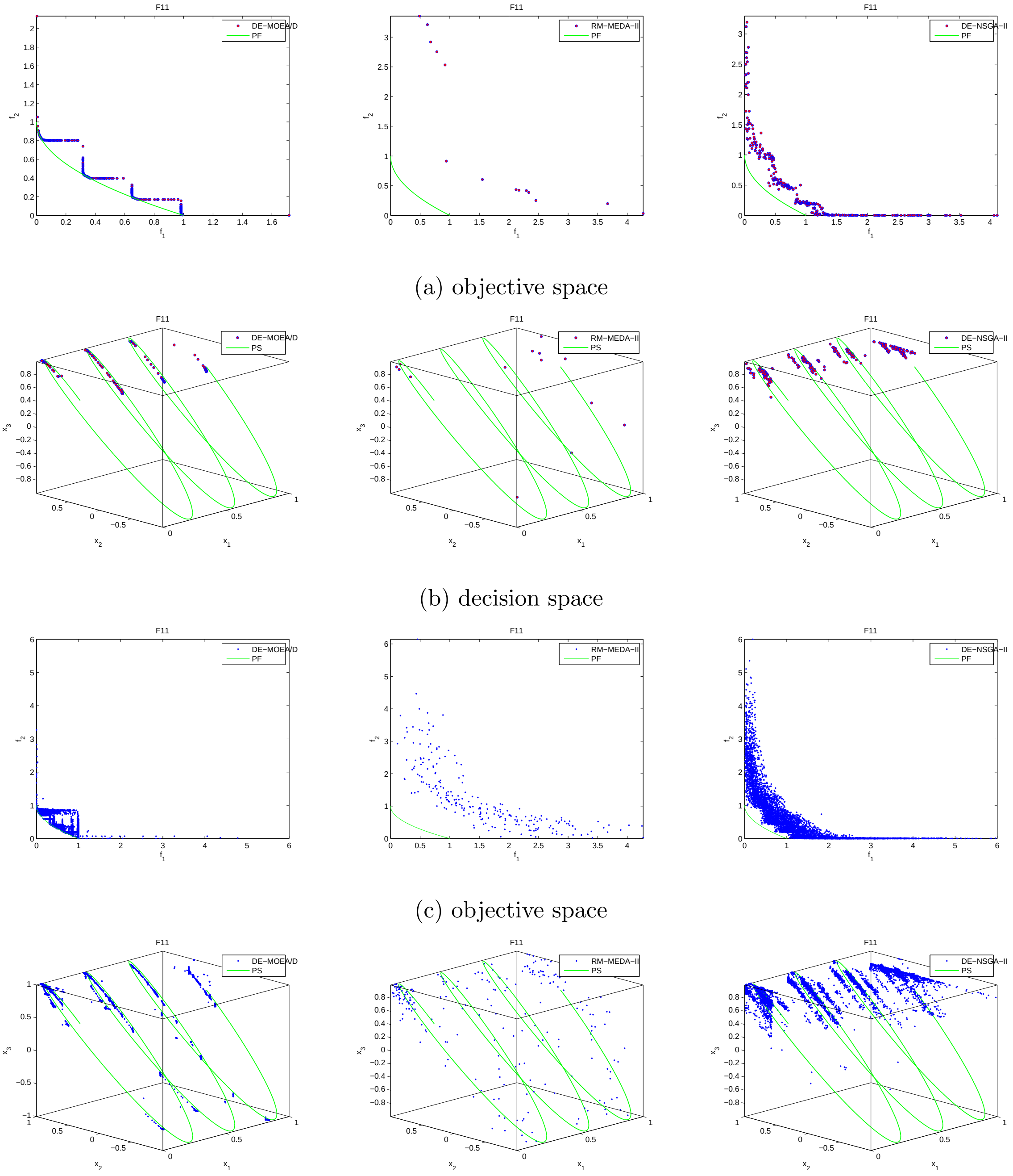
(a) objective space

(b) decision space

(c) objective space

(d) decision space

Figure 6.15: Plots of the solutions obtained by the run with the lowest D-metric value ((a) and (b)) and 20 runs ((c) and (d)) of each algorithm on F10.



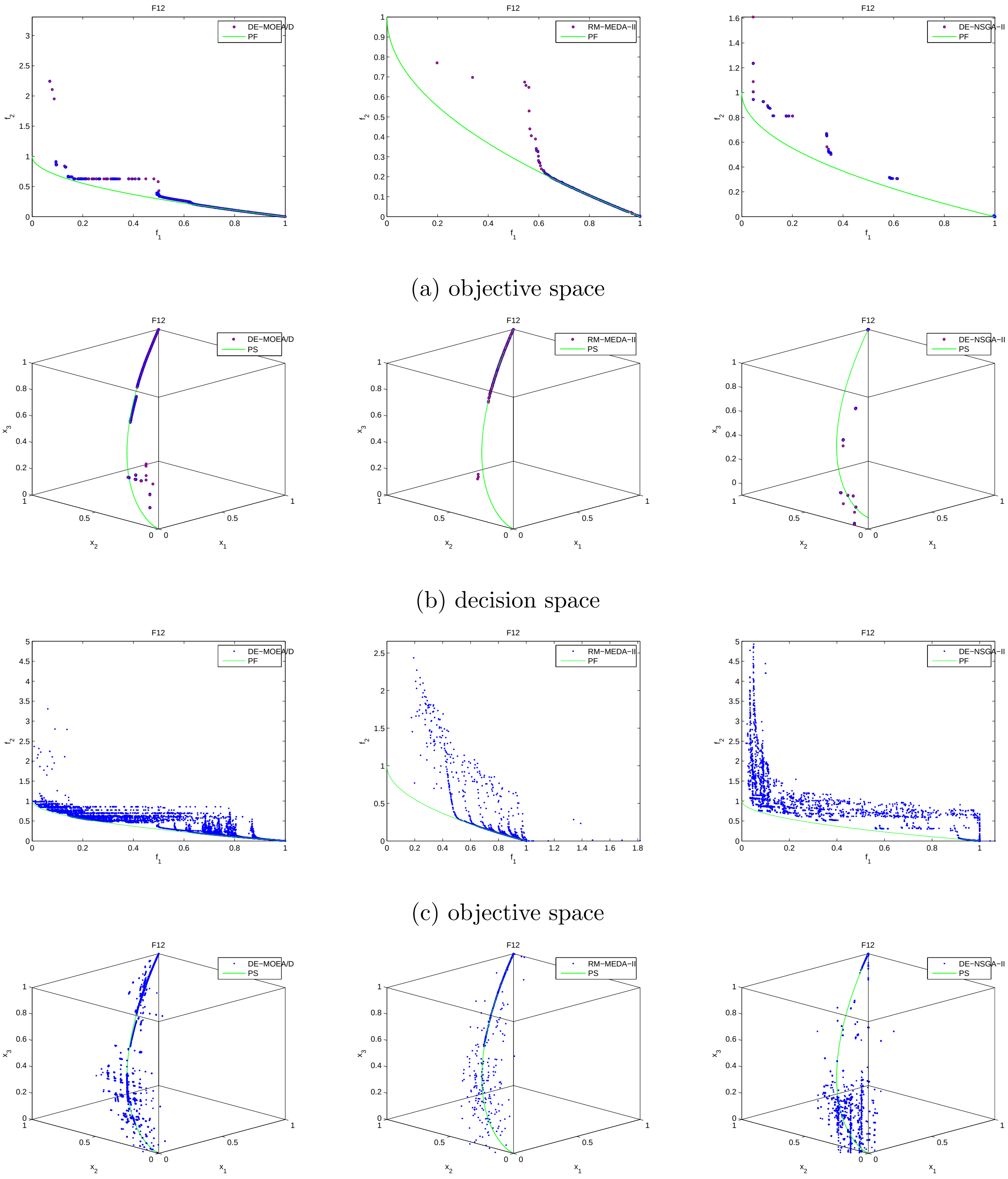
(a) objective space

(b) decision space

(c) objective space

(d) decision space

Figure 6.16: Plots of the solutions obtained by the run with the lowest D-metric value ((a) and (b)) and 20 runs ((c) and (d)) of each algorithm on F11.



(a) objective space

(b) decision space

(c) objective space

(d) decision space

Figure 6.17: Plots of the solutions obtained by the run with the lowest D-metric value ((a) and (b)) and 20 runs ((c) and (d)) of each algorithm on F12.

Table 6.7: The D-metric values of the nondominated solutions found by DE-MOEA/D, RM-MEDA-II and DE-NSGA-II on F13 and F14

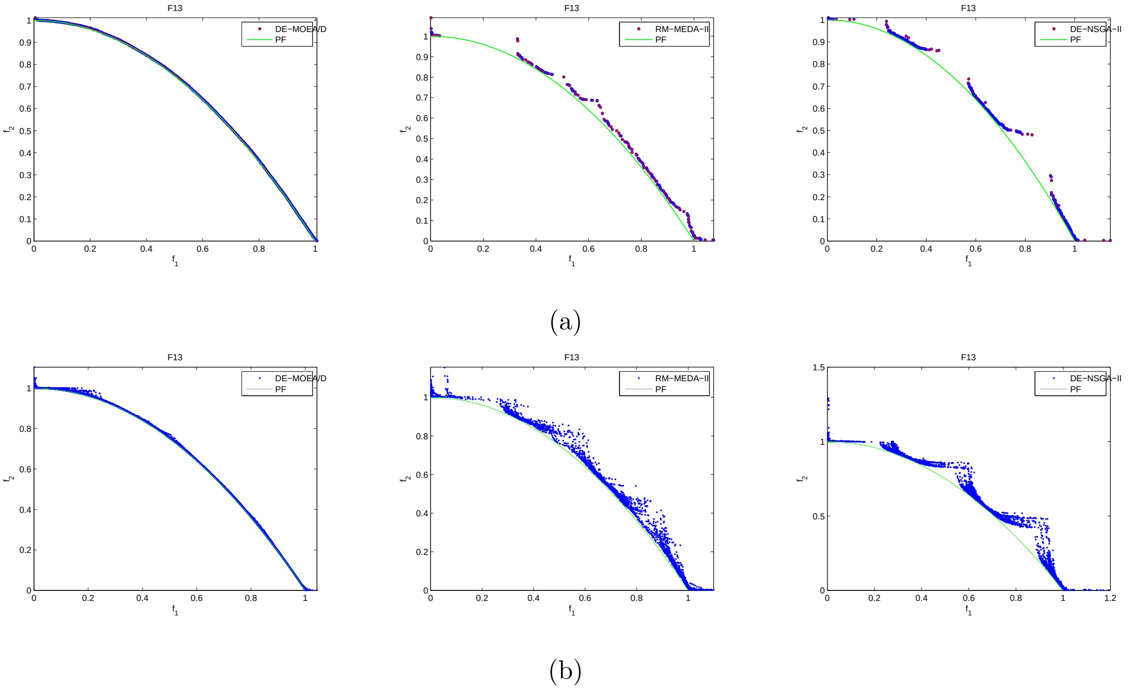
GD-value	DE-MOEA/D			RM-MEDA-II			DE-NSGA-II		
Instance	mean	min	std	mean	min	std	mean	min	std
F13	0.0035	0.0025	0.0008	0.0304	0.0238	0.0048	0.0395	0.0303	0.0061
F14	0.0128	0.0051	0.0089	0.0411	0.0275	0.0136	0.0377	0.0161	0.0233

Table 6.8: Average set coverage among DE-MOEA/D(D), RM-MEDA-II(R) and DE-NSGA-II (N) on F13 and F14

Instance	C(D, R)	C(R,D)	C(D, N)	C(N,D)	C(R, N)	C(N,R)
F13	1.42	96.25	0.97	93.01	49.92	4.25
F14	88.94	94.09	10.12	75.41	40.31	94.31

Table 6.5 and Table 6.6 show the D-metric and C-metric values of the nondominated fronts found by each algorithm on each test instance respectively. These experimental results reveal that DE-MOEA/D performs the best among three algorithms on all three multiomodal test instances in terms of both metrics. It can also be seen from this table that only DE-MOEA/D produces the good approximation of the PF for F10 with the average D-metric value less than 0.005.

Figure 6.14 plots the evolution of the D-metric values of the nondominated solutions in the current population and Figures 6.15-6.17 show that nondominated fronts found by each algorithm on F10-F12. It is very clear in Figure 6.14 that DE-MOEA/D, in terms of D-metric, converges faster than the other two algorithms on F10, F11 and F12. As also shown in Figure 6.15, only DE-MOEA/D finds the whole PF and PS of F10 in the run with the best D-metric value. For F11 and F12, all three algorithms fail to find a good approximation to the PFs in all 20 runs.



(a)

(b)

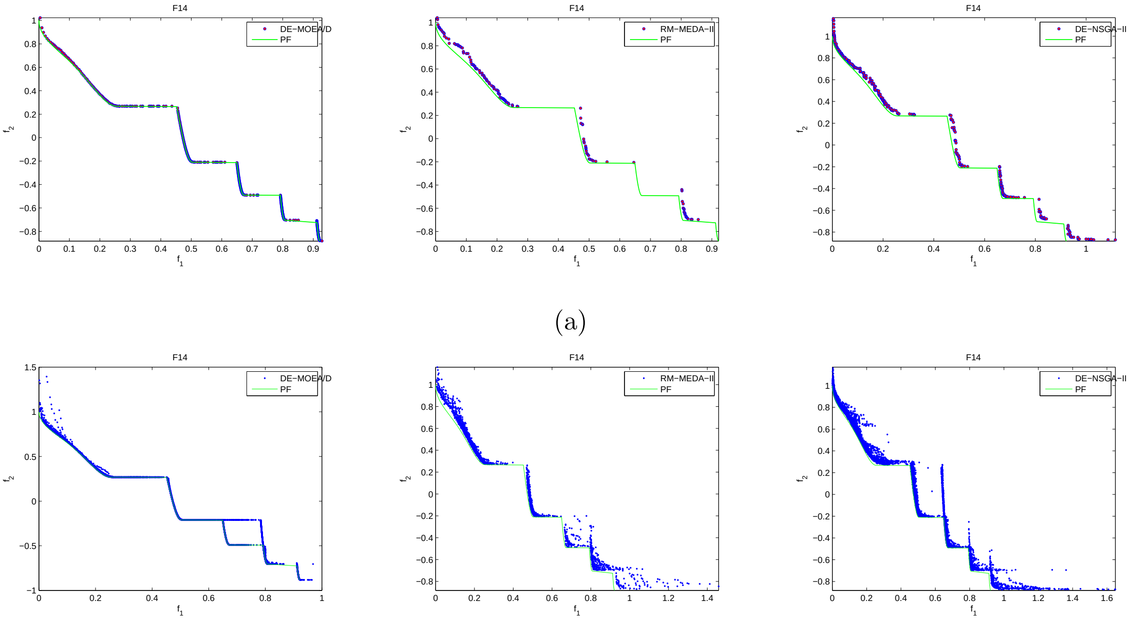
Figure 6.18: Plots of the solutions obtained by (a) the run with the lowest D-metric values and (b) 20 runs of each algorithm on F13.

Test Instances with Nonconvex PFs

F13 and F14 are two 2-objective test instances with concave and discontinuous PFs, respectively. F13 has the same PS as that of F2 while the PS of F14 is a part of the PS of F2.

Table 6.7 presents the D-metric values found by three algorithms on F13 and F14 and Table 6.8 shows the C-metric values found by each algorithm. Figures 6.18 and 6.19 plot the nondominated solutions of each algorithm in the objective space.

The above experimental results show that DE-MOEA/D still performs better than the other two algorithms. It is evident in Figures 6.18 and 6.19 that DE-MOEA/D fails to find the whole PF in some runs. This suggests that F13 and F14 are harder than their counterpart F2 due to the nonconvex PFs.



(a)

(b)

Figure 6.19: Plots of the solutions obtained by (a) the run with the lowest D-metric values and (b) 20 runs of each algorithm on F14.

6.4.4 More Discussions

In the following, we further investigate the nonuniforamlity of PS in our test instances and the mating restriction in DE-NSGA-II.

Nonuniformality of the PS

Revisiting Figures 6.8, 6.9 and 6.11, we can observe that the distribution of the nondominated solutions found by DE-MOEA/D is uniform along the PF in the objective space but not uniform along the PS in the decision space. This indicates that some neighboring subproblems may not have close optimal solutions. To further study the similarity between the optimal solutions of subproblems in the PS, we plot the optimal solutions of 50 subproblems with uniform weight vectors in the PS for F1-F6 in Figure 6.20.

For F2 and F5, the distribution of the optimal solutions of 50 subproblems is

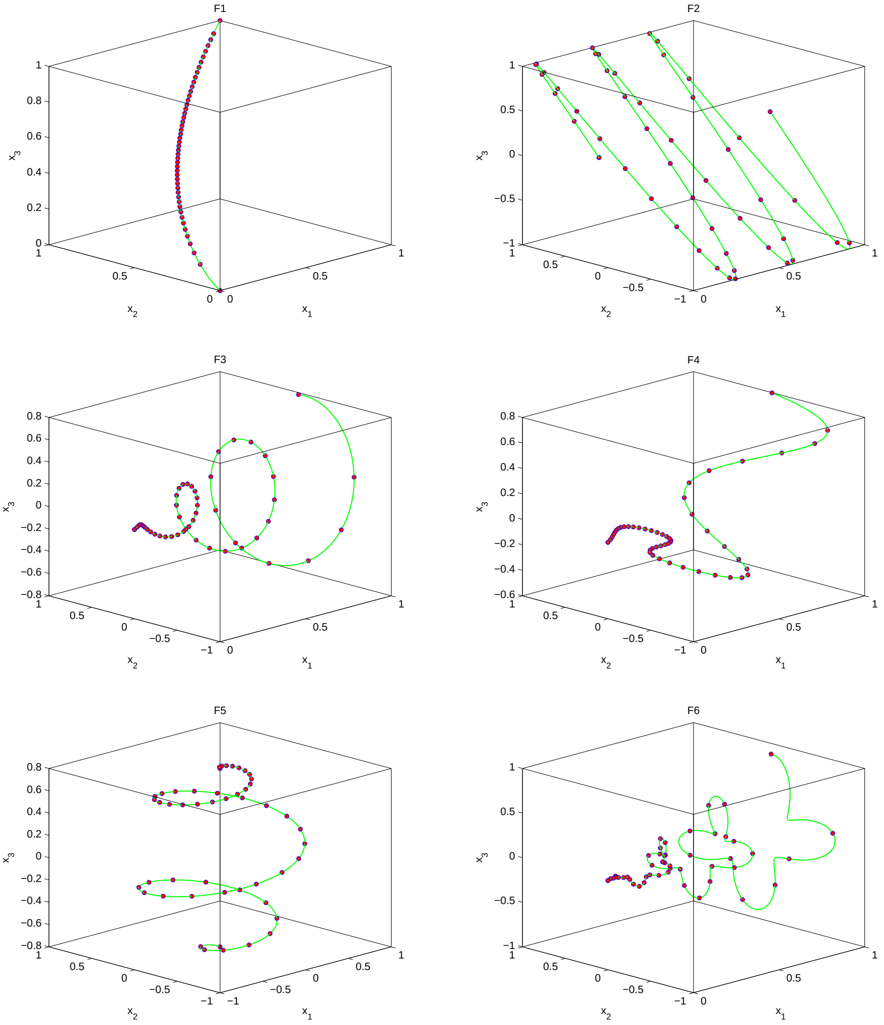


Figure 6.20: Plots of the optimal solutions of 50 subproblems with uniform weight vectors in the PS.

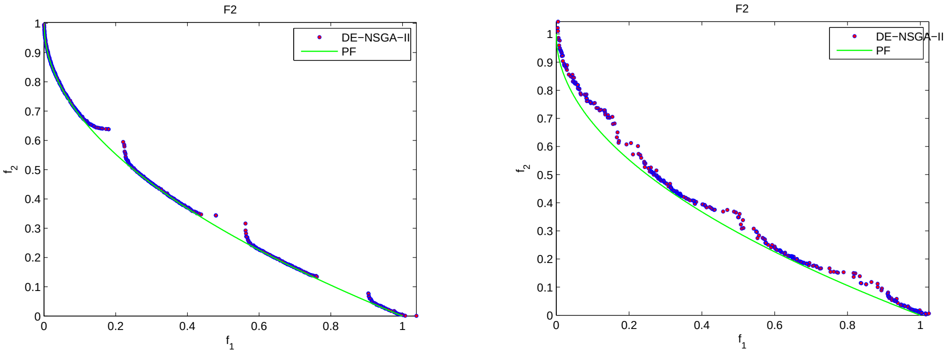
uniform in the decision space in a sense. In contrast, for F3, F4 and F6, the solutions of two neighboring subproblems are not close in the decision space. In this case, the recombination between the solutions of two neighboring subproblems has low probability to generate the promising solutions in the PS. This is the main reason why DE-MOEA/D failed to find the good approximation of the PF for F6.

DE-NSGA-II with Mating Restriction

One of the main differences between DE-MOEA/D and DE-NSGA-II is that the former uses mating restriction for recombination. However, using mating restriction in DE-NSGA-II is also straightforward. In the following, we also test the version of DE-NSGA-II with mating restriction on the test instance suggested in this chapter. In our experiments, the selection of mating parents in DE-NSGA-II is performed as follows.

- Step 1: randomly select the first parent x^{r_1} from the current population;
- Step 2: calculate the closeness of all other individuals in the current population to x^{r_1} based on the Euclidean distance in the objective space;
- Step 3: randomly select two parents x^{r_2} and x^{r_3} from T nearest neighboring individuals.

We have tested DE-NSGA-II with mating restriction on F2, where the neighborhood size T is the same as that in DE-MOEA/D and all other parameters remain the same as in Section 6.4.2. Figure 6.21 plots the nondominated front with the lowest D-metric value found by DE-NSGA-II with mating restriction. It is very clear that the performance of DE-NSGA-II is improved significantly by incorporating mating restriction. However, DE-NSGA-II with mating restriction still performs worse than DE-MOEA/D on F2 and has difficulty in converging towards the whole PF. This can be explained by the fact that



(a) (b)

Figure 6.21: The nondominated fronts found by DE-NSGA-II (a) with and (b) without mating restriction

DE-MOEA/D uses decomposition techniques in its fitness assignment such that MOEA/D spends the same computational effort to optimize each subproblem. The optimal solution of all subproblems should cover the whole PF very well. In contrast, the Pareto-based fitness assignment of DE-NSGA-II lacks the ability to force the population to converge towards the whole PF.

6.5 Sensitivity, Scalability and Reproduction Operators in DE-MOEA/D

In this section, the sensitivity of parameters-T, ζ and n_r in DE-MOEA/D is first analyzed. Then the scalability of the number of variables is investigated. Afterwards, the effect of reproduction operators in MOEA/D is studied.

6.5.1 Sensitivity of Parameters

Sensitivity of T

To study how the performance of DE-MOEA/D is sensitive to the size of neighborhood T, we have tested 9 values of neighborhood size (i.e., $T = 5, 10, 20, 50, 100, 150, 200, 250, 30)$ in DE-MOEA/D for F2. All other parameters remain the same as in Section 6.4.2 in our experiments.

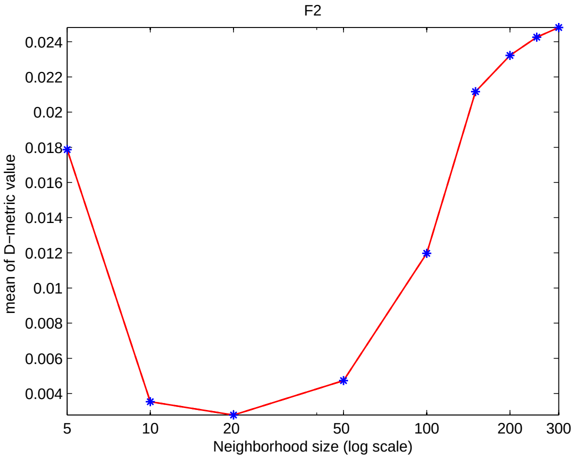


Figure 6.22: The average D-metric values vs. the size of neighborhood (T) in F2

Figure 6.22 shows the average D-metric values versus the neighborhood size T. It is clear that the performance of DE-MOEA/D is sensitive to the setting of neighborhood size T. The best (i.e., smallest) average value of the D-metric is obtained by DE-MOEA/D with $T = 20$. As we can also observe, DE-MOEA/D with $T = 10$ and $T = 50$ can find the average D-metric value below 0.006. This indicates that DE-MOEA/D can still find the good approximation of the PF of F2 when the neighborhood size varies in an appropriate range. The results from Figure 6.22 also reveal that DE-MOEA/D with either $T = 5$ or $T \geq 100$ fails to find the good approximation of the PF since the average D-metric values found are larger than 0.01. This can be explained by the exploration and exploitation of the search space in DE-MOEA/D. On the one hand, DE-MOEA/D with small T lacks the ability of exploring the search space. On the other hand, DE-MOEA/D with large T cannot exploit the PS well since the mating parents could be distant (i.e., very dissimilar) in the decision space.

Sensitivity of n_r

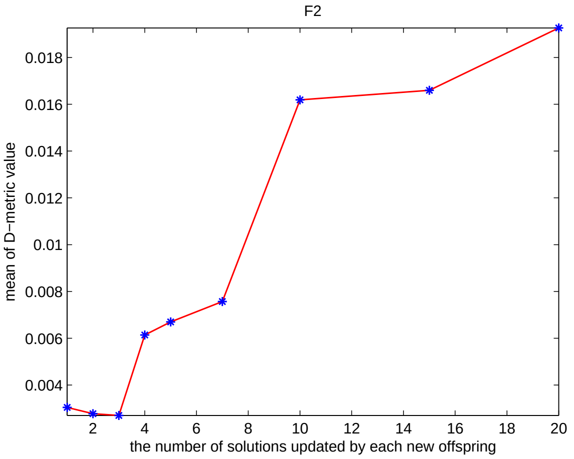


Figure 6.23: The average D-metric values vs. the maximal number of solutions updated by each child solution (n_r) in F2

For studying the sensitivity of n_r to the performance of DE-MOEA/D, we have tried 9 values of n_r (i.e., 1, 2, 3, 4, 5, 7, 10, 15, 20) for F2. All other parameters remain same in Section 6.4.2 in our experiments.

Figure 6.23 shows the average D-metric values versus the different values of n_r. As clearly shown in this figure, DE-MOEA/D with $n_r = 1, 2, 3$ can find the good approximation of the PF with the average D-metric values below 0.005. In contrast, DE-MOEA/D with large n_r (≥ 10) performs clearly worse in minimizing D-metric values. As mentioned before, the population in DE-MOEA/D may lose diversity when one child solution is of very good quality and replaces most of its neighboring solutions. As a result, the mating parents chosen from the current solutions of neighboring subproblems could be same. This will lead to the convergence towards local optima or the local part of the PF.

Sensitivity of ζ

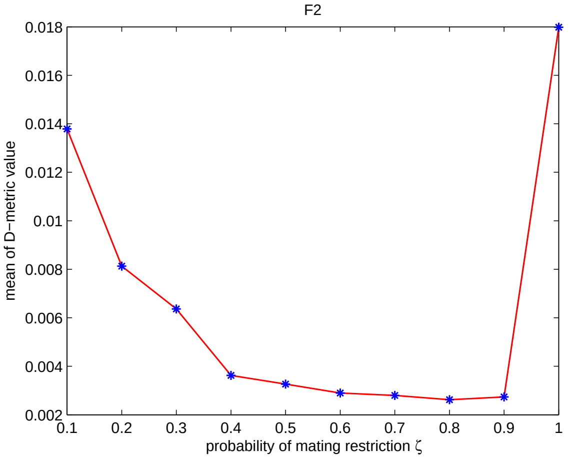


Figure 6.24: The average D-metric values vs. probability of selecting mating parents from neighborhood ζ in F2

To study the sensitivity of ζ to the performance of DE-MOEA/D, we have tried 10 values of ζ (i.e., 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0). All other parameters remain the same as in Section 6.4.2 in our experiments.

Figure 6.24 presents the average D-metric values versus the values of ζ. It is evident from the results in this figure that DE-MOEA/D with $\zeta = 0.9$ has the best performance in terms of the D-metric. This indicates that selecting 10% of solutions from the whole population for recombination does improve the performance of DE-MOEA/D. In this case, the distant solutions have chance to mate in a low probability. This strategy increases the ability of DE-MOEA/D to explore the search space. However, more computational efforts are still spent on the recombination between neighboring solutions in order to exploit the

PS efficiently. When ζ is set to 0.1, DE-MOEA/D performs similarly to DE-NSGA-II. This is because most recombination operations are performed without mating restriction in both algorithms.

6.5.2 Scalability of the Number of Variables

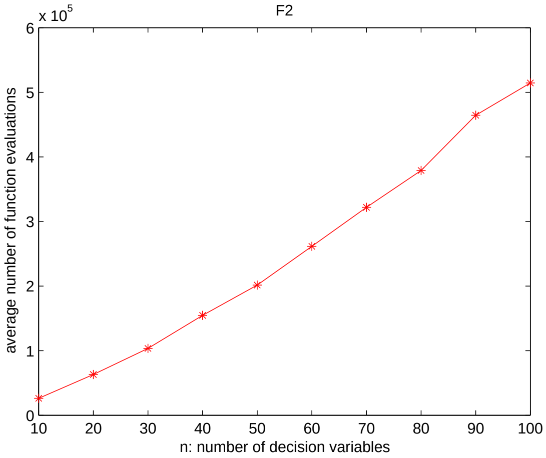


Figure 6.25: The average numbers of function evaluations used for reducing the D-metric value under 0.005 for F2 with different number of decision variables

We have tested DE-MOEA/D on F2 with different number of decision variables. All parameter settings remain the same as in Section 6.4.2. Figure 6.25 presents the average number of function evaluations needed in a successful run of MOEA/D, in which the D-metric value of the final solutions is below 0.005. It is very clear that the number of function evaluations linearly scales up with number of decision variables. This observation is same as that in Chapter 4.

6.5.3 Effect of Reproduction Operators

To investigate the effect of the reproduction operators in DE-MOEA/D, we have compared four versions of DE-MOEA/D with the following four operators on F2.

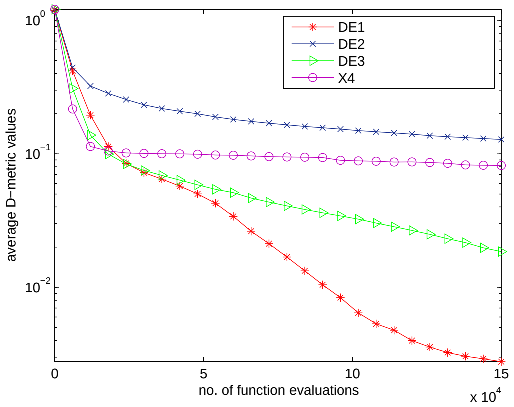


Figure 6.26: Comparison of four reproduction operators in DE-MOEA/D

- DE1 - DE operator with polynomial mutation (CR=1);
- DE2 - DE operator without polynomial mutation (CR=1);
- DE3 - DE operator without polynomial mutation (CR=0.9);
- X4 - SBX with polynomial mutation.

All parameter settings remain the same as in Section 6.4.2. Figure 6.26 shows the evolution of the average D-metric values of the solutions found by DE-MOEA/D with four reproduction operators. From the results in this figure, we can make the following remarks:

- DE2 performs much worse than other three reproduction operators on F2. This indicates that the polynomial mutation has a large influence on the performance of DE-MOEA/D since it can provide more diversity for child solutions.
- DE1 performs better than DE3 in minimizing D-metric values on F2. This reveals that DE operator with polynomial mutation is more powerful than the standard DE operators in DE-MOEA/D.

- X4 is outperformed by DE1 and DE3. This result indicates that SBX is not suitable for the continuous MOPs with nontrivial PS. It doesn't have the ability to model the PS shapes.

6.6 Summary

In this chapter, a set of continuous multiobjective test instances with prescribed PS shapes were suggested. A new version of MOEA/D with DE operators, called DE-MOEA/D, was suggested. Unlike two versions of MOEA/D used in Chapters 4 and 5, two new strategies for encouraging diversity were taken into consideration in DE-MOEA/D. On the one hand, mating parents selected from the whole population without restriction on their similarity were allowed for recombination with a low probability. On the other hand, the maximal number of solutions updated by each child solution was limited. DE-MOEA/D was compared with RM-MEDA-II and DE-NSGA-II, which have the ability to deal with the continuous MOPs with variable linkages. The following are the major conclusions drawn from our experimental results.

- The comparison results showed that all three algorithms can find the good approximation of the PFs for the test instances with simple PS shapes - F2 and F8. In terms of D-metric and C-metric, DE-MOEA/D performed the best among three algorithms for these test instances. DE-MOEA/D also found the better distribution of the nondominated solutions for the 3-objective test instance F8.

- DE-MOEA/D still outperformed other two algorithms for the test instances with complicated PS shapes - F2-F7 and F9 in terms of D-metric and C-metric. The experimental results also indicated that none of the algorithms can solve F6 well since its PS shapes is overcomplicated.

- The comparison results on the test instances with many local nondominated fronts showed that none of algorithms can solve them well except F10, which is the test instance with simple PS shape.

- The experimental results on F13 and F14 suggested that the test instances with nonconvex PFs are slightly more difficult than their counterparts with convex PFs.

- The performance of DE-NSGA-II was also improved by using mating restriction. However, DE-NSGA-II still performed worse than DE-MOEA/D.

- The experimental results on the sensitivities of parameters also showed that the new strategies suggested in this chapter for encouraging good diversity do improve the performance of DE-MOEA/D.

Chapter 7

Conclusions

Decomposition and Pareto dominance are two basic strategies for fitness assignment in MOEAs. In decomposition-based MOEAs, it is easy to find a diverse set of nondominated solutions if a MOP is decomposed into multiple subproblems appropriately. Besides, scalar optimization techniques can be readily integrated in these algorithms. In contrast, MOEAs using Pareto dominance treat a MOP as a whole and mainly rely on domination for their fitness assignment. These algorithms may not be very good at generating an even distribution of solutions along the PF.

This thesis proposed a simple and generic evolutionary algorithm based on decomposition called MOEA/D. It first uses a decomposition method to decompose the MOP into multiple scalar subproblems. Then, an EA is employed to solve these subproblems simultaneously. Each individual solution in the population of MOEA/D is associated with a subproblem. A neighboring relationship among subproblems is defined based on the distances of their weight vectors. In MOEA/D, the optimization of subproblem uses the current information its neighboring subproblems since two neighboring subproblems should have close optimal solutions.

This thesis started by the introduction of the basic definitions and concepts in multiobjective optimization and the overview of MOEAs. Chapter 2 was devoted to the decomposition techniques for multiobjective optimization in traditional mathematical programming. Chapter 3 presented the general framework of MOEA/D and discusses the major issues in MOEA/D. The comparison of MOEA/D with NSGA-II on a set of benchmark continuous MOPs and the investigation of decomposition methods in MOEA/D were provided in Chapter 4. The combination of MOEA/D with local heuristics was then suggested in Chapter 5. Chapter 6 presented a set of continuous multiobjective test problems with nontrivial PS shapes and developed the version of MOEA/D with DE algorithms.

7.1 Major Conclusions

From the applications of MOEA/D to the different MOPs considered in this thesis, the major conclusions are drawn in the following.

- Efficiency and effectiveness of MOEA/D

  We have compared MOEA/D with NSGA-II and MOGLS on benchmark continuous multiobjective test problems and multiobjective knapsack problems, respectively. Our analysis has shown that MOEA/D has lower computational complexity than NSGA-II and MOGLS. The experimental results have also confirmed it. In terms of solution quality, MOEA/D with simple decomposition method outperformed or performed similarly to NSGA-II and MOGLS.

  We have also compared MOEA/D with the SQP-based NBI on the benchmark 2-objective test instances. Our results have shown that MOEA/D clearly outperforms NBI for all test instances. Therefore, we concluded that MOEA/D is less sensitive to the connectedness of Pareto-optimal solutions and the multimodality of objective functions than then SQP-based NBI.

We have investigated the small population and reproduction operators in MOEA/D. Our experimental results on the benchmark continuous MOPs have shown that MOEA/D with small population performs better than the Pareto-based MOEA - NSGA-II. This can be explained by the fact that the optimal solutions of all subproblems in MOEA/D are always (weakly) Pareto-optimal to the original MOP. Besides, our experimental results on the continuous MOPs with nontrivial PS shapes have also shown that reproduction operators have a large influence on the performance of MOEA/D, where DE operators performed clearly better than the SBX crossover.

- Mating Restriction in MOEA/D

  We have investigated the mating restriction in MOEA/D for benchmark continuous MOPs, MOKP and continuous MOPs with nontrivial PS geometries. Our experimental demonstrated that mating restriction is not effective in MOEA/D for the benchmark continuous MOPs since all Pareto-optimal solutions have the same values in all components except the first one. For MOKP and continuous MOPs with nontrivial PS geometries, we demonstrated that mating restriction does improve the performance of MOEAs.

- Suitability of Decomposition Methods in MOEA/D

  We have shown that MOEA/D with PBI approach is able to generate a very uniform distribution of representative Pareto-optimal solutions on the PF on two 3-objective test problems. We have also demonstrated that MOEA/D with a naive objective normalization technique can deal with disparately scaled objectives very well. These results suggest that more efficient and effective implementation of MOEA/D could be obtained if appropriate decomposition methods are used.

We have also investigated the suitability of the weighted sum approach in MOEA/D for both continuous multiobjective optimization problems and multiobjective knapsack problems. For the benchmark continuous multiobjective test problems, the weighted sum approach is clearly worse than the weighed Tchebycheff approach. But the weighted sum approach outperforms the weighted Tchebycheff approach for multiobjective knapsack problems. These results suggested that the performance of decomposition methods is problem-dependent.

- Difficulties of the continuous multiobjective test problems with nontrivial PS shapes

  We have compared three algorithms - DE-MOEA/D, RM-MEDA-II and DE-NSGA-II on the continuous MOPs with nontrivial PS shapes. The experimental results have shown that both RM-MEDA-II and DE-NSGA-II were unable to find the good approximation of the PSs for the test instances with complicated PS. Particularly, when the test instances have overcomplicated PS shapes or many local nondominated fronts, none of the algorithms can find the good approximation of the PF and PS satisfactorily.

  All the test instances with nontrivial PS shapes suggested in our work are easy to describe and have known PFs and PSs. As mentioned above, some of them are very hard for existing MOEAs. Therefore, they should be able to provide a good platform for testing the performance of various MOEAs.

7.2 Future Work

MOEA/D provides a bridge between MOEAs and scalar optimization methods. The advanced approaches from both fields can be used to improve the effectiveness and efficiency of MOEA/D. In the following, we describe some future work based on our research in this thesis.

- Hybridization of MOEA/D with advanced scalar optimization methods

  In Chapter 5 and Chapter 6, we have demonstrated the success of the combination of MOEA/D with local heuristics and differential evolution, where local heuristics can speed up the convergence of MOEA/D and DE-based reproduction operators are effective in sampling promising offspring solutions for the continuous MOPs with nontrivial PS geometries. In fact, more advanced local search methods can be used in MOEA/D for solving different MOPs, such as gradient-based local search [99][80], tabu search [28], simulated annealing [47], guided local search [92], variable neighbourhood search [10], etc. Besides, many efficient sampling techniques for scalar EAs can also be used in MOEA/D. Examples are orthogonal genetic algorithms [96], regularity modelling [100], particle swarm optimization [7][5], guided mutation [98], and so on. The hybridization of MOEA/D with these optimization approaches should be very promising to improve the effectiveness and efficiency of MOEA/D for some hard optimization problems.

- Adaptive MOEA/D

  One of the main weaknesses of MOEA/D is that the weighted vectors of subproblems are fixed during the evolutionary process. This might not be very suitable for the MOPs with irregular PFs, such as disconnectedness. The main reason is that the optimal solutions of some subproblems might not be Pareto-optimal to the original MOP. Therefore, the computational effort spent on such subproblems is a waste. To overcome this weakness, we could tune the weight vectors of subproblems dynamically during the search. In this way, the optimal solutions of all subproblems are also Pareto-optimal.

- MOEA/D for single objective constrained optimization problems

  Recently, some researchers have paid some attention in applying MOEAs to solve single objective constrained optimization problems [23][4]. The basic idea is to convert a single objective constrained problem into a MOP by treating the violation of constraints as an extra objective. Since the optimal solution of the original problem is the Pareto-optimal solution with the minimal value of the extra objective in the PF, it is necessary to spend more computational effort to search the area near the optima of the original problem. Very interestingly, this is extremely easy in MOEA/D since it can force the population to converge towards the desired part of PF.

- MOEA/D for multiobjective combinatorial optimization problems.

  In this thesis, MOKP is the only multiobjective combinatorial optimization solved by MOEA/D. In fact, many real-life optimization problems, such as airline scheduling and timetabling [81], can be modelled as multiobjective combinatorial optimization problems. These problems could be very challenging to be solved due to their huge search space and complicated constraints. MOEA/D should be applicable to deal with these problems since many advanced scalar optimization methods can be readily used.

Appendix A

Benchmark Continuous Multiobjective Test Problems

- ZDT1

$$\begin{aligned} f_1(x) &= x_1, \\ f_2(x) &= g(x)\left[1-\sqrt{f_1(x)/g(x)}\right] \end{aligned}$$

where

$$g(x) = 1 + 9(\sum_{i=2}^{n} x_i)/(n-1)$$

and $x = (x_1, \ldots, x_n)^T \in [0,1]^n$. Its PF is convex. $n = 30$ in our experiments.

- ZDT2

$$\begin{aligned} f_1(x) &= x_1, \\ f_2(x) &= g(x)\left[1-(f_1(x)/g(x))^2\right] \end{aligned}$$

where $g(x)$ and the range and dimensionality of x are the same as in ZDT1. The PF of ZDT2 is nonconvex.

- ZDT3

$$
\begin{aligned}
f_1(x) &= x_1, \\
f_2(x) &= g(x)\left[1 - \sqrt{f_1(x)/g(x)} - \frac{f_1(x)}{g(x)}\sin(10\pi x_1)\right]
\end{aligned}
$$

where $g(x)$ and the range and dimensionality of x are the same as in ZDT1. Its PF is disconnected. The two objectives are disparately-scaled in the PF, f_1 is from 0 to 0.852 while f_2 from -0.773 to 1.

- ZDT4

$$
\begin{aligned}
f_1(x) &= x_1, \\
f_2(x) &= g(x)\left[1 - \sqrt{f_1(x)/g(x)}\right]
\end{aligned}
$$

where

$$g(x) = 1 + 10(n-1) + \sum_{i=2}^{n}\left[x_i^2 - 10\cos(4\pi x_i)\right],$$

and $x = (x_1, \dots, x_n)^T \in [0,1] \times [-5,5]^{n-1}$. It has many local PFs. $n = 10$ in our experiments.

- ZDT6

$$
\begin{aligned}
f_1(x) &= 1 - \exp(-4x_1)\sin^6(6\pi x_1), \\
f_2(x) &= g(x)\left[1 - (f_1(x)/g(x))^2\right]
\end{aligned}
$$

where

$$g(x) = 1 + 9\left[(\sum_{i=2}^{n} x_i)/(n-1)\right]^{0.25},$$

and $x = (x_1, \dots, x_n)^T \in [0,1]^n$. Its PF is nonconvex. The distribution of the Pareto solutions in the Pareto front is very nonuniform, i.e., For a set of uniformly distributed points in the Pareto set in the decision space, their images crowd in a corner of the Pareto front in the objective space. $n = 10$ in our experiments.

- DTLZ1

$$
\begin{aligned}
f_1(x) &= (1+g(x))x_1x_2 \\
f_2(x) &= (1+g(x))x_1(1-x_2) \\
f_3(x) &= (1+g(x))(1-x_1),
\end{aligned}
$$

where

$$g(x) = 100(n-2) + 100\sum_{i=3}^{n}\{(x_i-0.5)^2 - \cos[20\pi(x_i-0.5)]\}$$

and $x = (x_1, \dots, x_n)^T \in [0,1]^n$. Its PF is nonconvex. The function value of a Pareto optimal solution satisfies $\sum_{i=1}^{3} f_i = 1$ with $f_i \geq 0, i = 1, 2, 3$. $n = 10$ in our experiments.

- DTLZ2

$$
\begin{aligned}
f_1(x) &= (1+g(x))\cos(x_1\pi/2)\cos(x_2\pi/2), \\
f_2(x) &= (1+g(x))\cos(x_1\pi/2)\sin(x_2\pi/2), \\
f_3(x) &= (1+g(x))\sin(x_1\pi/2), \qquad \text{(A.1)}
\end{aligned}
$$

where

$$g(x) = \sum_{i=3}^{n} x_i^2$$

and $x = (x_1, \dots, x_n)^T \in [0,1]^2 \times [-1,1]^{n-2}$. Its PF is nonconvex. The function value of a Pareto optimal solution satisfies $\sum_{i=1}^{3} {f_i}^2 = 1$ with $f_i \geq 0, i = 1, 2, 3$. $n = 10$ in our experiments.

Appendix B

Continuous Multiobjective Test Problems with Prescribed Pareto Sets

- F1

$$
\begin{aligned}
f_1(x) &= x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - x_1^{0.5(1.0+\frac{3(j-2)}{30-2})}\right)^2 \\
f_2(x) &= 1-\sqrt{x_1} + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - x_1^{0.5(1.0+\frac{3(j-2)}{30-2})}\right)^2
\end{aligned}
$$

where $x = (x_1, \dots, x_{30})^T \in [0,1]^{30}$. $J_1 = \{2, 4, \dots, 30\}$ and $J_2 = \{3, 5, \dots, 29\}$.

- F2

$$
\begin{aligned}
f_1(x) &= x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - \sin(6\pi x_1 + \frac{j\pi}{30})\right)^2 \\
f_2(x) &= 1-\sqrt{x_1} + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - \sin(6\pi x_1 + \frac{j\pi}{30})\right)^2
\end{aligned}
$$

where $x = (x_1, \dots, x_{30})^T \in [0,1]\times[-1,1]^{29}$. $J_1 = \{2, 4, \dots, 30\}$ and $J_2 = \{3, 5, \dots, 29\}$.

- F3

$$
\begin{aligned}
f_1(x) &= x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - 0.8x_1\cos(6\pi x_1 + \frac{j\pi}{30})\right)^2 \\
f_2(x) &= 1 - \sqrt{x_1} + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - 0.8x_1\sin(6\pi x_1 + \frac{j\pi}{30})\right)^2
\end{aligned}
$$

where $x = (x_1, \ldots, x_{30})^T \in [0,1]\times[-1,1]^{29}$. $J_1 = \{2, 4, \ldots, 30\}$ and $J_2 = \{3, 5, \ldots, 29\}$.

- F4

$$
\begin{aligned}
f_1(x) &= x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - 0.8x_1\cos(\frac{6\pi x_1 + \frac{j\pi}{30}}{3})\right)^2 \\
f_2(x) &= 1 - \sqrt{x_1} + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - 0.8x_1\sin(6\pi x_1 + \frac{j\pi}{30})\right)^2
\end{aligned}
$$

where $x = (x_1, \ldots, x_{30})^T \in [0,1]\times[-1,1]^{29}$. $J_1 = \{2, 4, \ldots, 30\}$ and $J_2 = \{3, 5, \ldots, 29\}$.

- F5

$$
\begin{aligned}
f_1(x) &= x_1 + \frac{2}{|J_{1a}| + |J_{1b}|}\left(\sum_{j\in J_{1a}}(x_j - 0.8\sin(x_1\pi)\cos(6\pi x_1 + \frac{j\pi}{30}))^2\right. \\
&\qquad \left. + \sum_{j\in J_{1b}}(x_j - 0.8\cos(x_1\pi))^2\right) \\
f_2(x) &= 1 - \sqrt{x_1} + \frac{2}{|J_{2a}| + |J_{2b}|}\left(\sum_{j\in J_{2a}}(x_j - 0.8\sin(x_1\pi)\sin(6\pi x_1 + \frac{j\pi}{30}))^2\right. \\
&\qquad \left. + \sum_{j\in J_{2b}}(x_j - 0.8\cos(x_1\pi))^2\right)
\end{aligned}
$$

where $x = (x_1, \ldots, x_{30})^T \in [0,1]\times[-1,1]^{29}$. $J_{1a} = \{4, 7, 11, \ldots, 28\}$, $J_{1b} = \{3, 9, 15, \ldots, 27\}$, $J_{2a} = \{2, 5, 8, \ldots, 29\}$, $J_{2b} = \{6, 12, 18, \ldots, 30\}$.

- F6

$$
\begin{aligned}
f_1(x) &= x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - 0.3x_1(x_1\cos(4(6\pi x_1 + \frac{j\pi}{30}) + 2))\cos(6\pi x_1 + \frac{j\pi}{30})\right)^2 \\
f_2(x) &= 1 - \sqrt{x_1} + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - 0.3x_1(x_1\cos(4(6\pi x_1 + \frac{j\pi}{30})) + 2))\sin(6\pi x_1 + \frac{j\pi}{30})\right)^2
\end{aligned}
$$

where $x = (x_1, \dots, x_{30})^T \in [0,1]\times[-1,1]^{29}$. $J_1 = \{2, 4, \dots, 30\}$ and $J_2 = \{3, 5, \dots, 29\}$.

- F7

$$\begin{aligned} f_1(x) &= x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\sqrt{j}\left(x_j - \sin(6\pi x_1 + \frac{j\pi}{30})\right)^2 \\ f_2(x) &= 1 - \sqrt{x_1} + \frac{2}{|J_1|}\sum_{j\in J_2}\sqrt{j}\left(x_j - \sin(6\pi x_1 + \frac{j\pi}{30})\right)^2 \end{aligned}$$

where $x = (x_1, \dots, x_{30})^T \in [0,1]\times[-1,1]^{29}$. $J_1 = \{2, 4, \dots, 30\}$ and $J_2 = \{3, 5, \dots, 29\}$.

- F8

$$\begin{aligned} f_1(x) &= \cos(0.5x_1\pi)\cos(0.5x_2\pi) + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - \frac{j}{10}x_1^2 - \frac{10-j}{10}x_2\right)^2 \\ f_2(x) &= \cos(0.5x_1\pi)\sin(0.5x_2\pi) + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - \frac{j}{10}x_1^2 - \frac{10-j}{10}x_2\right)^2 \\ f_3(x) &= \sin(0.5x_1\pi) + \frac{2}{|J_3|}\sum_{j\in J_3}\left(x_j - \frac{j}{10}x_1^2 - \frac{10-j}{10}x_2\right)^2 \end{aligned}$$

where $x = (x_1, \dots, x_{10})^T \in [0,1]^2 \times [-2,2]^8$, $J_1 = \{4, 7, 10\}$, $J_2 = \{5, 8\}$, and $J_3 = \{3, 6, 9\}$.

- F9

$$\begin{aligned} f_1(x) &= \cos(0.5x_1\pi)\cos(0.5x_2\pi) + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - 2x_2\sin(2\pi x_1 + \frac{j\pi}{10})\right)^2 \\ f_2(x) &= \cos(0.5x_1\pi)\sin(0.5x_2\pi) + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - 2x_2\sin(2\pi x_1 + \frac{j\pi}{10})\right)^2 \\ f_3(x) &= \sin(0.5x_1\pi) + \frac{2}{|J_3|}\sum_{j\in J_3}\left(x_j - 2x_2\sin(2\pi x_1 + \frac{j\pi}{10})\right)^2 \end{aligned}$$

where $x = (x_1, \dots, x_{10})^T \in [0,1]^2 \times [-2,2]^8$, $J_1 = \{4, 7, 10\}$, $J_2 = \{5, 8\}$, and $J_3 = \{3, 6, 9\}$.

- F10

$$\begin{aligned} f_1(x) &= x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}(4y_j^2 - \cos(8y_j\pi) + 1.0) \\ f_2(x) &= 1 - \sqrt{x_1} + \frac{2}{|J_2|}\sum_{j\in J_2}(4y_j^2 - \cos(8y_j\pi) + 1.0) \end{aligned}$$

where $x = (x_1, \dots, x_{10})^T \in [0,1]^{10}$. $J_1 = \{2, 4, \dots, 10\}$ and $J_2 = \{3, 5, \dots, 9\}$. $y_j = x_j - x_1^{0.5(1.0+\frac{3(j-2)}{10-2})}, j = 2, \dots, 10$.

- F11

$$\begin{aligned} f_1(x) &= x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\big(4y_j^2 - \cos(8y_j\pi) + 1.0\big) \\ f_2(x) &= 1 - \sqrt{x_1} + \frac{2}{|J_2|}\sum_{j\in J_2}\big(4y_j^2 - \cos(8y_j\pi) + 1.0\big) \end{aligned}$$

where $x = (x_1, \dots, x_{10})^T \in [0,1]^{10}$. $J_1 = \{2, 4, \dots, 10\}$ and $J_2 = \{3, 5, \dots, 9\}$. $y_j = x_j - \sin(6\pi x_1 + \frac{j\pi}{10})$, $j = 2, \dots, 10$.

- F12

$$\begin{aligned} f_1(x) &= x_1 + \frac{2}{|J_1|}\left(4\sum_{j\in J_1} y_j^2 - 2\prod_{j\in J_1}\cos(\frac{40y_j\pi}{\sqrt{j}}) + 2\right) \\ f_2(x) &= 1 - \sqrt{x_1} + \frac{2}{|J_2|}\left(4\sum_{j\in J_2} y_j^2 - 2\prod_{j\in J_2}\cos(\frac{40y_j\pi}{\sqrt{j}}) + 2\right) \end{aligned}$$

where $x = (x_1, \dots, x_{10})^T \in [0,1]^{10}$. $J_1 = \{2, 4, \dots, 10\}$ and $J_2 = \{3, 5, \dots, 9\}$. $y_j = x_j - x_1^{0.5(1.0+\frac{3(j-2)}{10-2})}, j = 2, \dots, 10$.

- F13

$$\begin{aligned} f_1(x) &= x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - \sin(6\pi x_1 + \frac{j\pi}{30})\right)^2 \\ f_2(x) &= 1 - x_1^2 + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - \sin(6\pi x_1 + \frac{j\pi}{30})\right)^2 \end{aligned}$$

where $x = (x_1, \dots, x_{30})^T \in [0,1]\times[-1,1]^{29}$. $J_1 = \{2, 4, \dots, 30\}$ and $J_2 = \{3, 5, \dots, 29\}$.

- F14

$$
\begin{aligned}
f_1(x) &= x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - \sin(6\pi x_1 + \frac{j\pi}{30})\right)^2 \\
f_2(x) &= 1 - \sqrt{x_1} - x_1\sin(10x_1^2\pi) + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - \sin(6\pi x_1 + \frac{j\pi}{30})\right)^2
\end{aligned}
$$

where $x = (x_1, \ldots, x_{30})^T \in [0,1]\times[-1,1]^{29}$. $J_1 = \{2, 4, \ldots, 30\}$ and $J_2 = \{3, 5, \ldots, 29\}$.

Appendix C

Simulated Binary Crossover and Polynomial Mutation

C.1 Simulated Binary Crossover

Deb et al. developed Simulated Binary Crossover (SBX) in [17]. SBX creates two offspring solutions ($y^{(1)}$ and $y^{(2)}$) by recombining two parent solutions ($x^{(1)}$ and $x^{(2)}$), where both the parent and offspring solutions belong to $\mathbb{R}^n$. For each $i = 1, \ldots, n$, the components of $y^{(1)}$ and $y^{(2)}$ are generated as follows.

$$y_i^{(1)} = 0.5\left[(1+\xi_i)x_i^{(1)} + (1-\xi_i)x_i^{(2)}\right] \tag{C.1}$$

$$y_i^{(2)} = 0.5\left[(1-\xi_i)x_i^{(1)} + (1+\xi_i))x_i^{(2)}\right] \tag{C.2}$$

with

$$\xi_i = \begin{cases} (2u_i)^{\frac{1}{\eta_c+1}} & \text{if } u_i \leq 0.5; \\ \left(\frac{1}{2(1-u_i)}\right)^{\frac{1}{\eta_c+1}} & \text{otherwise} \end{cases} \tag{C.3}$$

where u_i is a random number between $[0, 1]$ and $\eta_c \in \mathbb{R}^+$ is the distribution index. SBX with a large value of η_c is more likely to create 'near-parent' offspring solutions. On the contrary, a small value of η_c in SBX allows the generation of distant offspring solutions.

C.2 Polynomial Mutation

In polynomial mutation [15], an offspring solution $y \in \mathbb{R}^n$ is created by perturbing the parent solution $x \in \mathbb{R}^n$ as follows.

$$y_i = x_i + \delta_i(b_i - a_i), \;\; i = 1, \dots , n \tag{C.4}$$

with

$$\delta_i = \begin{cases} (2u_i)^{\frac{1}{\eta_m+1}} - 1, & \text{if } u_i < 0.5, \\ 1 - (2 - 2u_i)^{\frac{1}{\eta_m+1}}, & \text{otherwise} \end{cases} \tag{C.5}$$

where u_i is a random number in $[0, 1]$ and $\eta_m \in \mathbb{R}^+$ is the distribution index. b_i and a_i are the upper and lower bounds of x_i, respectively.

Appendix D

List of Publications

- Journal
  - Q. Zhang and H. Li, MOEA/D: A Multi-objective Evolutionary Algorithm Based on Decomposition, IEEE Transactions on Evolutionary Computation, in press, 2007.
- Conference Papers
  - H. Li and Q. Zhang, A Multi-objective Differential Evolution Based on Decomposition for Multi-objective Optimization with Variable Linkages, the 9th International Conference on Parallel Problem Solving from Nature - PPSN IX , Reykjavik 10-13 September, 2006.
  - H. Li and Q. Zhang, A Decomposition-based Evolutionary Strategy for Bi-objective LOTZ Problem, Artificial Intelligence and Simulation of Behaviour, Symposia-Nature Inspired Systems, 3-6 April, 2006, Bristol, UK.
  - H. Li, Q. Zhang, E.P. Tsang, and J.A. Ford, Hybrid Estimation of Distribution Algorithm for Multi-objective Knapsack Problem, the 4th European Conference

on Evolutionary Computation in Combinatorial Optimization, 5-7 April 2004, Coimbra, Portugal (This paper was nominated in Evocop2004 for Best Paper Awards).

- Q. Zhang, H. Li and E.P. Tsang, Estimation of Distribution Algorithm with Local Search for Approximating the Discrepancy of a Finite Set of Points, XVI Conference of the European Chapter on Combinatorial Optimisation, 5-7 June 2003, Molde, Norway.

Bibliography

[1] H. A. Abbass, R. Sarker, and C. Newton. PDE: A pareto-frontier differential evolution approach for multi-objective optimization problems. In *Proceedings of the 2001 Congress on Evolutionary Computation CEC2001*, pages 971–978, Seoul, Korea, 27-30 2001. IEEE Press.

[2] H. G. Beyer. *Theory of Evolution Strategies.* Springer, 2001.

[3] V.J. Bowman. Othe the relationship of the tchebycheff normand the efficient frontier of multiple-criteria objectives. *Multiple Criteria Decision Making*, pages 76–85, 1976.

[4] Z. Cai and Y. Wang. A multiobjective optimization-based evolutionary algorithm for constrained optimization. *Evolutionary Computation, IEEE Transactions on Posted online: 2006-11-30 10:48:44.0*, 10(6):658–675, 2006.

[5] M. Clerc. *Particle Swarm Optimization.* ISTE Publishing Company, 2006.

[6] C. A. Coello Coello. An updated survey of ga-based multiobjective optimization techniques. *ACM Comput. Surv.*, 32(2):109–143, 2000.

[7] C. A. Coello Coello. Guest editorial: special issue on evolutionary multiobjective optimization. *IEEE Trans. Evolutionary Computation*, 7(2):97–99, 2003.

[8] C. A. Coello Coello. Recent trends in evolutionary multiobjective optimization. In *Evolutionary Multiobjective Optimization: Theoretical Advances And Applications*, pages 7–32. Springer-Verlag, London, 2005.

[9] C. A. Coello Coello and G. Lamont. *Applications of Multi-Objective Evolutionary Algorithms.* World Scientific Printers, Singapore, 2004.

[10] D. Corne, M. Dorigo, F. Glover, D. Dasgupta, P. Moscato, R. Poli, and K. V. Price. *New ideas in optimization.* McGraw-Hill Ltd., UK, Maidenhead, UK, England, 1999.

[11] I. L. L. Cruz, G. V. Willigenburg, and G. V. Straten. Efficient differential evolution algorithms for multimodal optimal control problems. *Appl. Soft Comput*, 3(2):97–122, 2003.

[12] P. Czyzak and A. Jaszkiewicz. Pareto simulated annealing – A metaheuristic technique for multiple objective combinatorial optimization. *Journal of Multi-Criteria Decision Analysis*, 7(1):34–47, 1998.

[13] I. Das and J. E. Dennis. Normal-bounday intersection: A new method for generating pareto optimal points in multicriteria optimization problems. *SIAM Journal on Optimization*, 8(3):631–657, August 1998.

[14] K. Deb. Multi-objective genetic algorithms: Problem difficulties and construction of test problems. *Evolutionary Computation*, 7(3):205–230, 1999.

[15] K. Deb. *Multi-Objective Optimization Using Evolutionary Algorithms.* John Wiley & Sons, Inc., 2001.

[16] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evolutionary Computation*, 6(2):182–197, 2002.

[17] K. Deb and A. Kumar. Real-coded genetic algorithms with simulated binary crossover: Studies on multimodal and multiobjective problems. *Complex Systems*, 9(6):431–454, August 1995.

[18] K. Deb, A. Sinha, and S. Kukkonen. Multi-objective test problems, linkages, and evolutionary methodologies. In *GECCO*, pages 1141–1148, 2006.

[19] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable multi-objective optimization test problems. In *CEC '02: Congress on Evolutionary Computation*, volume 1, pages 825–830, Piscataway, New Jersey, 2002. IEEE Service Center.

[20] M. Ehrgott. *Multicriteria Optimization.* Springer, Second edition, 2005.

[21] M. Farina, K. Deb, and P. Amato. Dynamic multiobjective optimization problems: test cases, approximations, and applications. *IEEE Trans. Evolutionary Computation*, 8(5):425–442, 2004.

[22] C. M. Fonseca and P. J. Fleming. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In S. Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423, , San Mateo, California, 1993. University of Illinois at Urbana-Champaign, Morgan Kaufman, San Francisco, CA.

[23] C. M. Fonseca and P. J. Fleming. Multiobjective optimization and multiple constraint handling with evolutionary algorithms. i. a unified formulation. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, 28(1):26–37, 1998.

[24] X. Gandibleux and M. Ehrgott. 1984-2004 – 20 years of multiobjective metaheuristics. but what about the solution of combinatorial problems with multiple objectives? In Carlos A. Coello Coello, Arturo Hernndez Aguirre, and Eckart Zitzler, editors, *Evolutionary Multi-Criterion Optimization. Third International Conference, EMO 2005, pp. 33–46, Springer. Lecture Notes in*, volume 3410, Guanajuato, Mxico, March 2005.

[25] X. Gandibleux and A. Freville. Tabu Search Based Procedure for Solving the 0-1 Multi-Objective Knapsack Problem: The Two Objectives Case. *Journal of Heuristics*, 6(3):361–383, AUG 2000.

[26] S. Gass and T. Saaty. The computational algorithm for the parameteric objective function. *Naval Research Logistics Quarterly*, 2:39–45, 1955.

[27] M. Gen and R. Cheng. *Genetic Algorithms and Engineering Design.* John Wiley & Sons, New York, 1997.

[28] F. Glover and M. Laguna. *Tabu Search.* Kluwer, Norwell, MA, 1997.

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

[30] P. H. Hanssen. Tabu search for multiobjective optimization: MOTS. In *Proceedings of the 13th International Conference on Multiple Criteria Decision Making*, Cape Town, South Africa, 1997.

[31] T. Higuchi, S. Tsutsui, and M.i Yamamura. Theoretical analysis of simplex crossover for real-coded genetic algorithms. In *PPSN*, pages 365–374, 2000.

[32] J. Horn, N. Nafpliotis, and D. E. Goldberg. A niched pareto genetic algorithm for multiobjective optimization. In *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, pages 82–87, Piscataway, New jersey, June 1994. IEEE Service Center.

[33] S. Huband, P. Hingston, L. Barone, and L. While. A review of multiobjective test problems and a scalable test problem toolkit. *IEEE Transactions on Evolutionary Computation*, 10(5):477–506, October 2006.

[34] E. J. Hughes. Multiple single objective pareto sampling. In *CEC '03: Congress on Evolutionary Computation*, volume 4, pages 2678–2684, Canberra, 2003.

[35] A. W. Iorio and X. Li. Incorporating directional information within a differential evolution algorithm for multi-objective optimization. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 691–698, Seattle, Washington, USA, 2006. ACM Press.

[36] Antony W. Iorio and X. Li. Solving rotated multi-objective optimization problems using differential evolution. In G.I. Webb and Xinghuo Yu, editors, *Proceeding of the 17th Joint Australian Conference on Artificial Intelligence*, Lecture Notes in Computer Science, pages 861–872, 2004.

[37] H. Ishibuchi and S. Kaige. Implementation of simple multiobjective memetic algorithms and its applications to knapsack problems. *Int. J. Hybrid Intell. Syst.*, 1(1):22–35, 2004.

[38] H. Ishibuchi and T. Murata. Multi-objective genetic local search algorithm and its application to flowshop scheduling. *IEEE Transactions on Systems, Man and Cybernetics*, 28(3):392–403, August 1998.

[39] H. Ishibuchi, K. Narukawa, N. Taukamoto, and Y. Nojima. An empirical study on the similarity-based mating for evolutionary multiobjective combinatorial optimization. *European Journal of Operational Research*, 2007, in press.

[40] H. Ishibuchi and Y. Shibata. An empirical study on the effect of mating restriction on the search ability of emo algorithms. In Carlos M. Fonseca, Peter J. Fleming, Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele, editors, *EMO '03: Evolutionary Multicriterion OPtimization*, pages 433–447. Springer, 2003.

[41] H. Ishibuchi, T. Yoshida, and T. Murata. Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling. *IEEE Trans. Evolutionary Computation*, 7(2):204–223, 2003.

[42] A. Jaszkiewicz. Comparison of local search-based metaheuristics on the multiple objective knapsack problem. *Foundations of computer and decision sciences*, 26(1):99–120, 2001.

[43] A. Jaszkiewicz. *Multiple Objective Metaheuristic Algorithms for Combinatorial Optimization.* PhD thesis, Poznan University of Technology, Poznan, 2001.

[44] A. Jaszkiewicz. On the performance of multiple-objective genetic local search on the 0/1 knapsack problem - a comparative experiment. *IEEE Trans. Evolutionary Computation*, 6(4):402–412, 2002.

[45] Y. Jin, T. Okabe, and B. Sendhoff. Adapting weighted aggregation for multiobjective evolutionary strategies. In *EMO '01: Evolutionary Multicriterion OPtimization*, pages 96–110, Springer LNCS 1993, 2001.

[46] Y. Jin and B. Sendhoff. Connectedness, regularity and the success of local search in evolutionary multi-objective optimization. In 8-12 Dec. 2003 Page(s): 1910 1917 Vol.3 Volume 3, Issue, editor, *The 2003 Congress on Evolutionary Computation*, volume 3, pages 1910 – 1917, Dec 2003.

[47] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[48] J. D. Knowles. *Local-Search and Hybrid Evolutionary Algorithms for Pareto Optimization.* PhD thesis, Department of Computer Science, University of Reading, 2002.

[49] J. D. Knowles and D. Corne. M-PAES: a memetic algorithm for multiobjective optimization. In *CEC '00 Congress on Evolutionary Computation*, pages 103–108, Piscataway, NJ, July 2000. IEEE Press.

[50] J. D. Knowles and D. Corne. Properties of an adaptive archiving algorithm for storing nondominated vectors. *IEEE Trans. Evolutionary Computation*, 7(2):100–116, 2003.

[51] J. D. Knowles and D. Corne. Memetic algorithms for multiobjective optimization: issues, methods and prospects. In *Recent Advances in Memetic Algorithms*, pages 313–352. Springer, 2004.

[52] J. D. Knowles and D. W. Corne. Local search, multiobjective optimization and the pareto archived evolution strategy. In *Proceedings of Third Australia-Japan Joint Workshop on Intelligent and Evolutionary Systems*, pages 209–216, Ashikaga, Japan, November 1999.

[53] J. D. Knowles and D. W. Corne. The pareto archived evolution strategy: A new baseline algorithm for multiobjective optimisation. In *CEC '99: Congress on Evolutionary Computation*, pages 98–105, Washington D.C., July 1999. IEEE Service Center.

[54] N. Krasnogor and J. Smith. A tutorial for competent memetic algorithms: model, taxonomy, and design issues. *IEEE Trans. Evolutionary Computation*, 9(5):474–488, 2005.

[55] S. Kukkonen and J. Lampinen. GDE3: The third evolution step of generalized differential evolution. In *CEC '05: Congress on Evolutionary Computation*, pages 239–246, Edinburgh, UK, September 2005. IEEE service Center.

[56] W. B. Langdon and R. Poli. *Foundations of Genetic Programming*. Springer-Verlag, 2002.

[57] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler. Combining convergence and diversity in evolutionary multi-objective optimization. *Evolutionary Computation*, 10(3):263282, 2002.

[58] Y. W. Leung and Y. Wang. Multiobjective programming using uniform design and genetic algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 30(3):293–304, 2000.

[59] H. Li and Q. Zhang. A multi-objective differential evolution based on decomposition for multiobjective optimization with variable linkages. In *PPSN*, pages 583–592, 2006.

[60] H. Li, Q. Zhang, E. P. K. Tsang, and John A. Ford. Hybrid estimation of distribution algorithm for multiobjective knapsack problem. In *EvoCOP '4: the 4th European Conference on Evolutionary Computation in Combinatorial Optimization*, pages 145–154, 2004.

[61] H. Lu and Gary G. Yen. Rank-density-based multiobjective genetic algorithm and benchmarktest function study. *IEEE Trans. Evolutionary Computation*, 7(4):325–343, 2003.

[62] N. K. Madavan. Multiobjective optimization using a pareto differential evolution approach. In *CEC '02: Congress on Evolutionary Computation*, pages 1145–1150, Honolulu, Hawaii, may 2002.

[63] P. Merz. *Memetic Algorithms for Combinatorial Optimization Problems: Fitness Landscapes and Effective Search Strategies*. PhD thesis, University of Siegen, Germany, 2000.

[64] A. Messac, A. Ismail-Yahaya, and C. A. Mattson. The normalized normal constraint method for generating the pareto frontier. *Structural and Multidisciplinary Optimization*, 25(2):86–98, 2003.

[65] Z. Michalewicz and J. Arabas. Genetic algorithms for the 0/1 knapsack problem. In *ISMIS '94: Proceedings of the 8th International Symposium on Methodologies for Intelligent Systems*, pages 134–143, London, UK, 1994. Springer-Verlag.

[66] K. Miettinen. *Nonlinear Multiobjective Optimization.* Kluwer Academic Publishers, 1999.

[67] T. Murata, H. Ishibuchi, and M. Gen. Specification of genetic search directions in cellular multi-objective genetic algorithms. In *EMO*, pages 82–95, 2001.

[68] A. C. Nearchou and Sotiris L. Omirou. Differential evolution for sequencing and scheduling optimization. *J. Heuristics*, 12(6):395–411, 2006.

[69] T. Okabe, Y. Jin, M. Olhofer, and B. Sendhoff. On test functions for evolutionary multi-objective optimization. In *PPSN VIII : the 8th International Conference of Parallel Problem Solving from Nature*, pages 792–802, Birmingham, 2004. Springer, Berlin.

[70] T. Okabe, Y. Jin, and B. Sendhoff. A Critical Survey of Performance Indices for Multi-Objective Optimisation. In *CEC '03: Proceedings of IEEE Congress on Evolutionary Computation*, pages 878–885. IEEE Press, 2003.

[71] T. Okabe, Y. Jin, B. Sendhoff, and M. Olhofer. Voronoi-based estimation of distribution algorithm for multi-objective optimization. In Y. Shi, editor, *CEC '04: Congress on Evolutionary Computation*, pages 1594–1602, Portland, 2004. IEEE.

[72] I. Ono and S. Kobayashi. A real-coded genetic algorithm for function optimization using unimodal normal distribution crossover. In *7th Int'l Conf. on Genetic Algorithms*, pages 246–253, 1997.

[73] L. Paquete and T. Stützle. A two-phase local search for the biobjective traveling salesman problem. In *EMO '03: Evolutionary Multicriterion OPtimization*, pages 479–493, 2003.

[74] I. C. Parmee and D. Cvetkovic. Preferences and their application in evolutionary multiobjective optimization. *IEEE Trans. Evolutionary Computation*, 6(1):42–57, 2002.

[75] L. Pedro and J. A. Lozano. *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation.* Springer, 2001.

[76] K. Price, R. M. Storn, and J.i A. Lampinen. *Differential Evolution: A Practical Approach to Global Optimization (Natural Computing Series).* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

[77] T. Robic and B. Filipic. DEMO: Differential evolution for multiobjective optimization. In *EMO '05: Evolutionary Multicriterion OPtimization*, pages 520–533, 2005.

[78] M. Sakawa. *Genetic Algorithms and Fuzzy Multiobjective Optimization.* Hardcover, 2001.

[79] J.R. Schott. *Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization.* PhD thesis, Massachusetts Institute of Technology, Boston, 1995.

[80] D. Sharma, A. Kumar, K. Deb, and K. Sindhya. Hybridization of SBX based NSGA-II and sequential quadratic programming for solving multi-objective optimization problems. Technical report, KanGAL Report No. 2007007, May 2007.

[81] J. D. Landa Silva, E. K. Burke, and S. Petrovic. *An Introduction to Multiobjective Metaheuristics for Scheduling and Timetabling*, volume 535 of *Lecture Notes in Economics and Mathematical Systems*, chapter Metaheuristic for Multiobjective Optimisation, pages 91–129. Springer, 2004.

[82] N. Srinivas and K. Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248, 1994.

[83] R. E. Steuer. *Multiple Criteria Optimization: Theory, Computation and Application.* John Wiley & Sons, New York, 1986.

[84] R. Storn and K. Price. Differential evolution: A simple and efficient adaptive scheme for global optimization over continuous spaces. *Journal of Global Optimization*, 11:341–359, 1997.

[85] J. Sun, Q. Zhang, and E. P. K. Tsang. DE/EDA: a new evolutionary algorithm for global optimization. *Inf. Sci. Inf. Comput. Sci.*, 169(3-4):249–262, 2005.

[86] J. Y. Sun. *Hybrid Esitimation of Distribution Algorithm for Global Optimization.* PhD thesis, Department of Computer Science, University of Essex, 2005.

[87] K. C. Tan, E. F. Khor, and T.H. Lee. *Multiobjective Evolutionary Algorithms and Applications.* Series: Advanced Information and Knowledge Processing, Springer Verlag, 2005.

[88] V. T'kindt and J.-C. Billaut. *Multicriteria Scheduling, Theory, Models and Algorithms.* Springer, 2006.

[89] E.L. Ulungu, J. Teghem, P. Fortemps, and D. Tuyttens. MOSA method: A tool for solving multiobjective combinatorial optimization problems. *Journal of Multi-Criteria Decision Analysis*, 8:221–236, 1999.

[90] D. A. V. Veldhuizen and G. B. Lamont. Multiobjective evolutionary algorithms: Analyzing the state-of-the-art. *Evolutionary Computation*, 8(2):125–147, 2000.

[91] D. A. Van Veldhuizen. *Multiobjective Evolutionary Algorithms: Classification, Analyses, and New Innovations.* PhD thesis, Air Force Institute of Technology, Air University, 1999.

[92] C. Voudouris and E.P.K Tsang. Guided local search and its application to the travelling salesman problem. *European Journal of Operational Research*, 113(2):469–499, 1999.

[93] F. Xue, A. C. Sanderson, and R. J. Graves. Pareto-based multi-objective differential evolution. In Ruhul Sarker, Robert Reynolds, Hussein Abbass, Kay Chen Tan, Bob McKay, Daryl Essam, and Tom Gedeon, editors, *CEC '03: Proceedings of the 2003 Congress on Evolutionary Computation*, pages 862–869, Canberra, 8-12 December 2003. IEEE Press.

[94] L. Zadeh. Optimality and non-scalar-valued performance criteria. *IEEE Transactions on Automatics Control*, 8:59–60, 1963.

[95] Q. Zhang. On stability of fixed points of limit models of univariate marginal distribution algorithm and factorized distribution algorithm. *IEEE Trans. Evolutionary Computation*, 8(1):80–93, 2004.

[96] Q. Zhang and Y. W. Leung. Orthogonal genetic algorithm for multimedia multicast routing. *IEEE Trans. Evolutionary Computation*, 3(1):53–62, 1999.

[97] Q. Zhang and H. Mühlenbein. On the convergence of a class of estimation of distribution algorithms. *IEEE Trans. Evolutionary Computation*, 8(2):127–136, 2004.

[98] Q. Zhang, J. Sun, and E.P.K. Tsang. An evolutionary algorithm with guided mutation for the maximum clique problem. *IEEE Transactions on Evolutionary Computation*, 9(2):192–201, 2005.

[99] Q. Zhang, J. Sun, E.P.K. Tsang, and J.A. Ford. Hybrid estimation of distribution algorithm for global optimization. *Engineering Computations*, 21(1):91–107, 2003.

[100] Q. Zhang, A. Zhou, and Y. Jin. RM-MEDA: Modelling the regularity in an estimation of distribution algorithm for continuous multiobjective optimisation with variable linkages. *IEEE Trans. on Evolutionary Computation*, 2007. in press.

[101] A. Zhou, Q. Zhang, and Y.C. Jin. Generating representative pareto optimal solutions in both the objective and decision space by using an enhanced rm-meda. Technical report, Department of Computer Science, University of Essex, 2007.

[102] E. Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications.* PhD thesis, Swiss Federal Institute of Technology (ETH) Zurich, 1999.

[103] E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8(2):173–195, 2000.

[104] E. Zitzler. and S. Kunzli. Indicator-based selection in multiobjective search. In et al. Yao, X., editor, *8th Intl Conf. on Parallel Problem Solving from Nature (PPSN VIII)*, pages 832–842, UK, 2004. Springer.

[105] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the strength pareto evolutionary algorithm for multiobjective optimization. In K. C. Giannakoglou, D. T. Tsahalis, J. Périaux, K. D. Papailiou, and T. Fogarty, editors, *Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems*, pages 95–100, Athens, Greece, 2002.

[106] E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Trans. Evolutionary Computation*, 3(4):257–271, 1999.

[107] E. Zitzler, L. Thiele, M. Laumanns, Carlos M. Fonseca, and Viviane Grunert da Fonseca. Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Trans. Evolutionary Computation*, 7(2):117–132, 2003.