Hindawi
Complexity
Volume 2018, Article ID 3051854, 16 pages
https://doi.org/10.1155/2018/3051854

Research Article

An Improved MOEA/D Based on Reference Distance for Software Project Portfolio Optimization

Jing Xiao,[1] **Jing-Jing Li**,[1] **Xi-Xi Hong**,[1] **Min-Mei Huang**,[2] **Xiao-Min Hu**,[3] **Yong Tang**,[1] **and Chang-Qin Huang**[4]

[1]*School of Computer Science, South China Normal University, Guangzhou, China*
[2]*School of Public Administration, South China Normal University, Guangzhou, China*
[3]*School of Computers, Guangdong University of Technology, Guangzhou, China*
[4]*School of Information Technology in Education, South China Normal University, Guangzhou, China*

Correspondence should be addressed to Jing-Jing Li; jingjing.li1124@gmail.com

Received 29 December 2017; Revised 14 April 2018; Accepted 17 April 2018; Published 30 May 2018

Academic Editor: Jesus Medina

As it is becoming extremely competitive in software industry, large software companies have to select their project portfolio to gain maximum return with limited resources under many constraints. Project portfolio optimization using multiobjective evolutionary algorithms is promising because they can provide solutions on the Pareto-optimal front that are difficult to be obtained by manual approaches. In this paper, we propose an improved MOEA/D (multiobjective evolutionary algorithm based on decomposition) based on reference distance (MOEA/D_RD) to solve the software project portfolio optimization problems with optimizing 2, 3, and 4 objectives. MOEA/D_RD replaces solutions based on reference distance during evolution process. Experimental comparison and analysis are performed among MOEA/D_RD and several state-of-the-art multiobjective evolutionary algorithms, that is, MOEA/D, nondominated sorting genetic algorithm II (NSGA2), and nondominated sorting genetic algorithm III (NSGA3). The results show that MOEA/D_RD and NSGA2 can solve the software project portfolio optimization problem more effectively. For 4-objective optimization problem, MOEA/D_RD is the most efficient algorithm compared with MOEA/D, NSGA2, and NSGA3 in terms of coverage, distribution, and stability of solutions.

1. Introduction

Project portfolio management (PPM) is a management process to help project managers to analyze and acquire all information of current proposed projects. PPM helps decision makers to sort and prioritize each project according to certain criteria, such as business goals, strategic value, cost, and resource constraints. A key step of PPM is to decide which projects to invest in an optimal manner. Project portfolio optimization (PPO) is the effort to make the best decisions to select the best mix of projects from all candidate projects. Manual approaches include PPO include Q-Sort, analytic hierarchy process, and portfolio matrices [1–3]. These approaches are time-consuming and limited to the number of projects they can deal with. The project portfolio problem may be dealt as a multiobjective optimization problem, and it is difficult to tackle [4, 5]. Software managers and researchers used branch-and-bound approach, simulated annealing and Tabu search, and so on to obtain the uniformly distributed Pareto-optimal solutions [6–8]. It is hard to find an algorithm to deal with this problem efficiently when the complexity of the problem grows exponentially with the number of projects.

Within this context, multiobjective evolutionary algorithms (MOEA) [9] which can obtain Pareto-optimal solutions are promising to solve the project portfolio optimization problem [10–12]. Pareto front-based MOEAs are superior to manual approaches in a way that they are able to create a set of efficient portfolios, for which it can be assured that there exist no solutions in the search space that promise better values in at least one of the objectives and offer at least the same values in all the other objectives [5].

MOEAs can obtain approximate optimal solutions. Furthermore, MOEAs can deal with the computational complexity with an increasing number of projects. That is why there are lots of publications devoted to solving portfolio optimization problems using MOEAs and also there are many applications of MOEAs in finance and economics areas [13].

Compared with general project portfolio optimization using MOEAs, the number of publications dedicated to the MOEAs' applications to software project portfolio problems is scarce. Kremmel et al. [5] introduced a multiobjective evolutionary approach, mPOEMS, to find the Pareto-optimal front for software project portfolio optimization problem. However, the paper only studied 2-objective optimization. In this paper, we first propose an improved MOEA/D [14] algorithm based on reference distance (MOEA/D_RD) to alleviate the inefficiency of MOEA/D's weighted sum approach. Then, we use MOEA/D_RD to solve the 2-, 3-, and 4-objective software portfolio optimization problem. Comparison and analysis experiments are conducted among MOEA/D_RD, MOEA/D, NSGA2 [15], and NSGA3 [16].

The rest of this paper is organized as follows. Section 2 discusses the related work of portfolio optimization using evolutionary algorithms. Section 3 describes the software portfolio selection model we have used. The proposed MOEA/D_RD is explained in detail in Section 4, and the empirical experiments are described and discussed in Section 5. The last section gives the conclusion and lists the future work.

2. Related Work

The first formalization of methodology for solving portfolio optimization problems was proposed by Markowitz [17] in the 1950s. Markowitz defined a portfolio as a vector of real numbers that contains the weight corresponding to each available asset and stated that the investor searches the portfolio that minimizes the risk while maximizing the return ideally. However, with the increasing number of projects and many constraints in real world, the simple assumptions in Markowitz model are infeasible and it is hard to find an exact algorithm to deal with the problem. As such, the first use of genetic algorithm (GA) for optimizing project portfolio was proposed by Arnone et al. in 1993 [18]. The authors divided the population of a GA into different subpopulations and produced different portions of the Pareto front.

An obvious advantage of MOEAs is their ability to produce, in one single run, a complete approximation of the Pareto front. MOEAs are suitable to solve the portfolio optimization problem since the aim of the problem is to provide a set of Pareto front solutions, that is, the best possible tradeoffs among the objectives, among which the managers can choose the most appropriate solution. In [19], the Markowitz model was solved with an MOEA in which the selection is carried out through a Pareto-ranking procedure. The authors used Sharpe's ratio instead of the classical density estimators such as crowding distance to break ties between solutions from the same Pareto front. Lin et al. [20] implemented integer encoding, simulated binary crossover, and parameter-based mutation within the NSGA2 to solve the investment portfolio optimization problem with fixed transaction costs and minimum lots. Subbu et al. [21] combined a Pareto-sorting evolutionary algorithm with linear programming for investment portfolio optimization. The Pareto-sorting evolutionary algorithm is used to retain the nondominated solutions found along the search by a small population size and an archive. Branke et al. [22] combined NSGA2 with the critical line algorithm to obtain a continuous Pareto front for portfolio optimization. NSGA2 was first employed to define convex subset of the original search space, then the critical line algorithm was applied on every subset to form the complete Pareto front. Bradshaw et al. [23] employed an evolutionary algorithm similar to SPEA2 [24] to solving the portfolio optimization problem. In [25], the authors compared six MOEAs on the classical Markowitz model. The results showed that SPEA2 and NSGA2 performed more effectively among the six studied algorithms. [26] compared three MOEAs, that is, NSGA2, SPEA2, and PESA [27], to solve the Markowitz model with three objectives: return value, risk, and number of assets in the portfolio and found that SPEA2 can obtain the best performance for the test cases.

Aforementioned work is based on Markowitz mean-variance model, and there are also a few publications devoted to other portfolio optimization models using MOEAs. Khalili-Damghani et al. [28] presented a hybrid fuzzy rule-based multiobjective framework for sustainable project portfolio selection. NSGA2 was applied to obtain the nondominated solutions. The proposed framework simultaneously considered the accuracy maximization and the complexity minimization objectives. Fernandez et al. [29] proposed a nonoutranked ant colony optimization II method for optimizing portfolio problem. The method incorporates integer linear programming to avoid clearly suboptimal regions in the search space and a priori preference system to focus the algorithmic effort on the most preferred region in the search space. Doerner et al. [4] introduced a Pareto ant colony optimization algorithm for solving the portfolio selection problem. Tofighian and Naderi [30] employed an ant colony optimization algorithm for solving the project selection and scheduling to optimize both total expected benefit and resource usage variation. Mavrotas et al. [31] studied the robustness analysis methodology for multiobjective project selection optimization.

Relatively speaking, the publication with respect to software project management using MOEAs is scarce. Rodríguez et al. [32] employed NSGA2 and a system dynamics simulation model to generate the Pareto front needed by software project managers to find the best values for initial team size and schedule estimates for a given project with the optimal cost, time, and productivity. Gueorguiev et al. [33] formulated software project planning problem as biobjective optimization. Robustness and complete time are treated as two competing objectives, and SPEA2 was employed to obtain the Pareto solutions. The most closely related to this paper is the work by Kremmel et al. [5] in which the authors used Constructive Cost Model II (COCOMO II) [34] and a multiobjective evolutionary algorithm to find the Pareto front for software project portfolio optimization. Only 2-objective Pareto front solutions were studied in Kremmel's work. In this paper, we extend Kremmel's work and propose an improved MOEA/D algorithm called MOEA/D_RD for software project portfolio optimization. Optimization problems with 2, 3, and 4 objectives are studied using 50 projects that follow the validated COCOMO II model criteria, and the proposed approach is compared with several state-of-the-art evolutionary algorithms.

The next section presents the list of goals borrowed from Kremmel's software portfolio selection model [5]. We only use the first 4 objectives and use the synergy goal as a constraint in our framework.

3. Software Portfolio Selection Model

Generally, a multiobjective optimization problem can be presented as the following:

Find a vector $x \in \Omega$, Ω is decision (variable) space.

$$\text{maximize} \quad \mathbf{Q}(x) = (q_1(x), q_2(x), \ldots, q_n(x)), \quad \mathbf{Q} : \Omega \rightarrow R^n, \tag{1}$$

under some constraints, where Ω is the decision (variable) space, $\mathbf{Q}$ consists of n real-valued objective functions, and R^n is called the objective space.

Suppose there are two solutions $u, v \in R^n$; u is said to dominate v if and only if $u_i \geq v_i$ for every $i \in \{1, 2, \ldots, \mathrm{n}\}$ and $u_j > v_j$ for at least one index $j \in \{1, 2, \ldots, \mathrm{n}\}$. A point $x^* \in \Omega$ is Pareto optimal to (1) if there is no point $x \in \Omega$ such that $\mathbf{Q}(x)$ dominates $\mathbf{Q}(x^*)$. $\mathbf{Q}(x^*)$ is then called a Pareto-optimal objective vector. In other words, any improvement in a Pareto-optimal point in one objective must lead to deterioration in at least one other objective. The set of all the Pareto-optimal points is called Pareto set, and the set of all the Pareto-optimal objective vectors is the Pareto front (PF).

Specifically, a solution for software project portfolio optimization is represented by a vector with the length of the maximum available projects. The task can be formalized as follows:

Find a vector $x = (x_1, x_2, \ldots, x_p) \in M_1 \times \cdots \times M_p$, where $M_i \subseteq M$, $M = \{0, 1, 2, \ldots, T \times 12\}$, such that the objective vector $\mathbf{y} = (q_1(x), q_2(x), \ldots, q_n(x))$ is maximum, where x_i is greater than 0 if project i is selected, and 0 if not; p is the number of candidate projects; M is a set of the number of months in the planning horizon; M_i is the months in which project i can start; T is the number of timeframes in the planning horizon. $q_i(x)$ is the ith optimization objective. In this work, we have considered the first 4 objectives defined in Kremmel's model, and thus the value of n is 4. These 4 objectives are defined as follows:

(1) Potential revenue $(q_1(x))$. Software project investors invest human resources, knowledge, and money into a project, with the goal of obtaining benefits from this investment. The potential projects for the project portfolio have to be evaluated with regard to their potential financial revenue. Thus, the first objective deals with the need to maximize potential overall portfolio return. It is calculated as the following:

$$q_1(x) = \sum_{i=1}^{p} r_i \cdot w_i, \tag{2}$$

where r_i is the potential revenue of project i, and w_i is 1 if $x_i > 0$ and 0 if $x_i = 0$. Obviously, the greater the overall potential revenue, the better the solution is.

(2) Strategic alignment $(q_2(x))$. Project selection optimization has to consider the problems with little commitment from business leaders, poor alignment of projects to strategy, little coordination between projects, and conflicting project objectives. The strategic alignment on the portfolio level should be maximized. It is calculated as follows:

$$q_2(x) = \sum_{i=1}^{p} a_i \cdot w_i, \tag{3}$$

where a_i is the strategic alignment value of project i, and w_i is 1 if $x_i > 0$ and 0 if $x_i = 0$. The greater the overall strategic value, the better the solution is.

(3) Resource usage distribution metric $(q_3(x))$. Resources in each timeframe are limited. This objective is to maximize the resource usage per timeframe and at the same time have the best distribution among the timeframe. Its value is between 0 and 1, where 1 means full resource consumption in each timeframe and 0 means that, at least in one timeframe, there is no resource consumed. Thus, the objective function to maximize is expressed as follows:

$$q_3(x) = \prod_{t=1}^{T} \left(\frac{\sum_{o=1}^{l} \sum_{i=1}^{p} r_{o,t,i} \cdot w_i}{\sum_{o=1}^{t} R_{o,t}} \right), \tag{4}$$

where o is the type of a resource (there are l different resource types); t is the timeframe; T is the number of timeframes in the planning horizon; $r_{o,t,i}$ is the type o resource consumption of project i in timeframe t, and $R_{o,t}$ is the type o resource limit in timeframe t. The closer the $q_3(x)$ is to one, the better the solution is.

(4) Risk $(q_4(x))$. The risk objective is calculated as follows:

$$q_4(x) = 1 - \frac{1}{p} \sum_{i=1}^{p} \text{risk}_i(x) \cdot w_i, \tag{5}$$

where $\text{risk}_i(x)$ is the risk value of project i. The closer the $q_4(x)$ is to one, the better the solution is.

The constraints we have used are listed as follows:

(a) Project starting timeframes. Most projects cannot start in an arbitrary timeframe, but very often in a few distinct timeframes. It is also possible that a project can only start in one timeframe in order to meet a special market opportunity. A feasible solution must adhere to the constraint of project starting time.

(b) The "must-select" restriction. Due to the legal and economic circumstances, a project may have to be included in a valid portfolio. Therefore, it should be possible to define a "must-select" restriction for portfolio optimization.

(c) Logical relationships. There are several logical relationships between projects such as linear, dependent, and mutually exclusive relationships. Linear relationship means if a certain project is selected for a portfolio, one or more predecessor projects must be selected obligatorily. If two projects are dependent, it means that the two projects must be selected to a portfolio together. On the contrary, two projects may not be selected for the same portfolio and thus are mutually exclusive.

(d) Synergy effects. The synergy effect constraint is one of the objectives in the original Kremmel's model. We consider it as a constraint when we optimize the first objective, that is, the potential revenue. If two projects are selected for the same portfolio, the total revenue could be more than the sum of the two project's revenues or less than the sum. The synergy effects are also considered in the Pareto ant colony optimization approach presented in [4].

In this paper, we consider the aforementioned four objectives and the four constraints for the software project portfolio problem. In the next section, we introduce the algorithm called MOEA/D_RD to solve the multiobjective optimization problem for software project selection.

4. An Improved MOEA/D Based on Reference Distance (MOEA/D_RD)

4.1. MOEA/D Based on Weighted Sum Approach. In this paper, we improve the weighted sum approach in MOEA/D algorithm [14] for solving the software project selection optimization problem. The approach considers a convex combination of the different objectives. Let $\lambda = (\lambda_1, \dots, \lambda_m)^T$ be a weight vector; m is the number of objectives; $f_i(x)$ is the ith objective to be optimized; and $\lambda_i \geq 0$ for all $i = 1, \dots, m$ and $\sum_{i=1}^{m} \lambda_i = 1$. Then the optimal solution to the following scalar optimization problem

$$\text{maximize} \quad g^{ws}(x \mid \lambda) = \sum_{i=1}^{m} \lambda_i f_i(x) \quad \text{subject to } x \in \Omega \qquad (6)$$

is a Pareto-optimal point to (1) as we can see that $q_i(x)$ corresponds to one of the objectives $f_i(x)$ in (6), where we use $g^{ws}(x \mid \lambda)$ to emphasize that λ is a coefficient vector in this objective function, where x is the variables to be optimized. To generate a set of different Pareto-optimal vectors, one can use different weight vectors $\lambda^1, \lambda^2, \dots, \lambda^N$ in the above scalar optimization problem and the optimized problem is divided into N subproblems. The greater N is, the wider the search space is. However, the weighted sum-based MOEA/D has several drawbacks and we illustrate them as follows.

Given an example, as shown in Figure 1, f_1 and f_2 are two objectives; $F(x_i)$ is the objective function of solution x_i; PF is the assumed optimal Pareto front; x_1, x_2, and x_3 are three solutions corresponding to weight vectors λ^1, λ^2, and λ^3. The ideal case is that the algorithm moves x_1, x_2, and x_3 to meet the PF. MOEA/D randomly picks up two solutions from the neighborhood of x_2 and generates a new solution using genetic operators. If the fitness value of the new solution is better than x_2, then x_2 is replaced by the new solution. If the new solution is fallen in the overlapping area of the search spaces of neighboring solutions x_1 and x_2, then both x_1 and x_2 are replaced by the new solution. The strategy is efficient at the earlier search stage of the algorithm, and it can make the search direction move fast to the PF. But at the late stage of the algorithm, as shown in Figure 2, there is no overlapping area among most of the search spaces of solutions. The neighboring solutions of x_2 cannot generate a new effective solution. The search process would stagnate at the late stage of the algorithm.

If the PF is a line, as shown in Figure 3, for the weight vector λ^2, all solutions on the PF line are the same optimal solutions with the same fitness values. Among the solutions between the weight vectors λ^1 and λ^2, the optimal solution is the intersection point of PF and f_2. Similarly, among the solutions between the weight vectors λ^2 and λ^3, the optimal solution is the interaction point of PF and f_1. Assume that a solution with respect to λ^2 during iteration is x_2, x_2 would not be replaced by x^* even if x^* is closer to λ^2 and is a better solution. It is because x_2 and x^* are equally optimal with the same fitness values on the PF line. The search process is in a standstill.

If the PF is a convex curve, as shown in Figure 4, assume that x_2 is the solution with respect to λ^2 during iteration; when the algorithm finds another solution x^*, x_2 will be replaced by x^* since the fitness value of x^* is better than x_2. Similarly, the solutions with respect to λ^1 and λ^3 will be replaced by the solutions that are located close to the ends of PF. At the late search process of the algorithm, most of solutions are aggregated at the ends of PF and the algorithm suffers in stagnation.

From the above analysis, we can see that the traditional weighted sum approach of MOEA/D suffers poor search ability. In the next subsection, we propose an improved MOEA/D based on reference distance to enhance the search ability of the algorithm.

4.2. An Improved Algorithm MOEA/D_RD Based on Reference Distance. To alleviate the aforementioned problems of MOEA/D, we propose an improved version based on reference distance, called MOEA/D_RD. Reference distance is the distance from each solution to the weight vector,

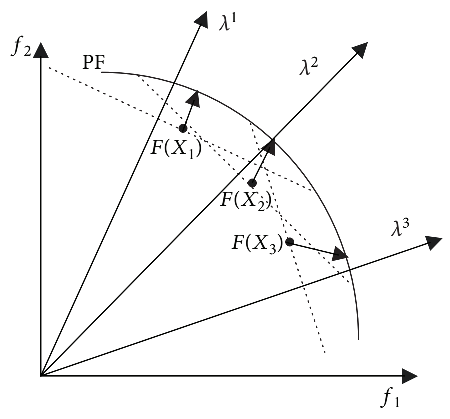

FIGURE 1: Solution generation process of MOEA/D.

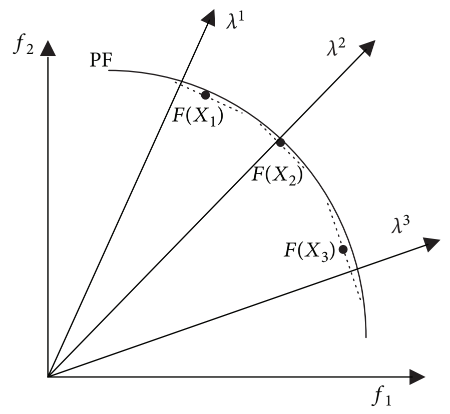

FIGURE 2: The late stage of MOEA/D.

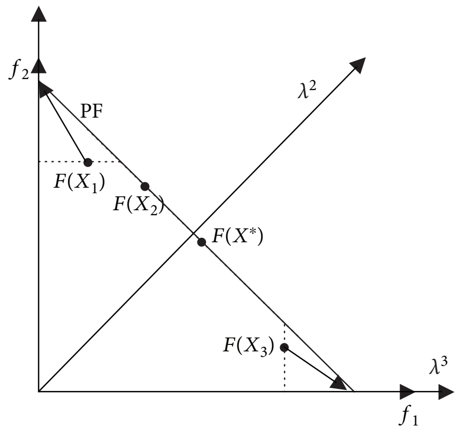

FIGURE 3: Pareto front as a line.

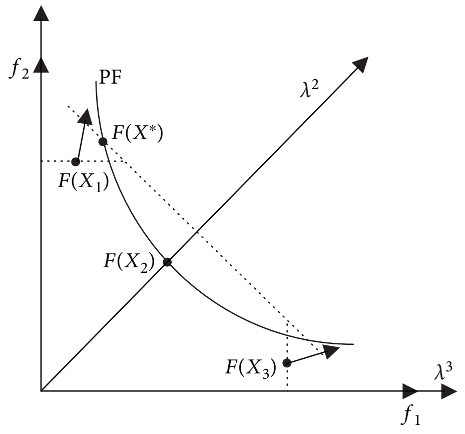

FIGURE 4: Pareto front as convex.

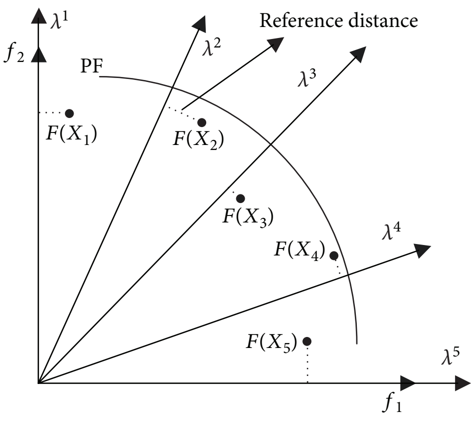

FIGURE 5: Reference distance.

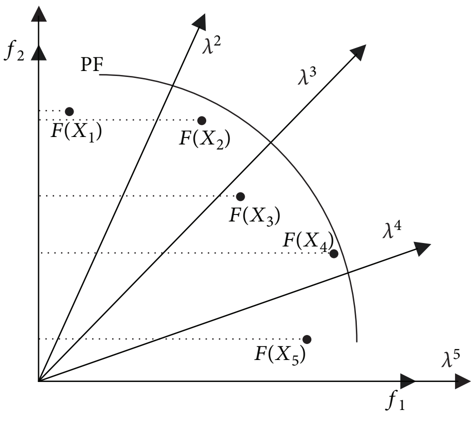

FIGURE 6: Reference distance to λ^1.

as shown in Figure 5. For each weight vector, we can calculate the distance of all solutions to it. For example, the distance of five solutions to weight vector λ^1 is depicted in Figure 6. We can see that x_1 is the solution with the shortest distance to λ^1 among the solutions. The calculation of reference distance is described in the following.

Given: the weight vector λ, the line from original point to λ L, the solution $F(x)$, the projection point from $F(x)$ to L is y; then, the distance d_1 from original point to y is

$$d_1 = \frac{\left\| F(x)^T \lambda \right\|}{\|\lambda\|}, \tag{7}$$

$$\text{Reference distance } d_2 = \|F(x) - d_1 \lambda\|. \tag{8}$$

MOEA/D_RD is described as follows.

```
Input
(i)Multiobjective optimization fitness function, that is, (6)
(ii)$R$ and $N$
(iii)A uniform spread of $N$ weight vectors: $\lambda^1, \lambda^2, \dots, \lambda^N$
Output: EP.
Step 1. Initialization:
    Step 1.1. Set EP = $\varnothing$, Count = 0;
    Step 1.2. Calculate the Euclidean distances between any two weight vectors and then work out the $T$ closest weight vectors to each weight vectors. 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 $\lambda^i$.
    Step 1.3. Randomly generate an initial population $\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^N \in \Omega$. Compute $FV^i = F(\mathbf{x}^i)$.
Step 2. Update:
For $i = 1, \dots, N$, do the following.
    Step 2.1. Randomly select two indexes, $k$ and $l$ from $B(i)$, and then generate a new solution $y$ from $\mathbf{x}^k$ and $\mathbf{x}^l$ by using general genetic operators.
    Step 2.2. Check if $y$ satisfies the constraints; if no, adjust $y$ to meet the constraints and mark $y$ to $y^*$.
    Step 2.3. Update the neighboring solutions. For each index $j \in B(i)$, if $g(y^* \mid \lambda) \geq g(\mathbf{x}^j \mid \lambda)$, then set $\mathbf{x}^j = y^*$ and $FV^j = F(y^*)$; Count = Count + 1.
    Step 2.4. Update the EP. Remove all the vectors dominated by $F(y^*)$ from EP. Add $F(y^*)$ to EP if there is no vector in EP that dominates $F(y^*)$.
    Step 2.5. If Count $\leq (N/R)$, go to Step 2.6; else, go to Step 3.
    Step 2.6. Find all the subproblems where solutions are not replaced and find the corresponding weight vectors to each subproblem.
    Step 2.7. Adjust the values of fitness functions for the solutions in EP and normalize them to [0,1].
    Step 2.8. For each weight vector found in Step 2.6, calculate the reference distances from the solutions in EP to the vector; find the solution with the shortest distance and use it to replace the current solution with respect to the corresponding subproblem.
Step 3. Stopping Criteria: If stopping criteria is satisfied, then stop and output EP. Otherwise, go to Step 2.
```

Algorithm 1

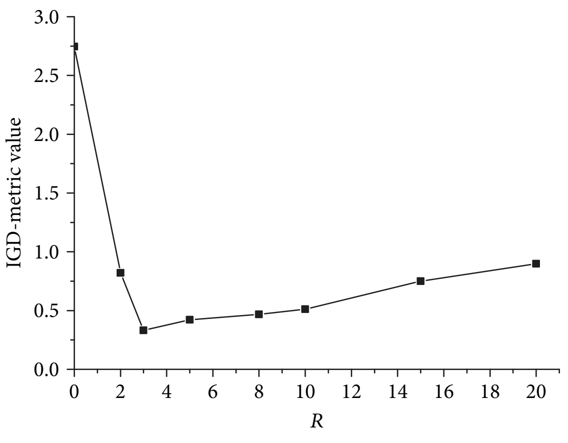

Figure 7: The effect of different R values.

At each generation t, MOEA/D_RD maintains

(i) N is the number of the subproblems considered in MOEA/D_RD,

(ii) a population of N points $\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^N \in \Omega$; where $\mathbf{x}^i$ is a vector and is the current solution to the ith subproblem; $\mathbf{x}^i$ corresponds to the weight vector λ^i,

(iii) $FV^1, \dots, FV^N$, where FV^i is the fitness value of $\mathbf{x}^i$, that is, $FV^i = F(\mathbf{x}^i)$ for each $i = 1, \dots, N$,

(iv) an external population (EP), which is used to store nondominated solutions found during the search,

(v) a variable R, $0 < R < N$; N/R stands for the replace rate; the value of R is empirically set. The variable Count is used to record the number of solutions being replaced at each generation.

The algorithm works as follows.

Take Figure 5 as an example. Assume $N = 5$ and $R = 2$; if the solutions corresponding to λ^1 and λ^4 are replaced, that is, Count = 2, since Count < N/R, the corresponding solutions of $\lambda^2, \lambda^3, \lambda^5$ need to be replaced. Because x_5 is dominated by x_4, x_5 is not in EP. x_1, x_2, x_3, and x_4 are in EP. x_2 is used to replace the solution in terms of λ^2 since x_2 has the shortest distance to λ^2. And x_3 is used to replace the solution in terms of λ^3 since x_3 has the shortest distance to λ^3. Although x_5 has the shortest distance to λ^5, but x_5 is not in EP, thus x_4 is used to replace the solution in terms of λ^5.

From the above example and algorithm description, we can see that MOEA/D_RD has the following features:

(1) The replacing strategy of MOEA/D_RD makes some unselected nondominate solutions in MOEA/D to generate the new population.

(2) Although MOEA/D uses uniform weight vector, the subproblems of multiple weight vectors may fall in the same area and it may bring about the low diversity of population. MOEA/D_RD brings the idea of reference distance, and it can help the individuals that stuck in local area to search more widely.

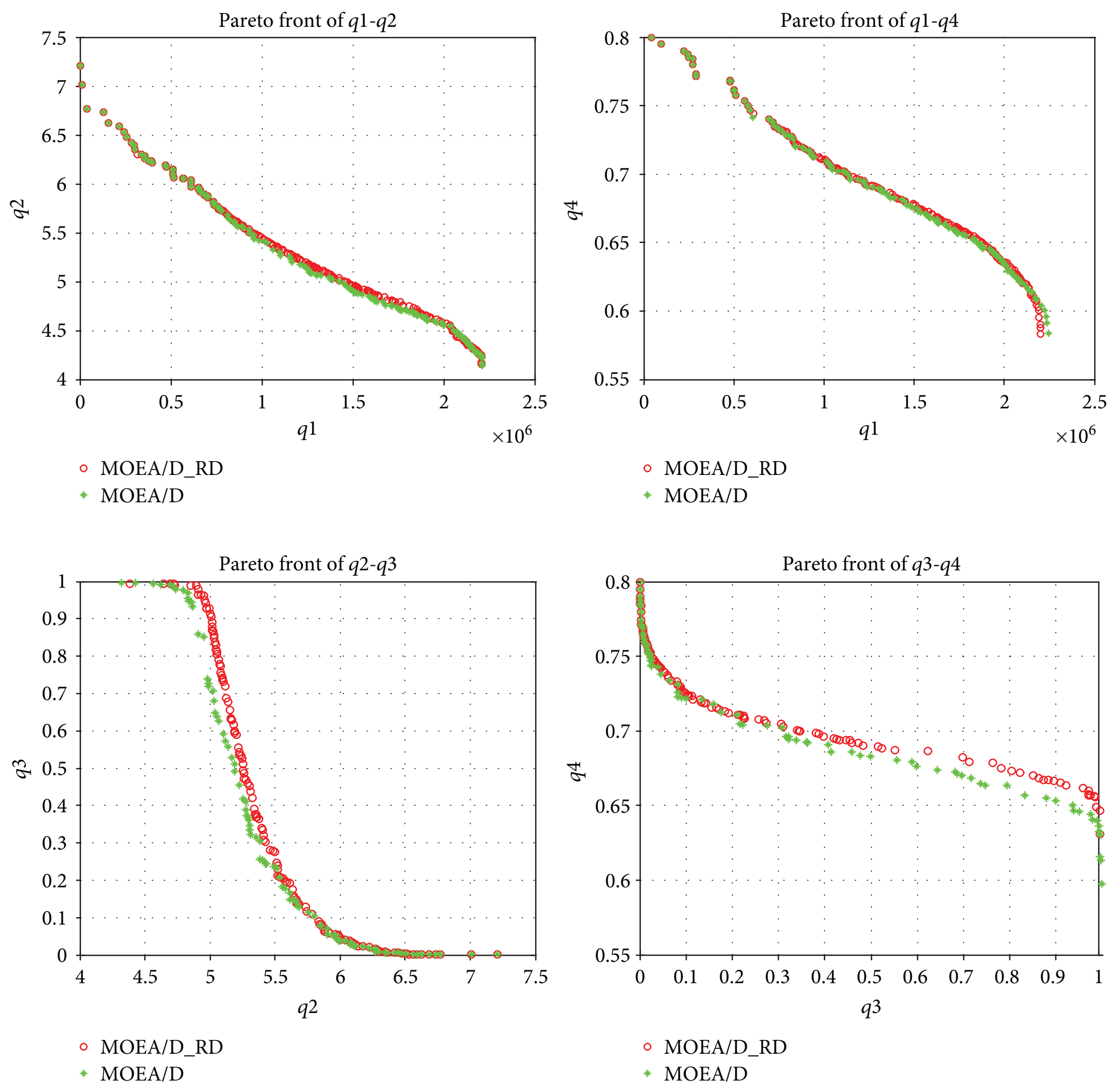

FIGURE 8: Comparison of PF between MOEA/D_RD (red circle) and MOEA/D (green *) for 2-objective optimization.

(3) MOEA/D_RD can bring new individuals when the algorithm is in stagnation; at the same time, the reference distance can guarantee that the new individuals are generated from the parents in neighborhood.

In brief, compared to the original MOEA/D, the replacing strategy based on reference distance in MOEA/D_RD increases the diversity of population and can obtain well-distributed solutions. The improved algorithm performs well in high-dimensional multiobjective optimization.

5. Experimental Evaluation

This section presents the experiments carried out to evaluate the performance of the proposed approach. First, the test data set based on the Constructive Cost Model (COCOMO II) is described. Three evaluating metrics are then introduced. Lastly, we compare MOEA/D_RD with MOEA/D, NSGA2, and NSGA3. All experiments were run on an Intel Core i5-2450M CPU@2.50 GHz, 4 GB memory PC with Win7 64-bit operating system.

5.1. COCOMO II Test Set. COCOMO II is a model to estimate the cost, effort, and schedule when planning a new software development activity. The test set is based on this model and consists of 50 software projects [5]. The number of lines of source code of these projects is between 1000 and 37000. The maximum duration of a project is 18 months, and the planning horizon is set to 3 years. The planning horizon is divided into 3 timeframes, one year (12 months) per timeframe. There are 1500 person-months in total for the planning horizon and 500 person-months per timeframe. Each project has an assigned risk value between 0.2 and 0.8. Potential revenue is set to the maximum of 150% and to the minimum of 85% of the initial costs. The total strategic alignment value is calculated by a weight sum of each strategy's alignment value which is set randomly. A maximum

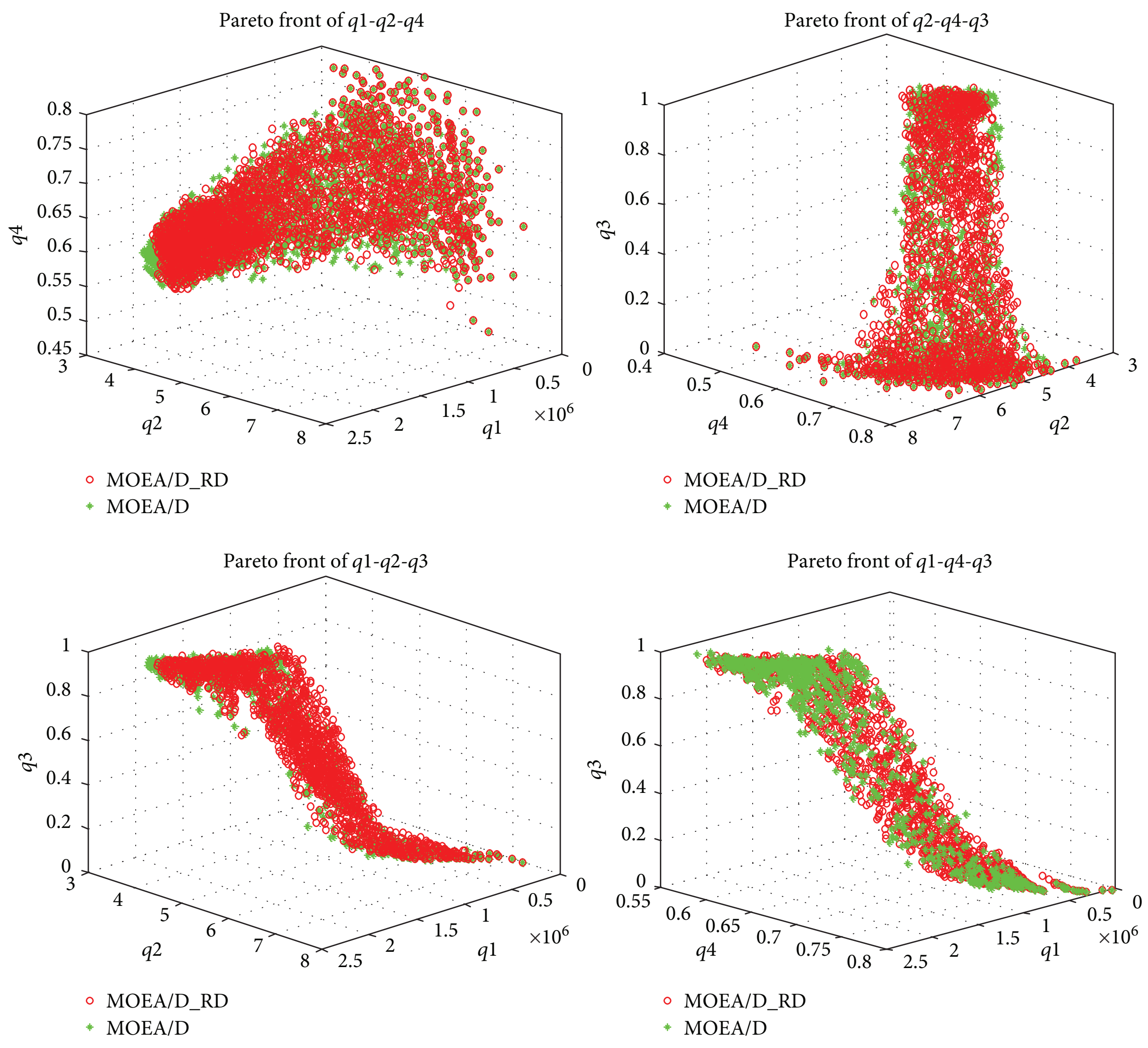

Figure 9: Comparison of PF between MOEA/D_RD (red circle) and MOEA/D (green *) for 3-objective optimization.

number of 30% of all projects are selected to have synergy effects with exactly one project where there is 15% of the positive synergy and 15% of the negative synergy. A number of 10% of all projects are selected randomly to be mandatory, and 4 projects are manually selected to be mutually exclusive.

5.2. Evaluation Metrics. In order to verify the proposed algorithm and compare to other state-of-the-art algorithms, we use three performance indexes as the following:

(i) Set coverage (C-metric) [14]: Let A and B be the two approximations to the PF of a multiobjective optimization problem, $C(A,B)$ is defined as the percentage of the solutions in B that are dominated by at least one solution in A, that is,

$$C(A,B)=\frac{|\{u\in B \mid \exists v\in A: v \text{ dominates } u\}|}{|B|}. \quad (9)$$

$C(A,B)=1$ 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.

(ii) IGD-metric [7]: Let A be a set of nondominated solutions obtained by the algorithm. Let P^* be the true PF. Since we do not know the actual PF for the software portfolio optimization problem in this paper, we use the optimal solutions obtained by all the compared algorithms as the approximation of P^*. The average distance from P^* to A is defined as

$$\mathrm{IGD}(A,P^*)=\frac{\sum_{v\in P*} d(v,A)}{|P^*|}, \quad (10)$$

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, $\mathrm{IGD}(A,P^*)$ could

Table 1: Comparison of C-metric of MOEA/D_RD and MOEA/D.

	C-metric (MOEA/D_RD, MOEA/D)	C-metric (MOEA/D, MOEA/D_RD)
q_1-q_2	0.559	**0.071**
q_1-q_4	0.527	**0.0628**
q_2-q_3	0.768	**0.021**
q_3-q_4	0.75	**0.018**
q_1-q_2-q_3	0.825	**0.005**
q_1-q_2-q_4	**0.265**	0.321
q_1-q_4-q_3	0.81	**0.01**
q_2-q_4-q_3	0.632	**0.063**
q_1-q_2-q_3-q_4	0.43	**0.176**

Table 2: Comparison of IGD-metric of MOEA/D_RD and MOEA/D.

	IGD-metric (MOEA/D)	IGD-metric (MOEA/D_RD)
q_1-q_2	3669.74	**313.09**
q_1-q_4	4273.46	**818.957**
q_2-q_3	0.037	**0.001**
q_3-q_4	0.007	**0.001**
q_1-q_2-q_3	1701.49	**46.68**
q_1-q_2-q_4	**212.82**	544.87
q_1-q_4-q_3	1732.64	**31.49**
q_2-q_4-q_3	0.021	**0.002**
q_1-q_2-q_3-q_4	100.86	**57.68**

Table 3: Comparison of GD-metric of MOEA/D_RD and MOEA/D.

	GD-metric (MOEA/D)		GD-metric (MOEA/D_RD)	
	Mean	Deviation	Mean	Deviation
q_1-q_2	4361.43	5017.01	**2751.47**	**3548.01**
q_1-q_4	4364.78	5496.39	**2435.11**	**3272.23**
q_2-q_3	0.062	0.063	**0.043**	**0.058**
q_3-q_4	0.013	**0.011**	**0.011**	**0.011**
q_1-q_2-q_3	1741.47	2734.43	**662.37**	**1422.7**
q_1-q_2-q_4	338.23	503.61	**310.92**	**459.31**
q_1-q_4-q_3	1591.24	3665.13	**559.19**	**1084.4**
q_2-q_4-q_3	0.022	0.02	**0.018**	**0.016**
q_1-q_2-q_3-q_4	**116.19**	215.604	142.091	**213.107**

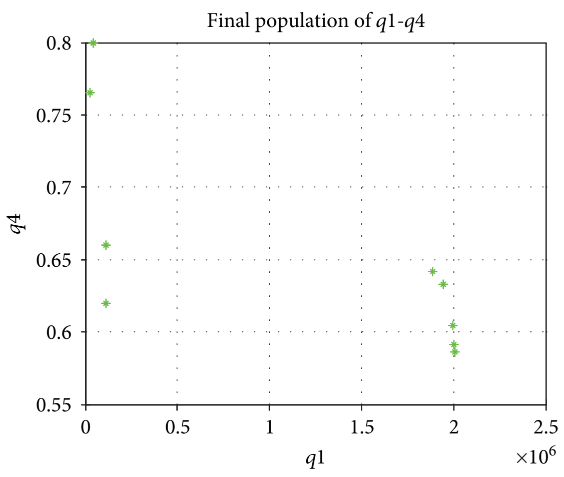

Figure 10: The final population of MOEA/D.

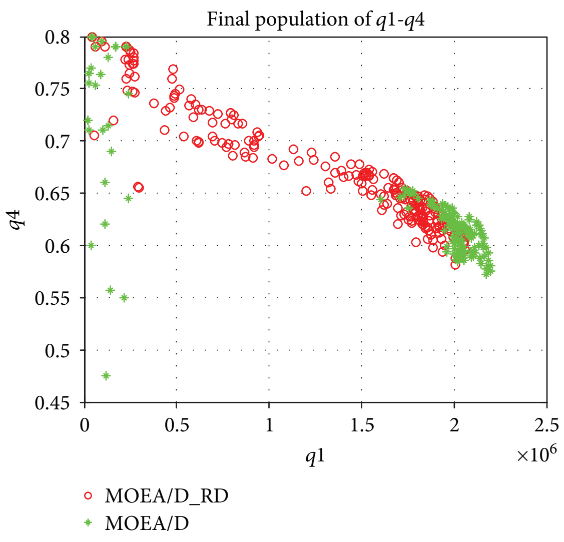

Figure 11: The final population of MOEA/D_RD and MOEA/D.

measure both the diversity and convergence of A in a sense. To have a low value of IGD(A, P^*), set A must be very close to the PF.

(iii) GD-metric [7]:

$$\text{GD}(A, P^*) = \frac{\sum_{v \in A} d(v, P^*)}{|A|}, \tag{11}$$

where P^* and A have the same definitions as in IGD-metric. $d(v, P^*)$ is the minimum Euclidean distance between v and the points in P^*. The smaller value GD-metric is, the more stable the algorithm.

To decide the value of R in MOEA/D_RD suitably, we first analyze the impact of R to the performance. Figure 7 shows the different IGD-metric values with different R values in the objectives of q_2 and q_3 optimization task. For every R value, the algorithm is run 20 times and the reference PF consists of all the nondominated solutions of the 20 runs.

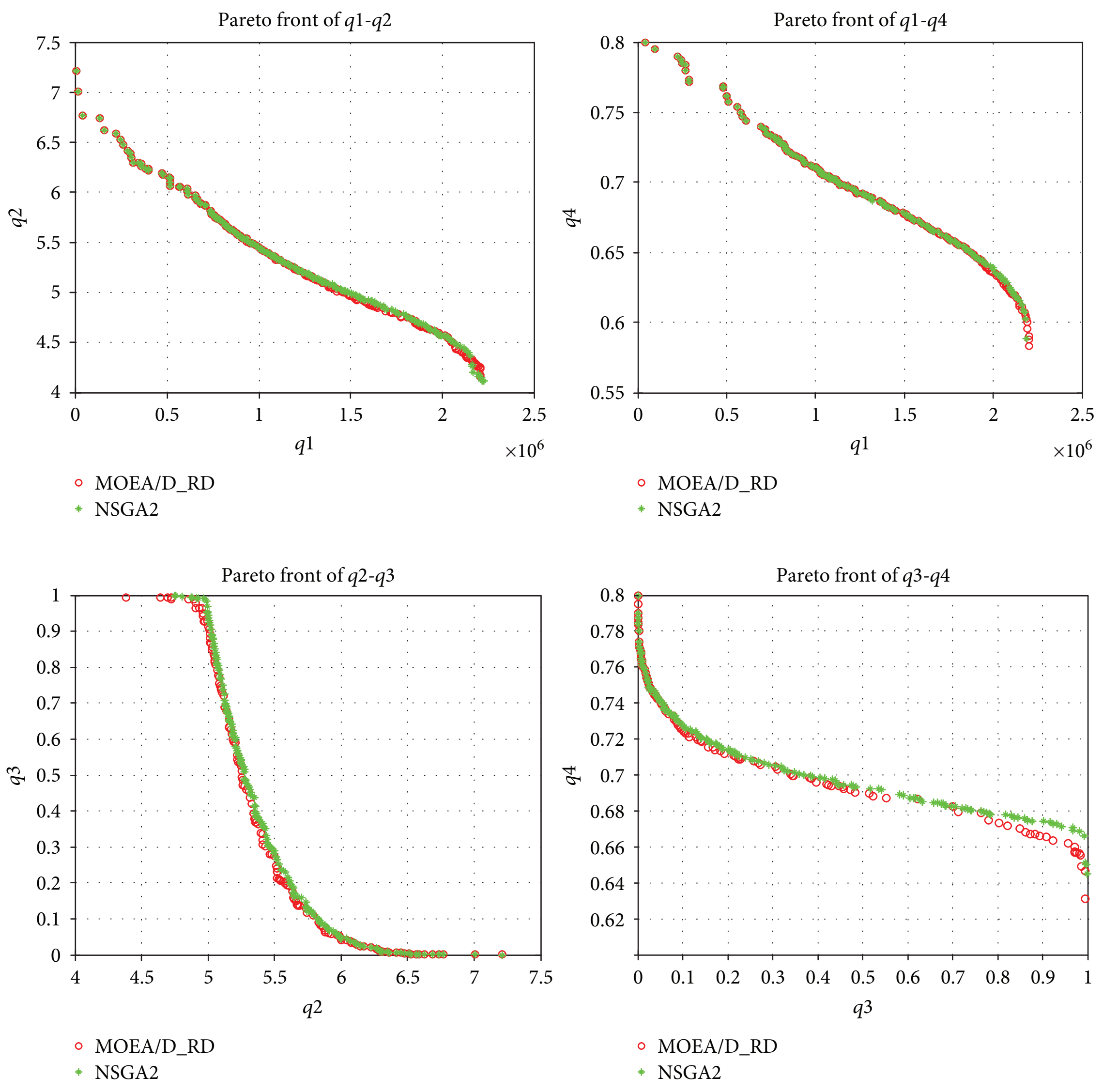

Figure 12: Comparison of PF between MOEA/D_RD (red circle) and NSGA2 (green *) for 2-objective optimization.

Calculate the IGD-metric for every single R value. When R equals to 0, it means reference distance is not used in the algorithm. We can see that the IGD-metric is the best when R is 3. The algorithm performs similarly when R is between 3 and 8. Considering that the convergence is slow if R is too small and the stagnation in search process is serious if R is too large, we set R to 5 in the following experiments.

5.3. Comparison between MOEA/D_RD and MOEA/D. As for the 4 objectives we mentioned in Section 3, q_1 and q_3 are positively correlated; that is, high revenue can be expected only when resources are effectively used throughout the whole planning horizon and vice versa. For q_2 and q_4, usually the project with either the lowest risk or the highest strategic alignment value is selected to the portfolio. Thus, q_1-q_3 and q_2-q_4 are not studied in our 2-objective optimization experiments. The experiments are conducted on 2-objective optimization problems: q_1-q_2, q_1-q_4, q_2-q_3, and q_3-q_4; 3-objective optimization problems: q_1-q_2-q_4, q_1-q_2-q_3, q_1-q_3-q_4, and q_2-q_3-q_4; and 4-objective optimization problem: q_1-q_2-q_3-q_4. There are 150 weight vectors in 2-objective optimization experiments, 351 weight vectors in 3-objective optimization experiments, and 455 weight vectors in 4-objective optimization experiment. The number of neighborhood is 10. The mutation rate is 0.01. The number of generation is 500 for 2-objective optimization and 1000 for 3- and 4-objective optimization problems. We run 20 independent runs with each of the compared algorithms where each run produced a set of nondominated solutions. The final population of nondominated solutions is plotted in Figures 8 and 9. We can see that MOEA/D-RD obtains more nondominated solutions.

Tables 1–3 give the comparisons between MOEA/D-RD and MOEA/D in terms of C-metric, IGD-metric, and GD-metric. The better performance is marked in bold. From Tables 1 and 2, we can see that MOEA/D outperforms

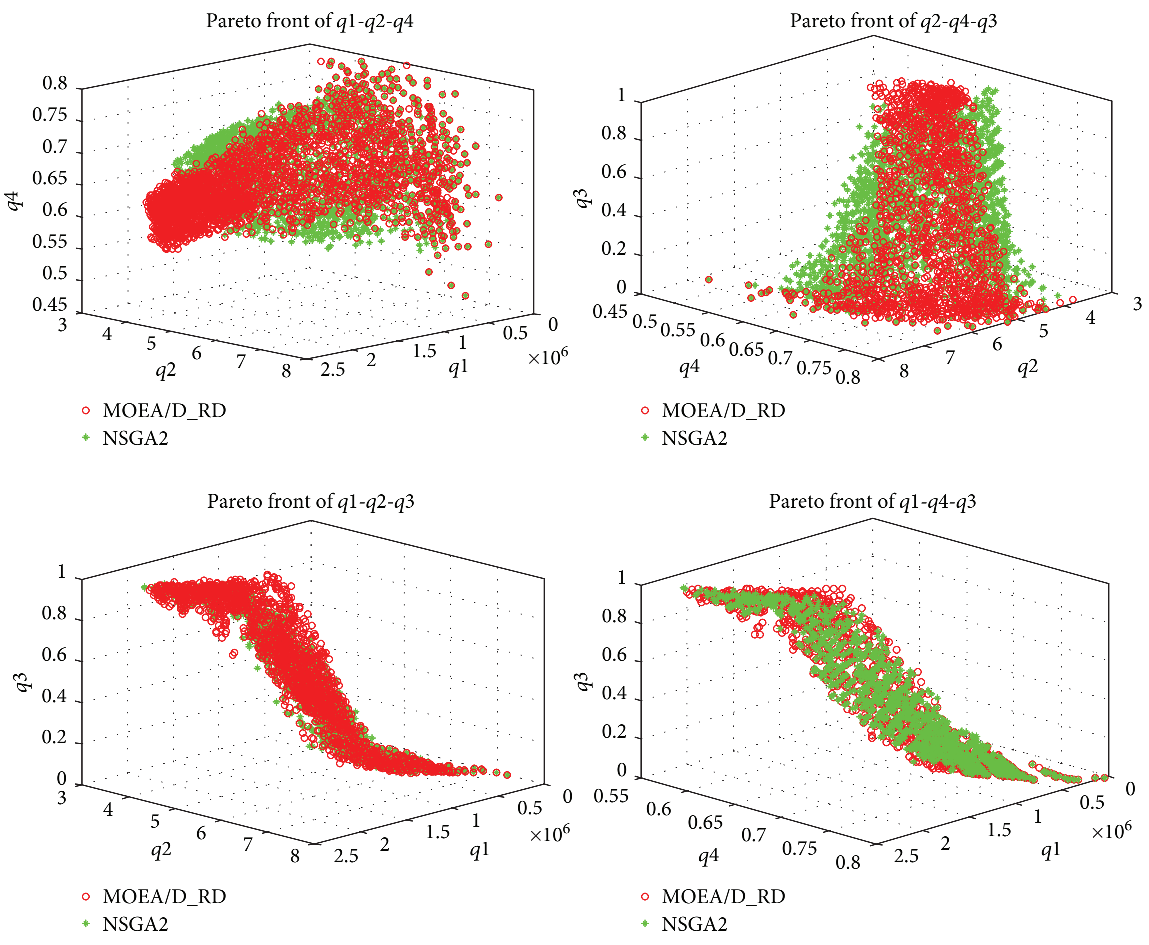

FIGURE 13: Comparison of PF between MOEA/D_RD (red circle) and NSGA2 (green *) for 3-objective optimization.

TABLE 4: Comparison of *C*-metric of MOEA/D_RD and NSGA2.

	C-metric (MOEA/D_D, NSGA2)	*C*-metric (NSGA2, MOEA/D_RD)
q_1-q_2	**0.048**	0.44
q_1-q_4	**0.036**	0.188
q_2-q_3	**0**	0.78
q_3-q_4	**0**	0.702
q_1-q_2-q_3	**0.027**	0.531
q_1-q_2-q_4	0.183	**0.15**
q_1-q_4-q_3	**0.024**	0.385
q_2-q_4-q_3	**0.052**	0.524
q_1-q_2-q_3-q_4	0.24	**0.14**

TABLE 5: Comparison of IGD-metric of MOEA/D_RD and NSGA2.

	IGD-metric (NSGA2)	IGD-metric (MOEA/D_RD)
q_1-q_2	**313.793**	2024.71
q_1-q_4	**427.488**	619.194
q_2-q_3	**0**	0.0127
q_3-q_4	**0**	0.006
q_1-q_2-q_3	**189.272**	324.263
q_1-q_2-q_4	40949.7	**207.512**
q_1-q_4-q_3	**138.696**	326.496
q_2-q_4-q_3	**0.008**	0.026
q_1-q_2-q_3-q_4	6633.18	**67.3109**

MOEA/D-RD in only one item of q_1-q_2-q_4. MOEA/D_RD performs better in the other 8 optimization problems with smaller *C*-metric and IGD-metric.

Figures 10 and 11 illustrate the improvement of the diversity of population and the distribution uniformity of solutions in MOEA/D_RD. Figure 10 presents the final population in one random run for q_1-q_4 problem. The number of population is 150, and there are only 9 different solutions at the last generation. We can see that the solutions in the neighborhood are almost the same and there is no new solution generated through genetic operators. The algorithm suffers in stagnation. Figure 11 gives the final population after 20 runs for q_1-q_4 problem. We can see that the nondominated solutions obtained by MOEA/D_RD are distributed more uniformly than the solutions by MOEA/D.

Table 6: Comparison of GD-metric of MOEA/D_RD and NSGA2.

	GD-metric (NSGA2)		GD-metric (MOEA/D_RD)	
	Mean	Deviation	Mean	Deviation
q_1-q_2	**957.144**	**2152.5**	2751.47	3548.01
q_1-q_4	**1223.3**	**2475.81**	2435.11	3272.23
q_2-q_3	**0.003**	**0.011**	0.043	0.058
q_3-q_4	**0.002**	**0.003**	0.011	0.011
q_1-q_2-q_3	**591.726**	2347.86	662.374	**1422.7**
q_1-q_2-q_4	**228.112**	**437.182**	310.917	459.31
q_1-q_4-q_3	**459.115**	**1074.91**	559.196	1084.4
q_2-q_4-q_3	**0.01**	**0.011**	0.018	0.016
q_1-q_2-q_3-q_4	184.506	324.817	**142.091**	**213.107**

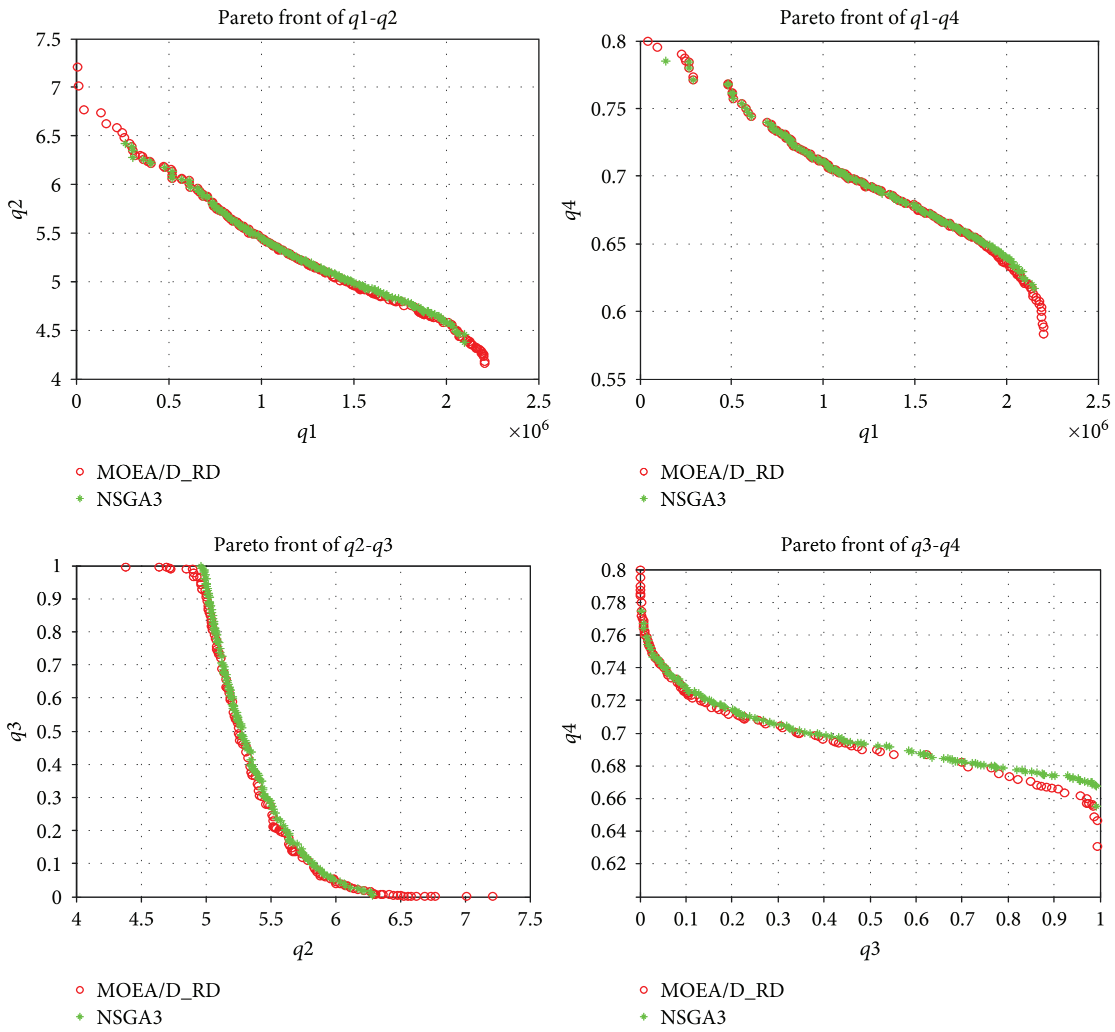

Figure 14: Comparison of PF between MOEA/D_RD (red circle) and NSGA3 (green *) for 2-objective optimization.

5.4. Comparison between MOEA/D_RD and NSGA2. NSGA2 has performed effectively in various optimization problems since it is invented. We also employ NSGA2 to solve our software project portfolio optimization model. The number of population of NSGA2 is 160, 360, and 500, respectively, for 2-objective, 3-objective, and 4-objective optimization problems. The mutation rate is 0.01. The number of generation is 500 for 2-objective optimization problem and 1000 for

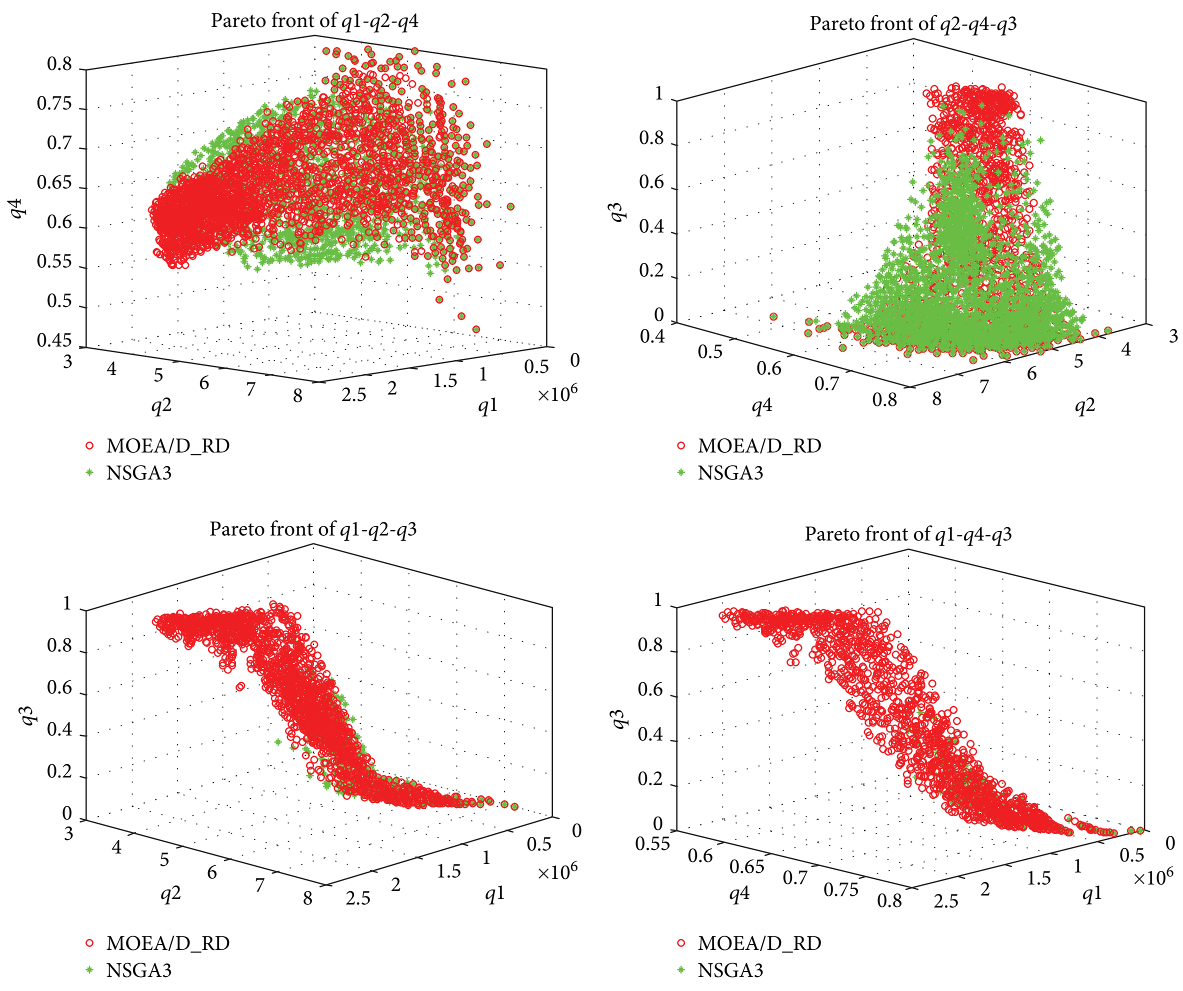

FIGURE 15: Comparison of PF between MOEA/D_RD (red circle) and NSGA3 (green *) for 3-objective optimization.

3- and 4-objective optimization problems. The final population of nondominated solutions is plotted in Figures 12 and 13 after 20 independent runs. We can see that NSGA2 performs better and obtains more nondominated solutions in 2- and 3-objective optimization problems. Tables 4–6 give the comparisons between MOEA/D-RD and NSGA2 in terms of C-metric, IGD-metric, and GD-metric. The better performance is marked in bold. MOEA/D_RD outperforms NSGA2 for 4-objective optimization problem.

5.5. Comparison between MOEA/D_RD and NSGA3. NSGA3 is the newest version of NSGA2 designed for many objective optimization problems. We use NSGA3 to solve the software project portfolio optimization problem especially the 3- and 4-objective optimization problems. The number of population for NSGA3 is set to 152, division parameter ρ 150, reference points 151 for 2-objective optimization experiments. The number of population is set to 352, division parameter ρ 25, reference points 351 for 3-objective optimization experiments. The number of population is 456, division parameter ρ 12, reference points 455 for 4-objective optimization experiments. The mutation rate is 0.01. The number of generation is 500 for biobjective optimization and 1000 for 3- and 4-objective optimization problems. The final population of nondominated solutions is plotted in Figures 14 and 15 after 20 independent runs. For 2-objective optimization problem, MOEA/D_RD can obtain more nondominated solutions and the distribution of solutions is more uniformly than NSGA3. For 3-objective optimization problem, the solutions obtained by NSGA3 are distributed in smaller area than MOEA/D_RD.

Tables 7–9 give the comparisons between MOEA/D_RD and NSGA3 in terms of C-metric, IGD-metric, and GD-metric. The better performance is marked in bold. From Tables 7 and 8, we can see that NSGA3 outperforms MOEA/D_RD in terms of C-metric while MOEA/D_RD outperforms NSGA3 in terms of IGD-metric. It means that NSGA3 can get better PF, but the distribution of solutions is worse than MOEA/D_RD. Table 9 indicates that NSGA3 can have better mean values but worse deviation values than MOEA/D_RD and we can have the same conclusion from Tables 7 and 8.

5.6. More Experiments. We also conducted some experiments to compare the four studied algorithms. The results using the PF obtained by the four algorithms and the correspondingly

Table 7: Comparison of C-metric of MOEA/D_RD and NSGA3.

	C-metric (MOEA/D_RD, NSGA3)	C-metric (NSGA3, MOEA/D_RD)
q_1-q_2	**0.02**	0.422
q_1-q_4	**0.005**	0.285
q_2-q_3	**0.024**	0.773
q_3-q_4	**0.007**	0.667
q_1-q_2-q_3	0.144	**0.065**
q_1-q_2-q_4	0.194	**0.142**
q_1-q_4-q_3	0.156	**0.055**
q_2-q_4-q_3	**0.009**	0.498
q_1-q_2-q_3-q_4	**0.083**	0.107

Table 8: Comparison of IGD-metric of MOEA/D_RD and NSGA3.

	IGD-metric (NSGA3)	IGD-metric (MOEA/D_RD)
q_1-q_2	8664.81	**2028.22**
q_1-q_4	2556.45	**1077.42**
q_2-q_3	0.027	**0.013**
q_3-q_4	**0.001**	0.006
q_1-q_2-q_3	156706	**43.468**
q_1-q_2-q_4	4337.33	**179.436**
q_1-q_4-q_3	203970	**54.698**
q_2-q_4-q_3	**0.013**	0.019
q_1-q_2-q_3-q_4	125345	**81.746**

Table 9: Comparison of GD-metric of MOEA/D_RD and NSGA3.

	GD-metric (NSGA3)		GD-metric (MOEA/D_RD)	
	Mean	Deviation	Mean	Deviation
q_1-q_2	**517.813**	**2271.78**	2751.47	3548.01
q_1-q_4	**746.94**	**1970.08**	2435.11	3272.23
q_2-q_3	**0.002**	**0.014**	0.043	0.058
q_3-q_4	**0.001**	**0.002**	0.011	0.011
q_1-q_2-q_3	**258.735**	1916.26	662.374	**1422.7**
q_1-q_2-q_4	**249.137**	594.39	310.917	**459.31**
q_1-q_4-q_3	**463.686**	4324.19	559.196	**1084.4**
q_2-q_4-q_3	**0.006**	**0.016**	0.018	**0.016**
q_1-q_2-q_3-q_4	**67.996**	393.873	142.091	**213.107**

computed IGD-metric are shown in Table 10. NSGA2 performs the best in 6 2- and 3-objective optimization problems with the smallest IGD-metric values. But for the 4-objective optimization problem, MOEA/D_RD outperforms the other algorithms.

Table 10: IGD-metric of the compared four algorithms.

	MOEA/D_RD	MOEA/D	NSGA2	NSGA3
q_1-q_2	2327.95	4616.15	**844.444**	9693.75
q_1-q_4	**1771.3**	4314.13	1924.44	3258.74
q_2-q_3	0.013	0.056	**0**	0.027
q_3-q_4	0.007	0.011	**0.001**	**0.001**
q_1-q_2-q_3	334.079	1908.11	**222.284**	120820
q_1-q_2-q_4	615.151	**341.423**	39321.5	5671.34
q_1-q_4-q_3	344	1893.54	**138.744**	148423
q_2-q_4-q_3	0.026	0.04	**0.012**	0.019
q_1-q_2-q_3-q_4	**136.089**	146.714	5616.01	113234

Table 11: Average running time of the four compared algorithms.

	MOEA/D_RD	MOEA/D	NSGA2	NSGA3
q_1-q_2	1.7	**1.56**	19.26	52.74
q_1-q_4	1.75	**1.68**	23.05	50.14
q_2-q_3	**0.97**	1.11	18.21	49.05
q_3-q_4	**1.54**	2.11	21.68	50.21
q_1-q_2-q_3	**14.62**	14.75	102.95	516.95
q_1-q_2-q_4	10.54	**10.45**	47.16	445.15
q_1-q_4-q_3	19.61	**19.01**	136.11	493.5
q_2-q_4-q_3	10.94	**10.76**	80.32	460.74
q_1-q_2-q_3-q_4	34.28	**33.25**	106.57	934.84

Table 11 presents the average running time of the four algorithms in 20 runs for the 9 optimization problems. We can see that MOEA/D consumes the least time and MOEA/D_RD costs the similar time compared with MOEA/D. NSGA2 needs several or more than ten times than MOEA/D_RD, and NSGA3 consumes the most time. To sum up, MOEA/D needs the least time but it is easy to suffer in stagnation and cannot obtain good nondominated solutions. NSGA3 is not likely suitable to solve the software project portfolio optimization problem. It needs the most time, and the distribution of solutions is not good. NSGA2 performs well in 2- and 3-objective optimization problems, but it needs much more running time than MOEA/D_RD. Generally speaking, MOEA/D_RD has the excellent overall performance. It can get the uniformly distributed nondominated solutions and performs the best for the 4-objective optimization problem. Although MOEA/D_RD is a little worse than NSGA2 in 2- and 3-objective optimization problems, but it consumes much less running time. In conclusion, we can say that MOEA/D_RD is an effective approach to the software project portfolio optimization problem.

5.7. Conclusion and Future Work. Based on Kremmel's model, a compact model with 4-objective optimization model for software project portfolio problem is proposed in this paper. The model is adaptive for software companies who can revise the objectives and constraints according to their own requirements. To solve the proposed model, an improved MOEA/D algorithm called MOEA/D_RD based on reference distance is proposed accordingly. The algorithm uses reference distance to select some solutions to generate new solutions. Compared with MOEA/D, NSGA2, and NSGA3, MOEA/D_RD performs well in terms of the quality of solutions and running time, especially for 4-objective optimization problem. Future work could cover several topics. A great number of projects for the test set could be conducted. It also would be important to test the approach on a test set with real-world data which may include some incomplete and uncertain data.

Conflicts of Interest

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

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (NSFC) Project nos. 61202296, 71102146, 61272067, 61370229, and 61370178, the National High-Technology Research and Development Program ("863" program) of China under Grant no. 2013AA01A212, the Natural Science Foundation of Guangdong Province Project no. S2012030006242, the Frontier and Key Technology Innovation Special Funds in Guangdong Province Project no. 2015B010109003, and the National Key Technology R&D Program of China Project no. 2013BAH72B01.

References

[1] S. Altuntas and T. Dereli, "A novel approach based on DEMATEL method and patent citation analysis for prioritizing a portfolio of investment projects," *Expert Systems with Applications*, vol. 42, no. 3, pp. 1003–1012, 2015.

[2] N. P. Archer and F. Ghasemzadeh, "An integrated framework for project portfolio selection," *International Journal of Project Management*, vol. 17, no. 4, pp. 207–216, 1999.

[3] J. Liu, X. Jin, T. Wang, and Y. Yuan, "Robust multi-period portfolio model based on prospect theory and ALMV-PSO algorithm," *Expert Systems with Applications*, vol. 42, no. 20, pp. 7252–7262, 2015.

[4] K. Doerner, W. J. Gutjahr, R. F. Hartl, C. Strauss, and C. Stummer, "Pareto ant colony optimization: a metaheuristic approach to multiobjective portfolio selection," *Annals of Operations Research*, vol. 131, no. 1–4, pp. 79–99, 2004.

[5] T. Kremmel, J. Kubalík, and S. Biffl, "Software project portfolio optimization with advanced multiobjective evolutionary algorithms," *Applied Soft Computing*, vol. 11, no. 1, pp. 1416–1426, 2011.

[6] C. H. Papadimitriou and M. Yannakakis, "On the approximability of trade-offs and optimal access of Web sources," in *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pp. 86–92, Redondo Beach, CA, USA, 2000, IEEE.

[7] P. M. Pardalos, I. Steponavičė, and A. Žilinskas, "Pareto set approximation by the method of adjustable weights and successive lexicographic goal programming," *Optimization Letters*, vol. 6, no. 4, pp. 665–678, 2012.

[8] I. Radziukynienė and A. Žilinskas, "Evolutionary methods for multi-objective portfolio optimization," in *Proceedings of the World Congress on Engineering 2008*, vol. 2, pp. 1155–1159, London, UK, 2008.

[9] A. Zhou, B. Y. Qu, H. Li, S. Z. Zhao, P. N. Suganthan, and Q. Zhang, "Multiobjective evolutionary algorithms: a survey of the state of the art," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 32–49, 2011.

[10] H.-G. Beyer, S. Finck, and T. Breuer, "Evolution on trees: on the design of an evolution strategy for scenario-based multiperiod portfolio optimization under transaction costs," *Swarm and Evolutionary Computation*, vol. 17, pp. 74–87, 2014.

[11] C. Lin and M. Gen, "Multi-criteria human resource allocation for solving multistage combinatorial optimization problems using multiobjective hybrid genetic algorithm," *Expert Systems with Applications*, vol. 34, no. 4, pp. 2480–2490, 2008.

[12] M. Rabbani, M. Aramoon Bajestani, and G. Baharian Khoshkhou, "A multi-objective particle swarm optimization for project selection problem," *Expert Systems with Applications*, vol. 37, no. 1, pp. 315–321, 2010.

[13] A. Ponsich, A. L. Jaimes, and C. A. C. Coello, "A survey on multiobjective evolutionary algorithms for the solution of the portfolio optimization problem and other finance and economics applications," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 3, pp. 321–344, 2013.

[14] Q. Zhang and H. Li, "MOEA/D: a multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

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

[16] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 577–601, 2014.

[17] H. Markowitz, "Portfolio selection," *The Journal of Finance*, vol. 7, no. 1, pp. 77–91, 1952.

[18] S. Arnone, A. Loraschi, and A. G. B. Tettamanzi, "A genetic approach to portfolio selection," *Neural Network World*, vol. 3, no. 6, pp. 597–604, 1993.

[19] S. C. Chiam, K. C. Tan, and A. Al Mamum, "Evolutionary multi-objective portfolio optimization in practical context," *International Journal of Automation and Computing*, vol. 5, no. 1, pp. 67–80, 2008.

[20] D. Lin, S. Wang, and H. Yan, "A multiobjective genetic algorithm for portfolio selection problem," in *Proceedings of ICOTA 2001*, pp. 567–574, Hong Kong, 2001.

[21] R. Subbu, P. Bonissone, N. Eklund, K. Bollapragada, and S. Chalermkraivuth, "Multiobjective financial portfolio design: a hybrid evolutionary approach," in *2005 IEEE Congress on Evolutionary Computation*, pp. 1722–1729, Edinburgh, UK, 2005, IEEE.

[22] J. Branke, B. Scheckenbach, M. Stein, K. Deb, and H. Schmeck, "Portfolio optimization with an envelope-based multi-objective evolutionary algorithm," *European Journal of Operational Research*, vol. 199, no. 3, pp. 684–693, 2009.

[23] N. Bradshaw, C. Walshaw, C. Ierotheou, and A. Parrot, “A multi-objective evolutionary algorithm for portfolio optimization,” in *Proceedings of the Symposium Evolutionary Systems*, pp. 27–32, Edinburgh, UK, 2009.

[24] E. Zitzler, M. Laumanns, and L. Thiele, “SPEA2: improving the strength Pareto evolutionary algorithm for multiobjective optimization,” in *Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001)*, pp. 95–100, Athens, Greece, 2002.

[25] I. Radziukyniene and A. Zilinskas, “Approximation of Pareto set in multi objective portfolio optimization,” in *Advances in Electrical Engineering and Computational Science*, vol. 39 of Lecture Notes in Electrical Engineering, pp. 551–562, Springer, Dordrecht, Netherlands, 2009.

[26] K. P. Anagnostopoulos and G. Mamanis, “A portfolio optimization model with three objectives and discrete variables,” *Computers & Operations Research*, vol. 37, no. 7, pp. 1285–1297, 2010.

[27] D. W. Corne, J. D. Knowles, and M. J. Oates, “The Pareto envelope-based selection algorithm for multiobjective optimization,” in *International Conference on Parallel Problem Solving from Nature (PPSN VIII)*, pp. 839–848, Paris, France, 2000.

[28] K. Khalili-Damghani, S. Sadi-Nezhad, F. H. Lotfi, and M. Tavana, “A hybrid fuzzy rule-based multi-criteria framework for sustainable project portfolio selection,” *Information Sciences*, vol. 220, pp. 442–462, 2013.

[29] E. Fernandez, C. Gomez, G. Rivera, and L. Cruz-Reyes, “Hybrid metaheuristic approach for handling many objectives and decisions on partial support in project portfolio optimisation,” *Information Sciences*, vol. 315, pp. 102–122, 2015.

[30] A. A. Tofighian and B. Naderi, “Modeling and solving the project selection and scheduling,” *Computers & Industrial Engineering*, vol. 83, pp. 30–38, 2015.

[31] G. Mavrotas, J. R. Figueira, and E. Siskos, “Robustness analysis methodology for multi-objective combinatorial optimization problems and application to project selection,” *Omega*, vol. 52, pp. 142–155, 2015.

[32] D. Rodríguez, M. Ruiz, J. C. Riquelme, and R. Harrison, “Multiobjective simulation optimisation in software project management,” in *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation - GECCO '11*, pp. 1883–1890, Dublin, Ireland, 2011.

[33] S. Gueorguiev, M. Harman, and G. Antoniol, “Software project planning for robustness and completion time in the presence of uncertainty using multi objective search based software engineering,” in *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation - GECCO '09*, pp. 1673–1680, Montreal, QC, Canada, 2009.

[34] B. Boehm, *Software Cost Estimation with COCOMO II*, Prentice Hall PTR, Upper Saddle River, NJ, USA, 2000.