Computing and Informatics, Vol. 31, 2012, 847–875

EVOLUTIONARY APPROACHES FOR MULTI-OBJECTIVE NEXT RELEASE PROBLEM

Xinye Cai, Ou Wei, Zhiqiu Huang

College of Computer Sciences and Technology
Nanjing University of Aeronautics and Astronautics
Nanjing, Jiangsu 210016, P.R. China
e-mail: {xinye, owei, zqhuang}@nuaa.edu.cn

Communicated by Vladimír Kvasnička

Abstract. In software industry, a common problem that the companies face is to decide what requirements should be implemented in the next release of the software. This paper aims to address the multi-objective next release problem using search based methods such as multi-objective evolutionary algorithms for empirical studies. In order to achieve the above goal, a requirement-dependency-based multi-objective next release model (MONRP/RD) is formulated firstly. The two objectives we are interested in are customers' satisfaction and requirement cost. A popular multi-objective evolutionary approach (MOEA), NSGA-II, is applied to provide the feasible solutions that balance between the two objectives aimed. The scalability of the formulated MONRP/RD and the influence of the requirement dependencies are investigated through simulations as well. This paper proposes an improved version of the multi-objective invasive weed optimization and compares it with various state-of-the-art multi-objective approaches on both synthetic and real-world data sets to find the most suitable algorithm for the problem.

Keywords: Multi-objective evolutionary optimization, multi-objective next release problem, requirement dependency, requirement engineering, software engineering

1 INTRODUCTION

In software industry, software companies usually develop and maintain large and complex software systems that have been sold to a range of diverse customers. One common problem that the companies face is to decide what enhancements/require-

ments should be implemented in the next release of the software. A company needs to find the ideal set of requirements while facing with

1. demands from their customers for a wide range of software requirements,
2. a situation where some requirements will require (one or more) prerequisite requirements,
3. customers who have different levels of importance to the companies, and
4. requirements that will take widely differing amounts of time and effort to meet [2].

The above problem is called next release problem (NRP) and is formulated in [2]. The NRP is usually considered as an example of a feature subset selection search problem [16] or an instance of a *knapsack problem* which is *NP*-hard [2]. In the above paper, a variety of techniques, including greedy algorithms and simulated annealing are applied to find requirements for the next release. During the past few years, van den Akker et al. used integer linear programming method to the NRP [1]. A comparison of analytical and evolutionary approaches for prioritizing software requirements was performed in [20]. Greer and Ruhe proposed an iterative genetic algorithm for NRP and applied it to the real-world problem [15]. Their proposed approach assesses and optimizes the degree to which the ordering conflicts with stakeholder priorities within technical precedence constraints; and also balances required and available resources.

In practical NRP, companies have to deal with multiple conflicting objectives, e.g. cost and customers' satisfaction. Single objective formulations have the drawback that the optimization of one objective might be achieved at the expense of other objectives, leading to the biased search of a certain part of the solution space. More recently, multi-objective formulations of the NRP have become increasingly popular. In the multi-objective next release problem (MONRP), each of the objectives is to be optimized as a separate goal, and they may be conflicting with each other. Compared to the single objective NRP, multi-objective algorithms applied to MONRP aim to explore the *Pareto* front (See Section 2) of non-dominated solutions, which is a set of "equally good" solutions instead of one solution in the single objective NRP. An example is given in Figure 1, where two objectives, cost and customers' satisfactions are to be optimized. In the example, four solutions for the MONRP exist, represented as a, b, c and d, respectively. Among these four solutions, a dominates b since solution a is better than solution b on both customers' satisfaction and cost of the next release requirements. On the other hand, a, c and d are non-dominated ("equally good") solutions, which forms a non-dominated front. Usually, a multi-objective approach needs to fulfil two equally important roles:

1. guiding the population of solutions towards Pareto front to achieve good convergence and
2. maintaining diversity in the population to have fully exploration of the search space.

These two roles can also be summarized in *convergence* and *diversity* as the performance measurements of a multi-objective approach.

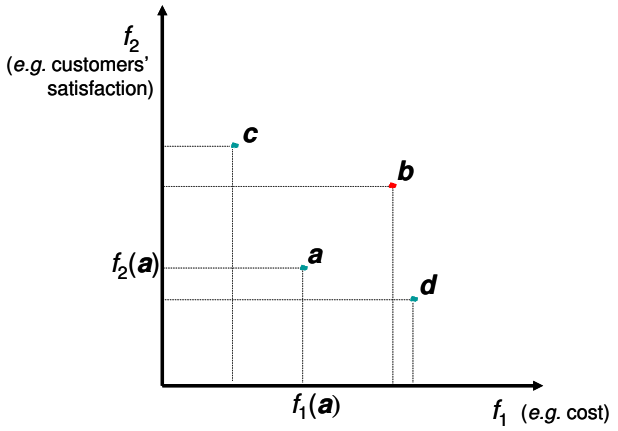


Fig. 1. An example of solutions for the MONRP, where a dominates b; and a, c and d are non-dominated solutions

In Zhang et al.'s formalization of MONRP [28], two objectives, cost of fulfilling requirements as well as the estimated satisfaction rating for the customers are considered. In almost the same time, Saliu and Ruhe formulated a two-objective MONRP that balances the tension between user-level and system-level requirements [25]. However, the above work with regard to MONRP did not consider requirement dependencies, which are prevalent and critical in the real NRP. For instance, some requirements may be coupled together. If one requirement is selected in one release, then others should be included as well in order to satisfy dependencies.

MONRP is a typical multi-objective optimization problem. There has been a variety of mathematical programming techniques developed to address multi-objective problems, such as linear programming and non-linear programming [12]. However, they all may have one or several of the following limitations [12, 23]:

1. prior knowledge of the true Pareto front is required;
2. they may not work when the Pareto front is concave or disconnected – they require differentiability of the objective functions and the constraints;
3. they can only obtain one solution from each run.

Evolutionary optimization approaches may be more suitable for MONRP as they operate on a population of solutions, which enables them to find several members of the Pareto optimal set in a single run. In addition, evolutionary algorithms are less susceptible to the shape or continuity of the Pareto front. For example, they can easily deal with discontinuous and concave Pareto front and do not require any prior information about the derivatives of the objectives.

Evolutionary algorithms are one class of search-based approaches that are inspired from Darwinian evolutionary theory. There are four main features in the evolutionary algorithms:

1. an individual, e.g. decision vector, represents a solution to the given problem;
2. according to the objective functions, each individual is evaluated and assigned a fitness to reflect the quality of the solution;
3. a selection process is performed on the population and
4. the population is updated and new solutions are generated in each generation.

More details of advantages of evolutionary approaches and other search based techniques can be read in [27].

The main contributions of the work reported in this paper are as follows:

- This paper proposes a requirement-dependency based multi-objective next release problem (MONRP/RD). Total cost of fulfilled requirements and customers' satisfaction are the two objectives that have been considered.
- The solution space of the formulated MONRP/RD is presented as non-dominated solution which allows the decision maker to see how the solutions can balance between the predefined two objectives. A popular multi-objective evolutionary approach NSGA-II is applied to the formulated MONRP/RD. Both the synthetic random and the real industrial data have been used to verify the effectiveness of the algorithms in the paper.
- Experiments analyze how algorithm performance scales up with problem size. Experiments also analyze the influence of the degree of the requirement dependencies on the performance of the multi-objective evolutionary approach.
- This paper proposed an improved version of multi-objective invasive weed optimization (IWO/MO2) and compares it with various multi-objective approaches on the formulated MONRP/RD. Two performance metrics, convergence metric ρ and diversity metric Δ are used to compare the performance of the different multi-objective algorithms quantitatively.

The rest of the paper is organized as follows. Section 2 introduces the basic concepts of multi-objective optimization. Section 3 defines the formulation of MONRP/RD formally. Section 4 briefs some popular multi-objective evolutionary algorithms. An improved version of IWO/MO is also proposed in this section. Section 5 explains the data set and describes the experimental setup. Section 6 presents the results of experiments on random and the real industrial data using NSGA-II algorithm. The scalability and the influence of requirement dependencies for the MONRP/RD are also considered in this section. Section 7 presents a comparison of different multi-objective evolutionary approaches. Non-dominated solutions produced by each algorithm can be visualized in terms of figures. Besides, the definition of the two metrics to compare the performances of the used approaches quantitatively is also defined in this section. Section 8 details the threats to validation and

Section 9 introduces the MONRP related work. Section 10 draws the conclusions and future work.

2 MULTI-OBJECTIVE OPTIMIZATION PROBLEM

2.1 The Definition of Multi-Objective Optimization Problem

In the multi-objective optimization problem (MOP), two or more usually conflicting objectives are required to be optimized simultaneously. A MOP can be defined as:

Definition 1 (Multi-objective optimization problem)**.** Given a problem involving N decision variables $x_1, x_2, \ldots, x_N$ in a search space $\boldsymbol{X} \subset \Re^N$, we assume, without loss of generality, M objectives $f_1(\cdot), \ldots, f_M(\cdot)$ in *objective function space* $\boldsymbol{Y} \subset \Re^M$, are to be minimized.

Minimize $\mathbf{f}(\vec{x}) = ((f_1(x_1, x_2, \ldots, x_N)), \ldots, f_M(x_1, x_2, \ldots, x_N))$. The vector function is a mapping $\boldsymbol{f}: \boldsymbol{X} \rightarrow \boldsymbol{Y}$.

For example, Figure 1 shows a two-objective NRP, where objective f_1 is total cost and objective f_2 is customer's satisfaction.

2.2 Pareto Optimal Front

In MOP, it is usually not possible to find a single solution which is optimal for all the objectives. Instead, many good solutions may exist. These solutions are always "trade-offs" or good compromises among the objectives. Since the conventional concept of optimality does not hold, a concept of Pareto optimality is adopted. The Pareto optimality concept, which was first proposed by Edgeworth and Pareto [26], is formally defined as follows [24, 9].

Definition 2 (Dominance and non-dominated solutions)**.** Let $\vec{x}$,$\vec{y}$ be two vectors of decision variables in MOP. $\vec{x}$ is considered to dominate $\vec{y}$ (written as $\vec{x} \prec \vec{y}$) iff they satisfy the conditions:

$$\begin{aligned} &\vec{x}, \vec{y} \in \mathbf{X}, \forall i \in 1, \ldots, M | f_i(\vec{x}) \leq f_i(\vec{y}) \\ &\vec{x}, \vec{y} \in \mathbf{X}, \exists j \in 1, \ldots, M | f_j(\vec{x}) < f_j(\vec{y}). \end{aligned} \tag{1}$$

On the contrary, a decision vector $\vec{x}$ is considered to be a non-dominated solution iff there is no other solution that satisfies Equation (1). The set of all non-dominated solutions forms a *Pareto set*.

Definition 3 (Pareto front)**.** The projection of the Pareto set $\boldsymbol{P}$ in the M dimensional objective function space $\boldsymbol{Y}$ is called *Pareto front*, $\boldsymbol{F}$.

$$\boldsymbol{F} = \{(f_1(\vec{x}), f_2(\vec{x}), \ldots, f_M(\vec{x})) | \vec{x} \in \boldsymbol{P}\} \tag{2}$$

In Figure 1, for instance, non-dominated solution a, c and d form a Pareto front.

3 PROBLEM FORMULATION

In this section, we formulate the requirement dependencies based multi-objective next release problem (MONPR/RD). We assume that an existing software system is associated with several *customers* whose requirements need to be considered in the next release of the software system. The set of the customers is denoted by

$$U = \{u_1, u_2, \ldots, u_m\}.$$

The set of all the *requirements* that need to be considered is denoted by

$$R = \{r_1, r_2, \ldots, r_n\}.$$

Implementation of each requirement needs to be allocated with certain number of resources, i.e., the cost of development. This is represented by a *cost* vector

$$C = \langle c_1, c_2, \ldots, c_i, \ldots, c_n \rangle$$

where c_i indicates that cost used for implementation of the requirement $r_i \in R$.

Moreover, it is assumed that not all requirements are equally important for a given customer. We use *score* to represent the importance of a requirement to a customer. Each customer u_j assigns a *score* to requirement r_i denoted by: $s(r_i, u_j)$ where $s(r_i, u_j) > 0$ if customer j desires implementation of the requirement i and 0 otherwise.

$$S = \begin{Bmatrix} s(1,1) & s(1,2) & \ldots & s(1,i) & \ldots & s(1,n) \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ s(j,1) & s(j,2) & \ldots & s(j,i) & \ldots & s(j,n) \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ s(m,1) & s(m,2) & \ldots & s(m,i) & \ldots & s(m,n) \end{Bmatrix}$$

Each customer u_j has a subset of requirements that they expect to be satisfied denoted by R_j such that $R_j \subseteq \Re$, $\forall r \in R_j$, $s(r, u_j) > 0$.

Requirements dependencies can be discovered manually by requirement managers or automatically with the aid of natural language processing tools [5, 6].

We use a *dependency graph* to model the dependencies over the requirements of a software systems.

Definition 4 (Dependency Graph)**.** A *dependency graph* is a graph $G = \langle R, E \rangle$, where R is a set of requirements, and $E \subseteq R \times R$ is a set of edges between the requirements in R such that for any pair of requirements $r_i, r_i \in R$, $(r_i, r_j) \in E$ only if there exists dependency between r_i and r_j.

Note that in the definition above, edge E in dependency graph represents some kind of dependent relations between requirement r_i and r_j. In our work, we consider 3 most common functional dependencies defined by Carlshamre et al. [5] as follows:

AND defines a relation δ on the dependency graph G such that $E(i,j) \in \delta$ means that requirement r_i is selected if and only if requirement r_j has to be chosen.

Precedence define a partial order ϵ on the dependency graph G such that $E(i,j) \in \epsilon$ means that requirement r_i has to be implemented before requirement r_j.

OR define a relation ϕ on the dependency graph G such that $E(i,j) \in \phi$ means that at most one of r_i,r_j can be selected.

We consider two objectives in the next relation problem, *customers satisfaction* and *required cost*. Requirement dependencies are considered as constraints of the optimization problem. The decision vector $\vec{x} = x_1, \ldots, x_n \in 0, 1$ determines the requirements that are to be satisfied in the next release. In this vector, x_i is 1 if requirement r_i is selected and 0 otherwise. The decision vector $\vec{x}$ denotes the solution to the MONRP/RD.

The level of satisfaction for a given customer depends on the requirements that are satisfied in the next release of the software. For m customers, the level of their total expected satisfaction is defined by

$$Score = \sum_{j=1}^{m} \sum_{i=1}^{n} s_{ij} \cdot x_i. \tag{3}$$

The required cost for implementing the requirements of S is

$$Cost = \sum_{i=1}^{n} cost_i \cdot x_i \tag{4}$$

subject to

- $x_i = x_j$ for all pairs $r(i,j) \in \delta$ (AND constraints)
- $x_i \neq x_j \vee x_i = x_j = 0$ for all pairs $r(i,j) \in \phi$ (OR constraints)
- $x_i = 1 \wedge x_j = 1 \vee x_i = 1 \wedge x_j = 0 \vee x_i = x_j = 0$ for all pairs $r(i,j) \in \epsilon$ (Precedence constraints).

The goal of the two objectives in the next relation problem is to determine the decision vector such that the customers satisfaction is maximized and the required cost is minimized. The two objectives can be unified into minimization problems by multiplying the customers satisfaction by -1.

In this paper, to address the MONRP/RD problem, we explore the Pareto front of multi-objective search algorithms, which provide valuable information for understanding of the trade-offs inherence in meeting the conflicting objectives while preserving satisfaction of the constraints imposed by requirements dependencies. In the following section we introduce multi-objective optimization problem and the search algorithms.

4 MULTI-OBJECTIVE EVOLUTIONARY ALGORITHMS

Evolutionary algorithms (EAs) belong to a class of search-based computational methods inspired by biological evolution which evolve solutions tailored to a particular task. EAs maintain a population of *individuals*, each of which represents one solution to the task. In our formulation of MONRP/RD, an individual is a decision vector, representing a subset of customers whose chosen requirements are to be satisfied. Each individual's suitability is evaluated using a task-specific *fitness function* (objective function). Once each individual (solution)'s fitness is evaluated, a *selection phase* is activated and the individuals with higher fitness value are more likely to be selected for continued evolution. Two operators, analogous to biological *crossover* and *mutation* are unutilized to produce variations of the high-fitness solutions. Mutation makes random changes to each bit of a decision vector (solution) based on *mutation rate* and crossover operator exchanges parts between two solutions based on *crossover rate*. The parent solutions are replaced by the produced offspring solutions and these offspring solutions will become the parent solutions in the next generation. The process repeats until EAs exceed a preset number of generations.

Over recent years, EAs have received increasing attention, particular in solving the multi-objective optimization problem as EAs operate on a population of solutions, which is able to find several solutions of the Pareto optimal set in a single run. Population-based nature enables the multi-objective evolutionary algorithms (MOEAs) to capture the dominance relations among solutions in the population and guide the search towards desirable Pareto front. In addition, multi-objective evolutionary algorithms are less susceptible to the shape or continuity of the Pareto front (e.g. they can easily deal with discontinuous and concave Pareto fronts) [7].

MOEAs is an extension of EAs in multiple and possibly conflicting objectives. We consider two objectives, total cost and customers' satisfaction, which are defined in Equations (3) and (4), respectively. MOEAs need to fulfil two roles:

1. guiding the population towards Pareto optimal to achieve good convergence and
2. maintaining diversity in the population to have full exploration of the search space.

Thus MOEAs aims to select diverse high-fitness solutions (usually non-dominated solutions) instead of one highest-fitness solution in EAs.

During the optimization process, sometimes good solutions are lost due to random effects. Thus methods of saving elite solutions, termed *elitism*, is another important issue in MOEAs. One possible method is to use a deterministic selection operator on the combined population of parent and offspring, instead of replacing the parent population with the offspring. Another alternative is the use of *archiving* – a secondary population which maintains promising solutions. Most MOEAs use the combination of both dominance and diversity information to select diverse non-dominated solutions to store into the archive. For a more detailed discussion of MOEAs, readers are referred to Coello et al. [4] and Deb [9].

Next we introduce three MOEAs, NSGA-II, SPEA2 and IWO/MO, which have been applied to address our formulated MONRP/RD. In addition, an improved version of IWO/MO, which is named IWO/MO2, is also proposed in this paper.

4.1 The NSGA-II Algorithm

The non-dominated sorting genetic algorithm-II (NSGA-II) [10], proposed by Deb et al., is an improved version of NSGA [8]. The algorithm maintains both a population and an archive with the size of N solutions. Elite preservation is applied in each generation after the population is merged with the archive. The N best-ranked solutions are preserved in the archive. NSGA-II proposes fast non-dominated sorting with complexity of $O(MN^2)$ (where M is the number of objectives) to assign ranks to the solution in the merged population. In this way, the population is sorted into several fronts. Each solution is assigned a fitness value according to its non-dominated level. In addition, when the number of non-dominated solutions exceeds the size of the archive, a crowded-comparison method for diversity is invoked to further discriminate among non-dominated solutions. The crowded-comparison method calculates the crowding distance for each solution belonging to the same rank (front). Subsequently, solutions with higher crowding distances are assigned higher fitness.

In NSGA-II, each solution i in the population has two attributes: non-dominated rank (i_{rank}) and crowding distance ($i_{distance}$). A partial order $\prec$ is defined as follows: $i \prec j$ if ($i_{rank} < j_{rank}$) or (($i_{rank} = j_{rank}$) and ($i_{distance} > j_{distance}$)). Between two solutions with different non-domination ranks, the solution with the lower (better) rank is preferred. Otherwise, if both solutions belong to the same front, then the solution that is located in a less crowded region is preferred [10].

The main loop of the NSGA-II is described in Algorithm 1. Initially, a random parent population P_0 with size of N is created. Tournament selection, crossover, and mutation operators are applied to create a child population Q_0 of size N.

In Section 5, the NSGA-II approach is applied to the formulated MONRP/RD to explore Pareto front in different scenarios.

4.2 The SPEA2 Algorithm

The Strength Pareto Evolution Algorithm (SPEA2) [31] is another popular MOEA with powerful elitism preserved mechanism. The archive is created for storing non-dominated solutions. It is then combined with the current population to create offspring for the next generation. The size of the archive is fixed and usually set to be equal to the population size. If the number of non-dominated solution is smaller than the archive size, then the other dominated solutions with better fitness are selected to fill the archive. Otherwise, if the number of non-dominated solutions is larger than the archive size, a truncation operator is applied based on the density estimation of each solution. The density metric is called k^{th} *nearest distance*, that is, the average distance from each solution to k nearest solutions. The non-dominated

Algorithm 1: NSGA-II (main loop) Deb (2001)

```
1  while t ≤ max_generation do
2    Let R_t = P_t ∪ Q_t
3    Let F = fast − non − dominated − sorting(R_t)
4    Let P_t + 1 = ∅ and i = 1
5    while |P_{t+1}| + |F_i| ≤ N do
6      Apply crowding-distance-assignment (F_i)
7      Let P_{t+1} = P_{t+1} ∪ F_i
8      Let i = i + 1
9    end
10   Sort(F_i, ≺)
11   Let P_{t+1} = P_{t+1} ∪ F_i[1 : (N − |P_{t+1}|)]
12   Let Q_{t+1} = make − new − pop(P_{t+1})
13   Let t = t + 1
14 end
```

solutions with large density metric value will be truncated from the archive. For more information about SPEA2 algorithm, readers are referred to [31].

4.3 The IWO/MO Algorithm and a Proposed Variant of IWO/MO

The Invasive Weed Optimization (IWO) algorithm [22] developed by Mehrabian and Lucas in 2006 mimics the principles and behaviors of weedy invasion and colonization in the shifting and turbulent environment to solve optimization problems. More recently, Kundu et al. proposed a multi-objective version of IWO (named IWO/MO) [21]. The procedures of IWO/MO can be described as follows [21]. In the beginning, a population of solutions with size of np is generated, evaluated and ranked. After that, each solution in the population is allowed to reproduce a number of seeds (child solutions) distributed over the search space around itself. This step is sometimes called intra-operator as it applies reproduction and mutation of a solution itself. The solution with better fitness value is allowed to produce more seeds according to the following formula:

$$Seed_i = floor(S_{\min} + (S_{\max} - S_{\min}) * ((np - rank_i)/np)) \tag{5}$$

where $rank_i$ is the rank of the i^{th} solution and $seed_i$ is the number of seeds produced by it. Subsequently, when the weed population exceeds a upper limit (pmax), the population of solutions is sorted and the best pmax solutions are allowed to survive for the next iteration. The whole process is continued until certain stopping criterion is met.

Although IWO/MO algorithm has shown its great efficiency on benchmark functions of the multi-objective problems, it lacks inter-information (merits) exchange among weed communities. This may lead the search process stucking in the local optima. In this paper, we propose an improved version of IWO/MO, which is

named IWO/MO2 to distinguish it from the original IWO/MO. In our proposed IWO/MO, both intra and Inter operator are applied to exploit the merits of members both in intra-communities and inter-communities. The intra operator has been already introduced in the original IWO/MO. On the other hand, the inter operator aims to exchange information among inter-communities of weeds. As each weed may carry some traits that are adapted for the growing environment and can influence its nearby individuals, the inter operator is applied by randomly selecting two individuals from inter-communities to exchange merits and produce new individuals. Then the new reproduced individual will be completed with the original population together and decided whether it will enter the next generation of the algorithm. In addition, we use the fast non-dominated sorting and crowding-distance-assignment [10] to maintain the convergence and diversity of the algorithm. The proposed IWO/MO2 is described in Algorithm 2 as follows.

Algorithm 2: IWO/MO2

```
1  Let F = initialize(p_init)
2  Let Q = evaluate(F)
3  Let rank = fast − non − dominated − sorting(Q)
4  while t ≤ max_generation do
5    Let children = intra − operator(F, rank)
6    Let P_t = F ∪ children
7    Let S_t = inter − operator(P_t)
8    Let P_t = S_t ∪ P_t
9    Let R_t = Evaluate(P_t)
10   if |P_t| ≥ p_max
11     rank = fast − non − dominated − sorting(R_t)
12     diversity = crowding − distance − assignment(R_t)
13     Let F = Select(R_t, rank, diversity)
14   end
15   else
16     Let F = P_t
17     Rank = Sort(R_t)
18   end
```

4.4 The Random Search

The random search approach has also been applied to the MONRP/RD. The search approach does not utilize the memory information that has been garnered in the search process. It is implemented simply by laying down a number of solutions randomly in the search space. The random search approach is merely a "sanity check". All other algorithms should be able to easily outperform random search for a well-formulated optimization problem.

5 EXPERIMENTAL SETUP

5.1 Data Sets

This section describes the test data sets used to fulfil the research tasks of MONRP-/RD. There are two types of data: the random data and the real data. The random data set is generated randomly according to the problem model. The range of costs is from 1 through 9 inclusive. We do not allow zero cost. The second data set is taken from Motorola [3]. The Motorola data set has 35 requirements and 4 customers. Table 1 shows the cost of each of the 35 requirements, which ranges from 10 to 1 100. In the Motorola data set, every requirement is demanded by only one customer exclusively. In order to increase the complexity of the MONRP/RD, we expand the number of customers to 10. Since the Motorola data does not contain information about the requirement dependencies, we randomly generate the requirement dependencies to make use of this data set. In order to investigate the influence of the requirement dependencies on the algorithm performance, we divide the degree of requirement dependencies into no, weak, median and strong levels. As noted by Harman in [16], customers prefer to divide the dependency level in such a coarse scale. While a finer level of granularity may be more theoretically interesting for the research purposes, in practice customers are uncomfortable with such fine-grained value assignments. Moreover, we have chosen the median degree of requirements for real industrial data which, we think, is appropriate for the problem.

r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9	r_{10}	r_{11}	r_{12}
100	50	300	80	70	100	1000	40	200	20	1100	10
r_{13}	r_{14}	r_{15}	r_{16}	r_{17}	r_{18}	r_{19}	r_{20}	r_{21}	r_{22}	r_{23}	r_{24}
500	10	10	10	20	200	1000	120	300	50	10	30
r_{25}	r_{26}	r_{27}	r_{28}	r_{29}	r_{30}	r_{31}	r_{32}	r_{33}	r_{34}	r_{35}	
110	230	40	180	20	150	60	100	400	80	40	

Table 1. Feature data from Motorola

5.2 Algorithm Parameter Setup

The parameters we set for our algorithms are based on previous empirical research. We choose parameter value obtained from benchmark test problems [10] and assume they are optimized for our MONRP/RD. Each algorithm was run for maximum 100 generations. The initial population was set to 200. A simple binary GA encoding was used, with each bit to code for a decision variable (the inclusion or exclusion of a customer's preferences). Therefore, the length of a solution is equal to the number of decision variables. The tournament selection, a single-point crossover and bitwise mutation operator for binary-coded GAs are applied. The probability of the crossover operator was set to 0.8 and probability of mutation operator (per gene) was set to $1/mn$, where m is the number of all possible customers and n is the number of solutions in the population. A more detailed introduction of GA is in [14]. Unless there is specification, all the following experiments have followed the above setups.

6 RESULTS AND ANALYSIS

In this section, by using data sets described in Section 5.1, we present the results of applying the NSGA-II algorithm to the following scenario:

- the influences of requirement dependencies on the NSGA-II algorithm performance,
- the scalability of the MONRP/RD,
- the performance of NSGA-II on the real industrial Motorola data set.

To demonstrate the convergence of the NSGA-II algorithm, the initial populations and the final non-dominated solutions denoted by solid dot "$\cdot$" and big circle "$\circ$", respectively, are plotted in figures. Each point represents a subset of satisfied customers and the corresponding requirements for the next release. In the experiments, we consider two objectives, score and cost, which are defined in Equations (3) and (4).

6.1 Influences of Requirement Dependencies on the Results

We first investigate the influences of requirement dependencies on the random data set. The number of customers is fixed to 80 and the number of requirements is set to 40. The results of applying NSGA-II algorithm to the various requirement dependency levels are shown in Figure 2 a)–d). It simulates situations that software requirement dependencies fall into a range of no, weak, median and strong. We observe that NSGA-II guide the population towards the Pareto front. It can be seen that the cost of the initial population denoted by "$\cdot$" becomes the same with the increase of requirement dependencies. In the situation with strong requirement dependencies, initial solutions become a horizontal line, which indicates

that the solutions have costs with the same value. This is partly because the software requirement R_i for each customer tends to include all the requirements with the increase of requirement dependencies. That is, every customer requests all the available requirements, therefore no matter which customer is selected to be satisfied, an equal number of requirements are to be fulfilled for each customer. Another interesting phenomenon is that the number of final non-dominated solutions becomes smaller with the increase of requirement dependencies. An extreme case is that there is only one final non-dominated solution when requirement dependencies are very strong, as shown in Figure 2 d). This can be explained by the fact that fewer feasible solutions exist in the solution space when the requirement dependencies become stronger. This result in turn proves that the MOEAs are very suitable in the MONRP/RD as search based techniques have proved to fit well in complex nonlinear optimization problems [12, 23].

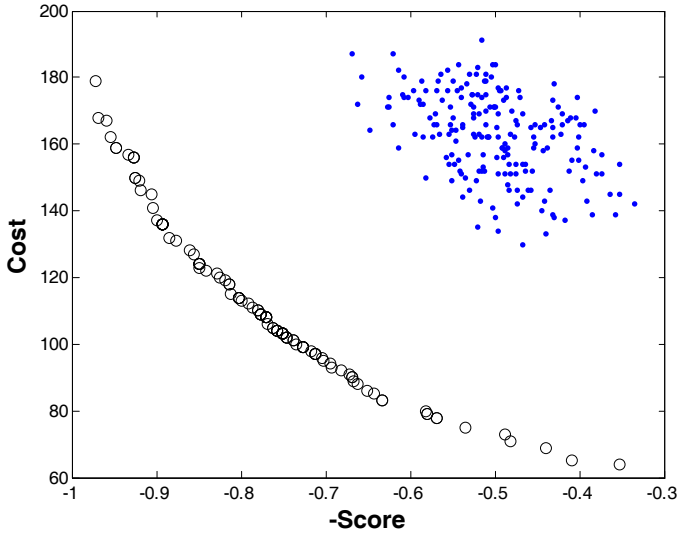
6.2 Scalability

In this section, we apply NSGA-II to the different size of random data set in order to investigate the scalability of the formulated MONRP/RD. Different scenarios are considered to investigate how the algorithm performance scales up with the problem size. We consider the MONRP with the weak requirement dependencies. The number of customers and requirements for each scenario is listed in Table 2.

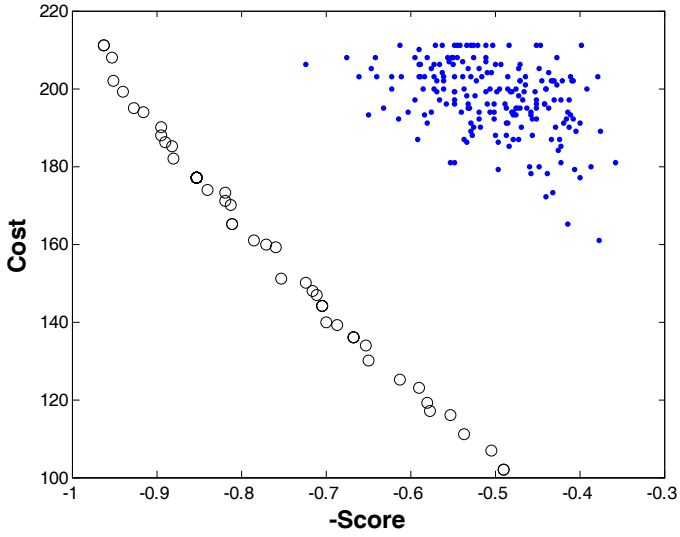
Scale	Customers	Requirements
S_1	20	10
S_2	80	40
S_3	140	100
S_4	200	150

Table 2. Scale test sets

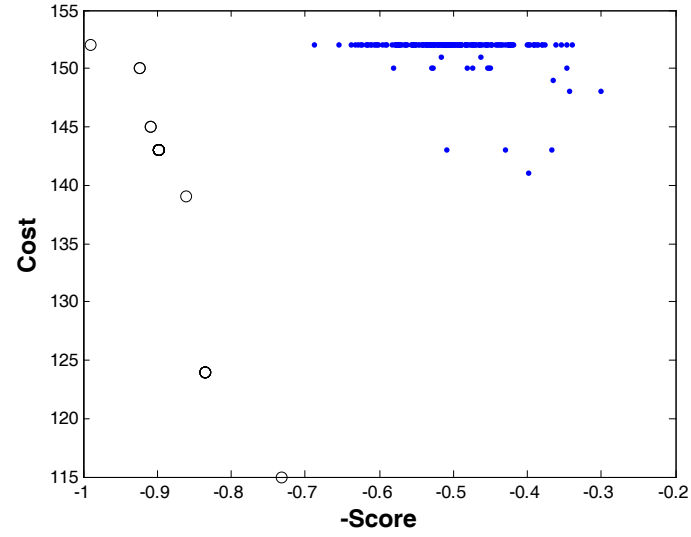
The results of different scenarios are shown in Figure 3 a)–d). The trend that four experiments demonstrate is that the larger the scale of the test problem, the wider the gap between initial population and final non-dominated solutions, which indicates better convergence of our algorithm to test problem of larger size. In other words, the performance improvement offered by NSGA-II scales as the problem scales. In addition, the number of non-dominated solutions increase when the problem size increases as there are more feasible solutions in the search space. The results in turn proves that the MOEAs are very suitable in large scale MONRP/RD, which is consistent with observation that search based techniques fit better in more complex nonlinear optimization problem [12, 23].


Cost
-Score
200
180
160
140
120
100
80
60
-1
-0.9
-0.8
-0.7
-0.6
-0.5
-0.4
-0.3

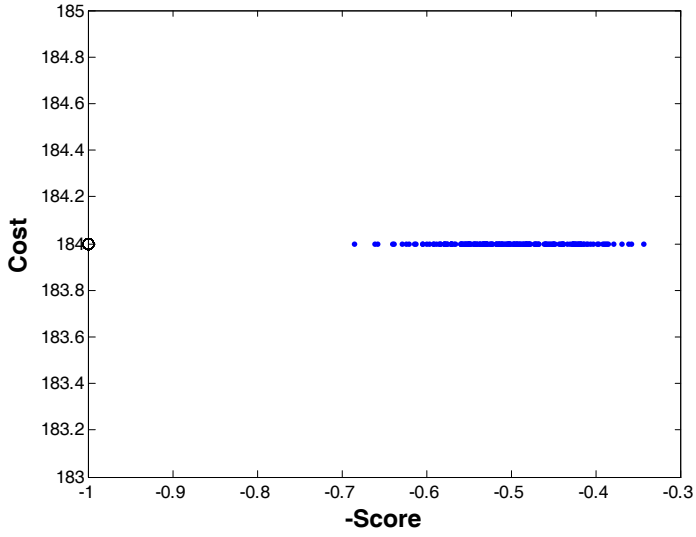
a)


Cost
-Score
220
200
180
160
140
120
100
-1
-0.9
-0.8
-0.7
-0.6
-0.5
-0.4
-0.3

b)

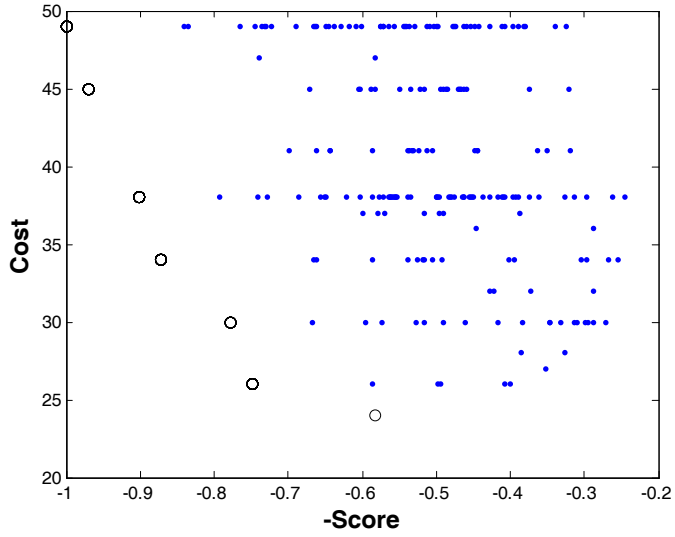


c)

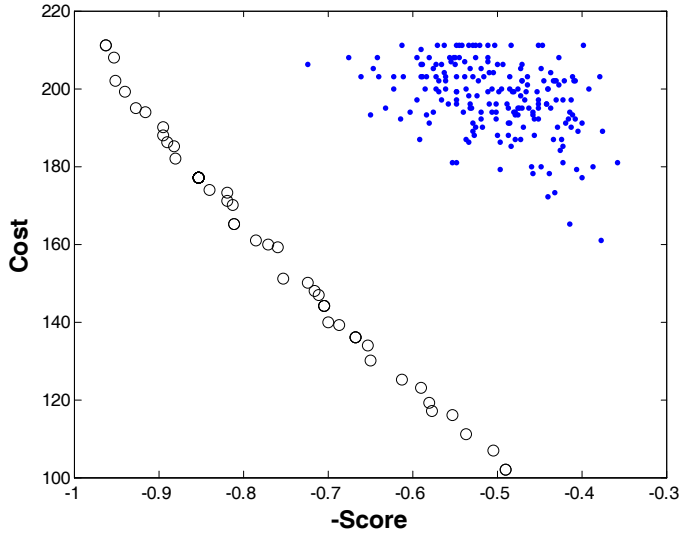


d)

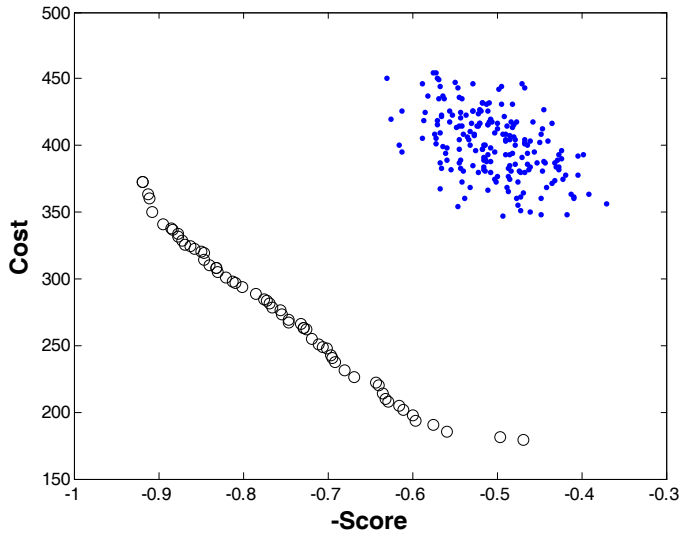
Fig. 2. Influences of the requirement dependencies: a) no requirement dependencies, b) weak requirement dependencies, c) median requirement dependencies, d) strong requirement dependencies


Cost
-Score
50
45
40
35
30
25
20
-1
-0.9
-0.8
-0.7
-0.6
-0.5
-0.4
-0.3
-0.2

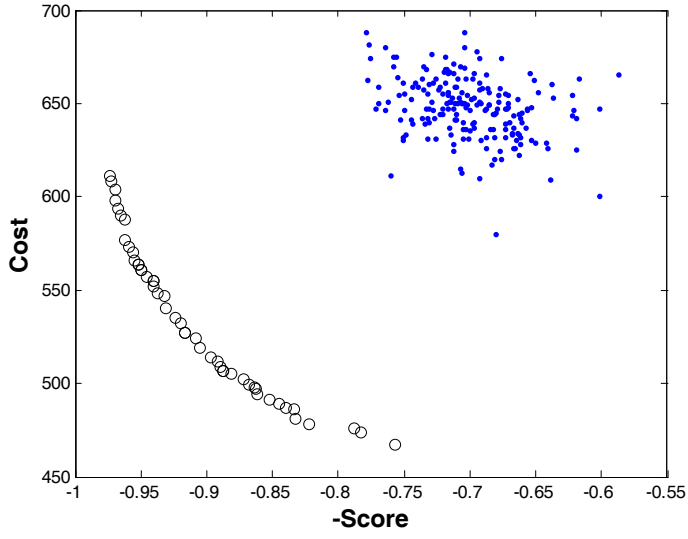
a)


Cost
-Score
220
200
180
160
140
120
100
-1
-0.9
-0.8
-0.7
-0.6
-0.5
-0.4
-0.3

b)



c)



d)

Fig. 3. Investigation of scale problem in MONRP/RD: a) 20 customers, 10 requirements; b) 80 customers, 40 requirements; c) 140 customers, 100 requirements; d) 200 customers, 150 requirements

6.3 Results on Real Data

In this section, we apply NSGA-II to the real industrial data set from Motorola. We consider two scenarios: 4 customers, 35 requirements and 10 customers, 35 requirements. The description of the Motorola data set was given in Section 5.1.

The two objectives in the experiments are the same as described in Section 3:

1. mini-mize the total cost of requirements determined by satisfied customers and
2. maxim-ize the total satisfaction of the customers.

The results are shown in Figure 4 a)–b). The initial population, the population generated by the median generation and the final non-dominated solutions are represented by the solid dot "·", asterisk "*"and big circle "○" symbols plotted in the figures, respectively. Each point represents a subset of customers and their corresponding requirements for the next release.

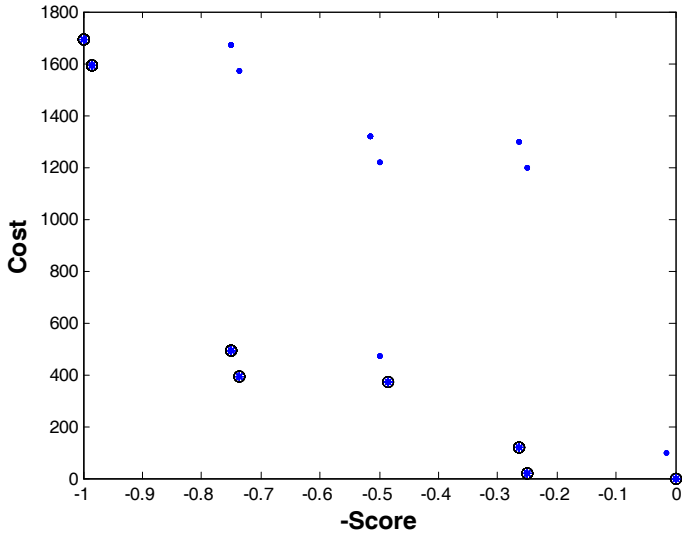
In Figure 4 a), the final non-dominated solutions ("○") overlaps with solutions generated by the median generation. This observation indicates that the NSGA-II converges before the median generation as the problem with the Motorola data set is of small size and thus an easy problem for the NSGA-II. In Figure 4 b), the NSGA-II approach pushes the non-dominated solutions to the Pareto front. As we can see from both Figures 4 a) and b), when the cost decreases, the score (customers' satisfaction) has also reduced. It is interesting to analyze two extreme solutions in Figure 4 a) for the original Motorola data (4 customers and 35 requirements). The downward rightmost solution represents the situation when no customers have been chosen; thus no requirements have been selected and customers' satisfaction and cost are both equal to 0. The upward leftmost solution represents the opposite situation when all 4 customers are chosen. In this scenario, the cost is maximized (the value of the cost is around 1 700) and all 4 customers are satisfied (the value of score is -1). Other solutions in the non-dominated front represent solutions that provide more balance between cost and customers' satisfaction.

The result provides decision-maker for the feasible solutions that balance between the above two objectives. The hidden tensions between the two objectives are revealed implicit in Figures 4 a) and 4 b).

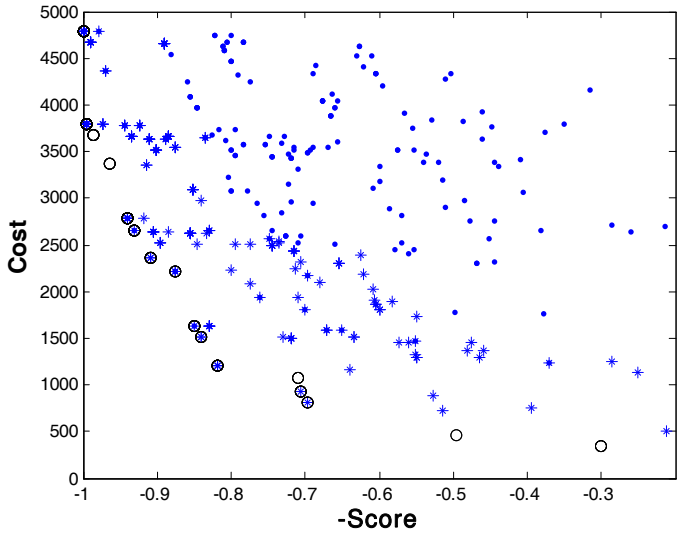
7 RESULTS OF THE EXPERIMENT TO COMPARE ALGORITHM PERFORMANCE

In this section, we present the results of applying different optimization approaches, NSGA-II, SPEA2, IWO/MO (original multi-objective invasive weed optimization), IWO/MO2 (Improved version of IWO/MO) and Random Search, to MONRP/RD. The experiments are conducted to

1. validate our experiment that has used NSGA-II in the last section and
2. find the most suitable algorithm for our MONRP/RD.



a)



b)

Fig. 4. Results of MONRP/RD on Motorola data: a) 4 customers; 35 requirements; b) 10 customers, 35 requirements

To compare the performance of different algorithms, experiments are designed concerning

1. total cost to implement requirements requested by chosen customers and
2. total customer's satisfaction (score). Each algorithm was executed 20 times for each data set.

Final non-dominated obtained by different approaches can be visualized in the figures. Besides, performance metrics of the non-dominated solutions generated by different approaches are provided and corresponding numerical analysis is given.

7.1 Performance Metrics

The performance of two search algorithms may overlap, which makes it difficult to compare them visually. Therefore, we defined two performance metrics based on the two fundamental roles that the search algorithms need to fulfil (see Section 4), i.e.

1. guiding the population towards Pareto optimal to achieve good convergence and
2. maintaining diversity in the population to have full exploration of the search space.

We define convergence and diversity metrics accordingly for the above two roles.

A reference non-dominated front was constructed to compare the non-dominated produced by different algorithms. The reference non-dominated front is obtained using the following method. First, all the solutions obtained from different approaches are merged together, then the non-dominated solutions among the merged population of solutions will form the reference non-dominated front. The reference non-dominated front denotes the best available approximation to the real Pareto front.

The non-dominated fronts generated by the different search algorithms may partly contribute to the reference non-dominated front. Therefore, the first performance metric, named convergence metric ρ, is defined as the percentage of solutions produced by each algorithm on the reference non-dominated front:

$$\rho = n_a / n_r \tag{6}$$

where n_a is the number of solutions produced by each algorithm on the reference non-dominated front and n_r is the number of solutions on the reference non-dominated front. Thus, for the most converged non-dominated front, the value of metric ρ would have a high percentage value.

The second performance metric, diversity metric Δ, is to measure the extent of spread achieved among the obtained non-dominated solutions by each algorithm [10]. Δ is defined as follows:

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{N-1} |d_i - \hat{d}|}{d_f + d_l + (N-1)\hat{d}}. \tag{7}$$

Here, d_f and d_l are the Euclidean distances between the two boundary solutions of the reference non-dominated front and the two boundary solutions of the non-dominated front produced by each algorithm; d_i is the Euclidean distance between consecutive solutions in the non-dominated front produced by each algorithm and $\hat{d}$ is the average of all distances d_i, $i = 1, 2, \ldots, (N-1)$, where N is the number of solutions that form the non-dominated front. With N solutions, there are $(N-1)$ consecutive distances. A good distribution would make all distances d_i equal to $\hat{d}$ and would make $d_f = d_l = 0$ (the solutions in the non-dominated front generated by the algorithm contribute the boundary solutions of reference non-dominated front). Thus, for the most widely and uniformly spread-out non-dominated front, the value of metric Δ would have a near-zero value. Equation (7) only addresses two objective optimization problems but can be extended to MOP with more than two objectives.

7.2 Results

The non-dominated fronts produced by different algorithms on the random and Motorola data sets are shown in Figures 5 a) and 5 b). In the figures, the symbols "+", "*", big "○", "x" and big "." represent non-dominated solutions obtained by the random search, the SPEA2, the NSGA-II, IWO/MO and IWO/MO2, respectively. More quantitative analysis are given in Table 3. Quantitative analysis of the random data set in the table corresponds to results visualized in Figure 5 a) and quantitative analysis of the Motorola data set in the table corresponds to results visualized in Figure 5 b).

For the random data set, IWO/MO2 outperforms other approaches in terms of convergence to the real Pareto front measured by convergence metric ρ and diversity of the non-dominated fronts' distribution measured by diversity metric Δ. Among the FIVE search approaches, IWO/MO2 contributes 36.99 % of the reference non-dominated front and its diversity metric value is as small as 0.051, indicating the non-dominated front it produced has the best distribution compared with that of the other four algorithms. Moreover, there are few duplicate solutions obtained by the three out of five algorithms (IWO/MO2, NSGA-II and SPEA2) in the final reference non-dominated front. In this case, there is almost no overlapped solution in the three sets of solutions obtained by the three algorithms. For this reason, adding up the numerical values of convergence metric ρ in random data set row is almost equal to 100 %. This interesting observation indicates different MOEAs can be complementary to address MONRP/RD.

Unlike the results of the random data set where IWO/MO2 outperforms other algorithms significantly, the Motorola data set tells a slightly different story. In Figure 5 b), we observe that the non-dominated fronts produced by SPEA2, the NSGA-II, IWO/MO and IWO/MO2 almost overlap completely. This means four algorithms perform equally well. For the Motorola data set, we can observe from Table 3 that even random search contributes the reference non-dominated front in some cases. The evolutionary algorithms having occasional good or bad performance may be due to the different characteristics and scale size of data sets. In terms of

random search, this may occur when the size of data set is comparatively small and therefore denotes the MONRP/RD problem presented by the Motorola data set is a relatively easy optimization problem.

	Random data set					**Motorola data set**				
	Random search	SPEA2	NSGA-II	IWO/MO	IWO/MO2	Random search	SPEA2	NSGA-II	IWO/MO	IWO/MO2
Convergence metric ρ	0	26.71%	36.3%	0	**36.99%**	4.17%	91.25%	95.83%	91.25%	95.83%
Diversity metric Δ	0.7248	0.7307	0.5636	0.12	**0.051**	0.1075	0	0	0	0

Table 3. Values of the performance metric on random and Motorola data sets

8 THREATS TO VALIDITY

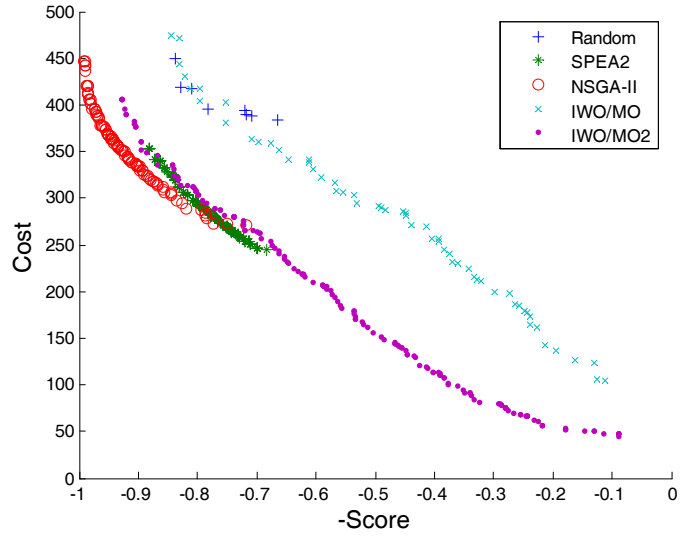
The results of the current research are subject to limitations which are inherent in any empirical study. This section sets out these limitation and indicate how they may affect the degree to which it is possible to generalize the results. The synthetic data sets might not be regarded as the fair representative; so the real industrial Motorola data set was used to provide a better investigation for this research. More real problems would result in a greater ability to generalize our findings. However, it is known it is difficult to obtain real industrial data sets since they are typically considered confidential by the companies.

NSGA-II, SPEA2 and IWO/MO used in this study denote three popular multi-objective search algorithms. In addition, we propose an improved version of IWO/-MO – IWO/MO2. Evolutionary algorithms have been very effective in many problems. Although they do not guarantee to find the best solutions for a given problem, they are able to obtain near-optimal solutions for decision making which is insightful for our MONRP/RD. We believe the chosen algorithms are appropriate to address the main research questions in this paper. Our results have demonstrated that evolutionary optimization algorithms are superior to random search in MONRP/RD. To demonstrate the the performance of different algorithms for MONRP/RD qualitatively, two performance metrics are also defined.

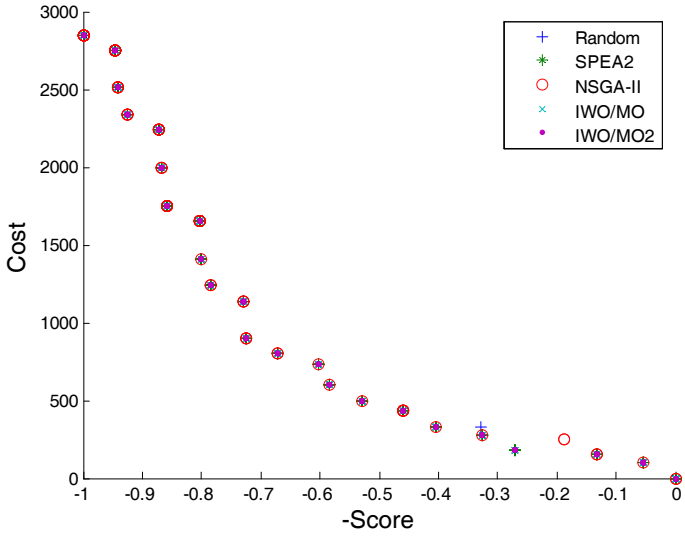
The parameter values of the evolutionary algorithms can influence the results. We choose parameter values obtained from benchmark test problems and assume they are optimized for our MONRP/RD. However, this assumption is the same to minimize environmental factor on the performance during the experimentations. For a specific algorithm, performance could have been further improved by individual fine tuning empirically or through systematic experimentation; we leave further investigation of this as a future work.

9 RELATED WORK

Requirement analysis and optimization have been very popular in the area of requirement engineering. Requirement priorities assignment methodologies have been



a)



b)

Fig. 5. Comparison of the random search, SPEA2, NSGA-II, IWO/MO and IWO/MO2: a) Results for the random data set; b) Results for the Motorola data set

investigated in Karlsson's work [17, 19, 18]. Stratus for selecting an optimal set of requirements for implementation are also proposed.

The term *Next Release Problem* was first suggested in [2] by Bagnall et al. He also described various metanephritic approaches to find near optimal solutions for the formulated next release problem. Later, Van den Akker et al. [1] uses integer linear programming to find exact solutions within budgetary constraints. A comparison of both analytical and evolutionary approaches for prioritizing software requirements is performed in [20]. Greer and Ruhe proposed an iterative genetic algorithm for NRP and applied it to the real world problem [15]. They addressed the NRP by minimizing total penalty and maximizing total benefit in the form of an integrated objective function with user defined weights for each objective.

More recently, multi-objective formulation of the NRP has become popular. Finkelstein et al. discuss the issue of *fairness* in requirement analysis [13]. Saliu and Ruhe formulated a two-objective MONRP that balances the tension between user-level and system-level requirements [25]. Zhang et al. considered value and cost as two criteria in their multi-objective next release problem formulation [28]; the scalability of the approach, in terms of the number of requirements and customers was also discussed in the paper. Later, Zhang et al. summarized existing research and described future challenges for requirement optimization using search based methods [27] and compared different multi-objective approaches in [11, 30]. Recently, Zhang et al. [29] consider requirement interaction management in next release problem. We formulate the requirement-dependency-based MONRP by using *dependency graph*. We also test the scalability and influence of requirement on the model. Furthermore, we compare various MOEAs on the model to determine the most suitable algorithm for it.

10 CONCLUSION AND FUTURE WORK

This paper formulates requirement-dependency based multi-objective next release problem. Two objectives, cost of chosen requirements and customers' satisfaction are considered. MOEAs are applied to provide decision-maker for the feasible solutions that balance between the above two objectives.

The experiments in this paper demonstrate that search based techniques, such as MOEAs, can be very effective to both synthetic and real industrial data sets for the formulated MONRP/RD. The hidden tensions between the two aimed objectives are revealed implicit in these data sets.

In addition, we manipulate the scale of the MONRP/RD as well as the degree of requirement dependencies to understand the influences of them on the performance of our applied approaches. The results show that the MONRP/RD becomes more complex with the increase of problem scale and have less feasible solutions with the increase of the degree of the requirement dependencies. Another observation is that the performance of the algorithm become better when the problem scales, which in turn proves that the MOEAs are very suitable in large scale MONRP/RD as search

based techniques have proved to fit well in more complex nonlinear optimization problem [12, 23].

The paper also gives a comparative study of the random search and four more sophisticated MOEAs, SPEA2, NSGA-II, IWO/MO, as well as a proposed algorithm IWO/MO2. The results show MOEAs outperform the random search. Among the four MOEAs, results in large scale MONRP/RD (the random data set) show that the proposed IWO/MO2 outperforms other three approaches in terms of both convergence and diversity but the four approaches can be complementary to address MONRP/RD.

Future work will verify these findings by applying more search techniques and also classical optimization techniques, such as non-linear programming. Considering other types of requirement dependencies in MONRP is another interesting direction.

Acknowledgements

The research reported herein was supported in part by the Fundamental Research Funds for the Central Universities of China (No. NS2012074), the Scientific Research Startup Fund of NUAA of China (No. S0945-042) and National Natural Science Foundation of China (No. 61170043).

REFERENCES

[1] Akker, S.—Diepen, G.—Versendaal, J.: Software Product Release Planning Through Optimization and What-If Analysis. Information & Software Technology, Vol. 50, 2008, No. 1-2, pp. 101–111.

[2] Smith, V.—Bagnall, A. J.—Whittley, I.: The Next Release Problem. Information & Software Technology, Vol. 43, 2001, No. 14, pp. 883–890.

[3] Baker, K.—Harman, M.—Skaliotis, A.: Search Based Approaches to Component Selection and Prioritization for the Next Release Problem. In ICSM 2006, pp. 176–185.

[4] Coello Coello, C. A.—Veldhuizen, D.: Evolutionary Algorithms for Solving MultiObjective Problems (Genetic and Evolutionary Computation). Springer-Verlag New York, Inc., Secaucus, NJ, USA 2006.

[5] Carlshamre, P.—Sandahl, K.—Lindvall, M.—Regnell, B.—Dag, J.: An Industrial Survey of Requirements Interdependencies in Software Product Release Planning. IEEE International Symposium on Requirements Engineering (RE 2001) 2001, pp. 84–93.

[6] Dag, J.—Regnell, B.—Carlshamre, P.—Andersson, M.—Karlsson, J.: A Feasibility Study of Automated Natural Language Requirements Analysis in Market-Driven Development. Requir. Eng., Vol. 7, 2002, No. 1, pp. 20–33.

[7] Coello Coello, C. A.: Evolutionary Multiobjective Optimization: A Historical View of the Field. IEEE Computational Intelligence Magazine, Vol. 1, 2006, No. 1, pp. 28–36, 2006.

[8] Srinivas, N.—Deb, K.: Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. Evolutionary Computation, Vol. 2, 1994, No. 3, pp. 221–248.

[9] Deb, K.:. Multi-Objective Optimization Using Evolutionary Algorithms. John Wiley & Sons, Inc., New York, NY, USA 2001.

[10] Deb, K.—Agrawal, S.—Meyarivan, T.: A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. IEEE Trans. Evolutionary Computation, Vol. 6, 2002, No. 2, pp. 182–197.

[11] Durillo, J.—Zhang, Y.—Alba, E.—Harman, M.—Nebro, A.: A Study of the Biobjective Next Release Problem. Empirical Software Engineering, Vol. 16, 2011, No. 1, pp. 29–60.

[12] Figueira, J.—Greco, S.—Ehrgott, M.: Multiple Criteria Decision Analysis: State of the Art Surveys. Springer Verlag, Boston, Dordrecht, London 2005.

[13] Finkelstein, A.—Harman, M.—Mansouri, S.—Ren, J.—Zhang, Y.: A Search Based Approach to Fairness Analysis in Requirement Assignments to Aid Negotiation, Mediation and Decision Making. Requir. Eng., Vol. 14, 2009, No. 4, pp. 231–245.

[14] Goldberg. D.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA 1989.

[15] Greer, D.—Ruhe, G.: Software Release Planning: An Evolutionary and Iterative Approach. Information & Software Technology, Vol. 46, 2004, No. 4, pp. 243–253.

[16] Harman, M.: The Current State and Future of Search Based Software Engineering. In Lionel C. Briand and Alexander L. Wolf (Eds.): Proceedings of Workshop on the Future of Software Engineering (FOSE '07), pp. 342–357.

[17] Karlsson, J.: Software Requirements Prioritizing. In ICRE 1996, pp. 110–116.

[18] Ryan, K.—Karlsson, J.: Prioritizing Software Requirements in an Industrial Setting. In ICSE 1997, pp. 564–565.

[19] Karlsson, S.—Ryan, K.: Improved Practical Support for Large-Scale Requirements Prioritising. Requirements Engineering Journal, Vol. 2, 1997, No. 1, pp. 19–27.

[20] Karlsson, S.—Regnell, B.: An Evaluation of Methods for Prioritizing Software Requirements. Information & Software Technology, Vol. 39, 1998, No. 14-15, pp. 939–947.

[21] Kundu, D.—Suresh, K.—Ghosh, S.—Das, Sw.—Panigrahi, B. K.—Das, Sa.: Multi-Objective Optimization With Artificial Weed Colonies. Information Science, Vol. 181, 2011, No. 12, pp. 2441–2454.

[22] Mehrabian, A. R.—Lucas, C.: A Novel Numerical Optimization Algorithm Inspired from Weed Colonization. Ecological Informatics, Vol. 1, 2006, No. 1, pp. 355–366.

[23] Miettinen, K.: Nonlinear Multiobjective Optimization. International Series in Operations Research and Management Science. Vol. 12, Kluwer Academic Publishers, Dordrecht 1999.

[24] Miettinen, K.: Nonlinear Multiobjective Optimization. Kluwer Academic Publishers 1999.

[25] Saliu, M.—Ruhe, G.: Bi-Objective Release Planning for Evolving Software Systems. In ESEC/SIGSOFT FSE 2007, pp. 105–114.

[26] Stadler, W.: A Survey of Multicriteria Optimization or the Vector Maximum Problem. Part I: 1776C1960, Journal of Optimization Theory and Applications, Vol. 29, 1979, No. 1, pp. 1–52.

[27] Finkelstein, A.—Zhang, Y.—Harman, M.: Search Based Requirements Optimisation: Existing Work and Challenges. In REFSQ 2008, pp. 88–94.

[28] Harman, M.—Zhang, Y.—Mansouri, S.: The Multi-Objective Next Release Problem. In GECCO 2007, pp. 1129–1137.

[29] Zhang, Y.—Harman, M.: Search Based Optimization of Requirements Interaction Management. In SSBSE 2010, pp. 47–56.

[30] Zhang, Y.—Harman, M.—Finkelstein, A.—Mansouri, S.: Comparing the Performance of Metaheuristics for the Analysis of Multi-Stakeholder Tradeoffs in Requirements Optimisation. Information & Software Technology, Vol. 53, 2011, No. 7, pp. 761–773.

[31] Zitzler, E.—Laumanns, M.—Thiele, L.: SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Technical report, Swiss Federal Institute of Technology 2001.

Xinye Cai got his B. Eng. degree at Electronic and Information Engineering Department of Huazhong Univeristy of Science and Technology in China in 2004. He received his M. Sc. degree at Electronic Department, University of York, UK in 2006. Later, he received his Ph. D. degree at Electrical and Computer Engineering Department of Kansas State University. Currently, he is a lecturer at College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics. His main research interests include evolutionary computation, requirement engineering and computational biology.

Ou Wei received the Ph. D. degree in computer science from the University of Toronto in 2009. He is currently a faculty member at the Nanjing University of Aeronautics and Astronautics. His main research interests include software verification and requirements engineering.

Zhiqiu Huang received the Ph. D. degree from Nanjing University of Aeronautics and Astronautics in 1999. He is currently a Professor at the Nanjing University of Aeronautics and Astronautics. His main research interest is software engineering.