Received August 4, 2018, accepted October 1, 2018, date of publication October 10, 2018, date of current version November 8, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2875122

Supporting Many-Objective Software Requirements Decision: An Exploratory Study on the Next Release Problem

JIANGYI GENG, SHI YING, XIANGYANG JIA, TING ZHANG, XUAN LIU, LANQING GUO, AND JIFENG XUAN, (Member, IEEE)

School of Computer Science, Wuhan University, Wuhan 430072, China

Corresponding author: Xiangyang Jia (jxy@whu.edu.cn)

This work was supported in part by the National Key R&D Program of China under Grant 2016YFC1202204, in part by the National Natural Science Foundation of China under Grant 61872273, Grant 61876136, and Grant 61502345, in part by the Young Elite Scientists Sponsorship Program by CAST under Grant 2015QNRC001, and in part by the Technological Innovation Projects of Hubei Province under Grant 2017AAA125.

ABSTRACT The decision of which requirements should be satisfied in the next release is crucial to software company. The next release problem, a family of requirements selection decision, aims to maximize profits by satisfying requirements to balance customer profits and development costs. However, due to diverse practical scenarios, solutions to the next release problem have to face many different objectives. In this paper, we propose an exploratory study on the many-objective next release problem with five evolutionary optimization algorithms. The goal of this study is to use the experimental results to assist project managers to make the requirements decision in the scenario of many decision objectives. This study focuses on four research questions, including the effectiveness of optimization, the significance of results, the distribution of metric values, and the correlation between metrics. We design the study to explore five objective functions of the next release problem, including the maximum of customer profits, the minimum of requirements costs, the fairness of requirements selection, etc. The study is conducted on 12 benchmark instances from three real-world projects via evaluating six metrics. Our study suggests that among algorithms in comparison, either eMOEA or IBEA is the best choice for the many-objective next release problem.

INDEX TERMS Requirements decision, the many-objective next release problem, fairness analysis, search-based requirements engineering

I. INTRODUCTION

Requirements selection is an inevitable phase that determines the benefits from customers [1], [2]. Software companies (or organizations) try to satisfy all requirements from customers to pursue the maximum value of profits, such as incomes, subscribers, or reputation. For large-scale software projects, it is impossible to satisfy all potential requirements due to the tremendous development costs, such as the budget or the time. To balance the customer profits and the development costs, a software company has to decide which requirements should be completed in the next software release.

The Next Release Problem (NRP), a family of requirements selection problems, aims to find out the best decision of satisfying requirements from customers. In a large project, exhausting every solution is time-consuming due to the large search space. Bagnall *et al.* [3] proposed the first model of the NRP. This work has examined the performance of several heuristics, such as hill climbing and genetic algorithms. Jiang *et al.* [4] designed an approximate backbone based multilevel algorithm to pursue the maximum profits under a limited budget. Ren [5] has pushed the NRP into a new model of multiple objectives: the multi-objective NRP is to simultaneously maximize the customer profits and minimize the requirements costs. Researchers have proposed several models and methods to support the decision of the multi-objective NRP, including fairness analysis [6], robust analysis [2], interactive optimization [7], and hardness exploration [8].

Under the umbrella of the multi-objective NRP, all the existing work copes with two objectives, e.g., profits and costs [9], or profits and fairness [6]. A general method of solving the multi-objective NRP is to conduct a *Pareto frontier* [10], which implies the set of current optimal solutions. Based on such multi-objective NRP, a software company selects requirements to balance two conflicting objectives for the next release. In practice, however, more objectives exist.

2169-3536 © 2018 IEEE. Translations and content mining are permitted for academic research only.
Personal use is also permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

To determine the requirements selection, a company has to simultaneously cope with three or more objectives. For instance, the maximum of customer profits, the minimum of requirements costs, and the fairness of requirements selection should be considered together to find out the optimal decision. The assumption of two objectives in the NRP is violated; meanwhile, finding out the Pareto frontiers of more than two objectives may be infeasible [11]. There exists no prior study to provide support to the requirements decision for the NRP with many decision objectives.

In this paper, we addressed the NRP model with more than three objectives, namely the Many-Objective NRP (ManyNRP for short). We proposed an exploratory study on the ManyNRP with five state-of-the-art evolutionary optimization algorithms: NSGA-II, DBEA, NSGA-III, eMOEA, and IBEA. We designed the study to explore how to determine five objectives of the ManyNRP, including the maximum of customer profits, the minimum of requirements costs the fairness of requirements selection, etc. This study focuses on four research questions, including the effectiveness of optimization, the significance of results, the distribution of metric values, and the correlation between metrics. The study is conducted on 12 benchmark instances from three real-world projects, Eclipse, Gnome, and Mozilla; six metrics are used to evaluate the effectiveness of solving ManyNRP. Our study suggests that among algorithms in comparison, either eMOEA or IBEA is the best choice for the many-objective next release problem for the evaluation of most metrics; NSGA-II can obtain the best solutions for the metric of Spacing. This study provides a general way to support the decision of many-objective requirements selection.

Application Scenario: A project manager may face many optimization objectives (e.g., four or more) during the phase of requirements analysis. Due to the large number of objectives, it is challenging to manually select customers that should be satisfied in the next release. Our work aims to provide a general solution to this scenario; our experiment shows that eMOEA or IBEA can be directly used to assist the decision by the project manager.

This paper makes the following major contributions:

- We addressed the ManyNRP, namely the scenario of simultaneously optimize more than three objectives for the NRP. To the best of our knowledge, this is the first work that seeks the solution to the emerging problem of the ManyNRP.
- We proposed an exploratory study on the ManyNRP via examining the effectiveness of five optimization algorithms on 12 instances from three real-world projects.
- We explored the optimization via answering four research questions and suggest that eMOEA and IBEA are the two best algorithms, which outperform the others under comparison.

The remainder of this paper is organized as follows. Section II presents the background and the problem definition. Section III shows the design of our study on instances from three projects. Section IV shows the result of exploring the effectiveness of solving the ManyNRP. Section V discusses the threats to the validity of our work. Section VI lists the related work and Section VII concludes this paper.

II. BACKGROUND AND PROBLEM DEFINITION

In this section, we present the background and the problem definition of the many-objective NRP.

A. THE NEXT RELEASE PROBLEM

Requirements analysis is to model potential real-world applications into software descriptions [12]. The NRP is to optimize the selection of satisfied requirements in the next software release. Fig. 1 illustrates the NRP with an example. This example is extracted from the project of Eclipse Java Developer Tools (JDT). JDT is a static Java analysis tool, which can be used to analyze and transform Java source code.

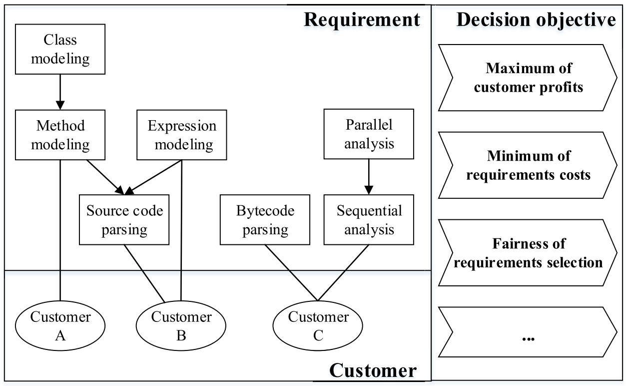


FIGURE 1. Illustration of the many-objective NRP with seven requirements and three customers.

As shown in Fig. 1, three customers request for seven requirements in the release. There exist dependencies among requirements. For instance, the requirement of *method modeling* relies on completing the requirement of *source code parsing*. The dependencies enlarge the complexity of requirements selection. Since the dependencies are known to the software company, these dependencies among requirements can be simplified as direct requests between customers and requirements. Bagnall *et al.* [3] and Xuan *et al.* [13] have introduced the method of simplifying the NRP by removing such dependencies.

To determine whose requirements should be satisfied (i.e., which customer should be prioritized), a software company considers both customer profits and requirements costs [14]. In practice, customer profits and requirements costs are conflicting for the decision of requirements because maximizing the customer profits leads to the maximum of requirements costs. Bagnall *et al.* [3] first formalized the NRP by optimizing the requirements selection under a limited budget. This is a single-objective optimization problem and has become one of pioneering scenarios of search-based software engineering [6].

It is straightforward to convert the limited budget into another objective for the NRP. This leads to another model that simultaneously optimizes two objectives, i.e., to maximize customer profits and to minimize customer costs. This model is referred to as the multi-objective NRP. Note that in the literature of the NRP, such multi-objective NRP only contains two objectives. Various multi-objective optimization models have been proposed or applied to solve the multi-objective NRP to balance the conflicts between two decision objectives. For instance, Finkelstein *et al.* [6] have addressed the conflict between the number of satisfied requirements (i.e., the maximum value) and the fairness of satisfied ones for each customer (i.e., the minimum of the standard deviation of satisfied requirements). In their study, even the requested requirements by a customer are not completely satisfied, the satisfied ones can bring benefits. The scenario of such multi-objective NRP can better modeling the real-world requirements selection than the single-objective NRP.

B. MANY-OBJECTIVE OPTIMIZATION

The multi-objective NRP in Section II-A has not provided a solution to the optimization of more than two objectives. Recently, many-objective optimization is proposed to address such optimization problem [15]. The many-objective optimization is a specific model of the multi-objective one, but contains more than three objectives. These objectives enlarges the scale of the search space. Let X be a generic form of a variable. The many-objective optimization model can be generally stated as follows,

$$\begin{aligned} &\text{minimize } \overrightarrow{f}(X) = [f_1(X), f_2(X), \ldots, f_t(X)]^T, \quad t \geq 4 \\ &\text{subject to: } g_j(X) = 0, \quad 1 \leq j \leq p \\ &\qquad\qquad h_k(X) \geq 0, \quad 1 \leq k \leq q \\ &\qquad\qquad x_i^l \leq x_i \leq x_i^u, \end{aligned}$$

where t is the number of objectives, p and q are the number of equality constraints and inequality constraints, x_i is the ith component of the variable X, and x_i^l and x_i^u are the lower boundary and the upper boundary, respectively.

An assumption of many-objective optimization is that the number of objectives is four or more.[1] This unique feature makes many-objective optimization hard to be solved. Most existing algorithms for multi-objective optimization lose their magic for many-objective optimization. For instance, NSGA-II by Deb *et al.* [10] is widely-used in optimizing two conflicting objectives. In NSGA-II, solutions are ranked according to the two objectives; then a Pareto frontier is conducted to converge top-ranking solutions. However, the assumption of four or more objectives in many-objective optimization makes ranking solutions difficult: an optimization algorithm cannot distinguish solutions due to the conflicting objective values. A study by Jaimes *et al.* [16] shows that finding the top-ranking solutions via multi-objective optimization requires examine over 62K, 1,953K, and 1,708,984K solutions for 4, 5, and 7 objectives, respectively. To address the weakness by the multi-objective optimization, several many-objective optimization algorithms are proposed, such as NSGA-III (an upgraded version of NSGA-II via embedding reference points) and IBEA.

C. THE MANY-OBJECTIVE NEXT RELEASE PROBLEM

In requirements engineering, either selecting a set of customers or selecting a set of requirements leads to the decision of satisfying requirements. In this paper, we followed [3] to treat the solution to the NRP as the selection of customers; then requirements requested by these selected customers can be directly extracted. We define the general problem of the Many-Objective NRP (*ManyNRP* for short) according to the simplified model [3].

Definition 1: The simplified ManyNRP.

Given a set R of candidate requirements and a set S of customers, each requirement $r_j \in R$ $(1 \leq j \leq m)$ has a cost c_j and each customer $s_i \in S$ $(1 \leq i \leq n)$ has a profit w_i. A request $q_{i,j} = 1$ or 0 indicates that a customer s_i requests for satisfying a requirement r_j in the next release or not. Followed [3], a solution $X \subseteq S$ is defined as a subset of all customers. The selection of customers can be directly transferred into the selection of requirements. Given a solution X, the selection of requirements is $R(X) = \cup_{s_i \in X, q_{i,j}=1}\{r_j\}$.

The goal of the ManyNRP is to find out an optimal solution X^* to simultaneously minimize given t objectives. Each minimum objective function $f_d(X)$ $(1 \leq d \leq t)$ is defined according the application scenario.[2]

Taking Fig. 1 as an example, to optimize the requirements selection of three customers and seven requirements, many objectives can be considered, including maximizing the customer profits, minimizing the requirements costs, satisfying the fairness of requirements, balancing the ratio of selected requirements, etc. Due to diverse customer profits and requirements costs, the ManyNRP can be adapted to different application scenarios.

III. AN EXPLORATORY STUDY ON THE MANYNRP

We present an exploratory study on the effectiveness of the ManyNRP via evaluating five algorithms on 12 instances from three projects, Eclipse, Gnome, and Mozilla.

A. STUDY SETUP

Requirements are one of private data in software companies. Most of existing work uses randomly generated data to conduct experiments. In our study, we followed [17]–[20] to use 12 relatively-realistic NRP instances, which are extracted from three real-world projects [13], including Eclipse, Gnome, and Mozilla.[3] All these NRP instances are mined by the analogy between the requests of requirements by customers and the bug reports by testers or end users.

[1] In some problems, finding out a solution to three-objective optimization is as hard as that to two-objective optimization [16]. In this paper, we address the problem with four or more objectives and do not discuss the potential theory of the hardness of problem solving.

[2] Maximizing an objective function can be transferred by minimizing its opposite value.

[3] Dataset of the NRP instances, http://cstar.whu.edu.cn/p/nrp/.

TABLE 1. Statistics of 12 instances from Eclipse, Gnome, Mozilla.

Instance	Eclipse				Gnome				Mozilla			
	nrp-e1	nrp-e2	nrp-e3	nrp-e4	nrp-g1	nrp-g2	nrp-g3	nrp-g4	nrp-m1	nrp-m2	nrp-m3	nrp-m4
# Requirements	3502	4254	2844	3186	2690	2650	2512	2246	4060	4368	3566	3643
# Customers	536	491	456	399	445	315	423	294	768	617	765	568
Requirement cost	1-7	1-7	1-7	1-7	1-7	1-7	1-7	1-7	1-7	1-7	1-7	1-7
Customer profit	10-50	10-50	10-50	10-50	10-50	10-50	10-50	10-50	10-50	10-50	10-50	10-50
#Requests by customer	4-20	5-30	4-15	5-20	4-20	5-30	4-15	5-20	4-20	5-30	4-15	5-20

A bug report is an online document that records the description of a newly-submitted bug [21]. The values of instances are obtained according to different bug report data as well as related testers or end users. Table 1 lists the basic statistics of these NRP instances from three projects. Each project consists of four instances.

In our study, we examined five objectives in the ManyNRP. As mentioned in Section II-B, diverse application scenarios can lead to many objectives. This study does not aim to exhaustively check the results of all existing objectives. Instead, we aim to propose a general method to explore solutions to the ManyNRP; meanwhile, under the assumption of many objectives, we do not discuss which single solution is the best. Our study involves five objectives as follows.

1) MAXIMUM OF CUSTOMER PROFITS

The sum of customer profits is the major objective in the early stage of the NRP. The profits to the company is the original intention of proposing the NRP model [3]. We define this objective as follows,

$$\text{maximize } f_1(X) = \sum_{s_i \in X} w_i.$$

2) MINIMUM OF REQUIREMENTS COSTS

The sum of requirements costs is expected to be the minimum. In the first original model [3], the cost of implementing requirements is a pre-defined budget. Then Zhang *et al.* [22] first treated this budget as a minimum objective. We define it as follows,

$$\text{minimize } f_2(X) = \sum_{r_j \in R(X)} c_j.$$

3) COVERAGE OF REQUIREMENTS FOR CUSTOMERS

From the perspective of the company, an ideal solution is to satisfy all requirements that are requested by customers. Finkelstein *et al.* [6] proposed the first analysis to check the coverage of requirements by all customers. In our study, we check this coverage via minimizing the standard deviation of satisfied requirements for all customers. Let $\sigma(v)$ be the standard deviation of a set of values $\{v\}$, i.e., $\sigma(v) = \sqrt{\sum(v-\overline{v})^2}$, where $\overline{v}$ is the average of elements in $\{v\}$. Let $R(s_i)$ be the set of satisfied requirements for the customer s_i, i.e., $R(s_i) = \cup_{r_j \in R(X), q_{i,j}=1}\{r_j\}$. Note that from the definition of $R(s_i)$, even not all requirements from a customer are satisfied, a part of requested requirements can be counted for a customer. For instance, if a customer requests for five requirements, then three out of five can be in the next release via satisfying requirements for other customers. The objective is defined as follows,

$$\text{minimize } f_3(X) = \sigma(|R(s_i)|)$$

where $|\cdot|$ is the cardinality of a set $\cdot$.

4) FAIRNESS OF CUSTOMERS

The ratio between the satisfied requirements and the total requests is important to a customer. Harman *et al.* [2] have analyzed the sensitivity of implementing requirements for customers. We define the fairness of customers via dividing the number of satisfied requirements by the total requests. Let $A(s_i)$ be the set of total requested requirements and $A(s_i) = \cup_{q_{i,j}=1}\{r_j\}$. This objective is defined as follows,

$$\text{minimize } f_4(X) = \sigma\left(\frac{|R(s_i)|}{|A(s_i)|}\right).$$

5) FAIRNESS OF RESOURCE ALLOCATION

Finally, we examine the fairness of the resource allocation in requirements selection, i.e., how much is spent to develop requirements for one customer [6]. This objective is defined as follows,

$$\text{minimize } f_5(X) = \sigma\left(\sum_{r_j \in R(s_i)} c_j\right).$$

We implemented the experiment via Java JDK 1.7 on the top of the MOEA framework[4] and the jMetal lib[5] [23]. In the experiment, each algorithm with each setup repeats running for 30 times and the maximum number of iterations during one run is set to 5000. Section III-B will introduce the five optimization algorithms under evaluation.

B. ALGORITHMS UNDER COMPARISON

Our study is to explore the effectiveness of several existing evolutionary optimization algorithms. Therefore, we briefly introduce the algorithms under evaluation: NSGA-II, DBEA, NSGA-III, eMOEA, and IBEA. We selected these five algorithms because these algorithms are widely used. These algorithms can be roughly divided into five categories,

[4]MOEA framework, http://moeaframework.org/.
[5]jMetal, http://jmetal.sourceforge.net/.

i.e., optimization via genetic algorithms, decomposition, reference points, ϵ-dominance, and the indicator, respectively [11], [15].

NSGA-II [10] is a well-known evolutionary algorithm for multi-objective optimization. When facing two objectives, NSGA-II can easily detect the *dominance* between two solutions, i.e., the relationship that a solution is better than another. This makes NSGA-II robust for most application scenarios of multi-objective optimization.

DBEA [24] is a typical decomposition-based evolutionary algorithm that is designed for many-objective optimization. DBEA generates uniformly distributed reference points via sampling techniques and tends to balance the convergence and the diversity. A *reference point* denotes a known best solution under a particular objective.

NSGA-III [15] is an upgraded version of NSGA-II and targets the difficulty of ranking solutions under the scenario of many-objective optimization. The fundamental difference between NSGA-III and its previous version NSGA-II is the niche preservation operation [11]. NSGA-III starts its search from a set of reference points. This makes NSGA-III outperform NSGA-II in many applications, but fail in the robustness.

eMOEA [25] is an ϵ-dominance-based algorithm for multi-objective optimization. The ϵ value is a small constant. The kernel idea of eMOEA is to divide the objective space into hyper-boxes with a given size ϵ and cluster nondominated solutions to each box.

IBEA [26] is an indicator-based evolutionary algorithm. This algorithm combines arbitrary indicators to adapt to the preference without requiring any additional diversity preservation mechanism during calculation.

C. EVALUATION METRICS

We evaluated the algorithms via six metrics. Table 2 briefly describes the metrics in the evaluation for the sake of space. In the description, the *approximation set* and the *reference set* represents two sets of solutions: one approximates all obtained solutions and the other denotes the defined best solutions or the setup. We also list the preferred values of each metric. For instance, a high value of HV denotes a better Pareto space while a low value of Epsilon denotes a close distance between obtained solutions and optimal solutions. Details of calculating these metrics can be found in [10], [11], [15], [27], and [28]. Among the metrics, the hypervolume is used as the major metric in existing studies.

D. RESEARCH QUESTIONS

In this exploratory study, we focus on four Research Questions (RQs), including the effectiveness of optimization, the significance of results, the distribution of metric values, and the correlation between metrics, respectively.

RQ1: How effective can existing optimization algorithms perform on the ManyNRP?

Effectiveness is the most important goal of evaluating optimization. In RQ1, we compare the results by five evolutionary optimization methods via evaluating six metrics. A method that reaches the best results is the first choice in practice.

RQ2: Is the comparison among algorithms statistically significant?

Statistical significance can aid the evaluation of results. In particular, significant results can be used to identify whether two methods can be distinguished by checking their results. RQ2 is designed to check the significance between metric values.

RQ3: How do metric values distribute on benchmark instances?

In RQ3, we examine the distribution of metrics values on 12 benchmark instances. Such distribution can help understand the results of particular algorithms as well as instances.

RQ4: Is there any correlation between metric values?

We speculate that there may exist correlations between metric values. Therefore, such a correlation can reduce the duplicate evaluation in under limited resources. The aim of RQ4 is to examine the existence of the correlation between metric values.

IV. RESULTS

We conducted an exploratory study via answering four RQs.

A. RQ1. HOW EFFECTIVE CAN EXISTING OPTIMIZATION ALGORITHMS PERFORM ON THE ManyNRP?

Tables 3, 4, and 5 present the comparison results of six metrics by five optimization algorithms on 12 benchmark instances from three projects, Eclipse, Gnome, and Mozilla. Each table

TABLE 2. Evaluation metrics for many-objective optimization.

Abbr.	Metric	Preferred	Description
HV	Hypervolume	High	Volume of objective space, which is dominated by solutions in the approximation set
Epsilon	Epsilon	Low	Minimum value of ϵ for the approximation set to the ϵ-dominate reference set
IGD	Inverted generational distance	Low	Average distance from solutions in the reference set to the nearest solution in an approximation set
Spacing	Spacing	High	Spread space of the Pareto front
GD	Generational distance	Low	Average distance from solutions in an approximation set to the nearest solution in the reference set
MPFE	Maximum Pareto front error	Low	Maximum distance from solutions in an approximation set to the nearest solution in the reference set

TABLE 3. Comparison of six metrics by five optimization algorithms on four instances from Eclipse.

Instance	Metric	NSGA-II		DBEA		NSGA-III		eMOEA		IBEA	
		Average	St.dev.	Average	St.dev.	Average	St.dev.	Average	St.dev.	Average	St.dev.
nrp-e1	HV	0.0636	0.0105	0.0479	0.0056	0.0431	0.0037	0.1796	0.0197	**0.2434**	0.0220
	Epsilon	0.4184	0.0323	0.4703	0.0196	0.4952	0.0181	**0.2877**	0.0240	0.4504	0.0735
	IGD	0.4129	0.0367	0.4886	0.0245	0.5066	0.0222	**0.2364**	0.0209	0.2880	0.0142
	Spacing	35.3915	4.4081	32.2177	6.3386	28.0043	5.0479	**37.1086**	11.2747	23.0579	4.1539
	GD	0.0176	0.0020	0.0205	0.0015	0.0151	0.0008	0.0102	0.0015	**0.0057**	0.0013
	MPFE	0.3666	0.0530	0.3881	0.0459	0.4236	0.0346	0.2379	0.0473	**0.1096**	0.0178
nrp-e2	HV	0.0495	0.0098	0.0300	0.0035	0.0317	0.0039	0.1895	0.0238	**0.2376**	0.0260
	Epsilon	0.4295	0.0353	0.5135	0.0213	0.5026	0.0269	**0.3317**	0.0478	0.4756	0.0657
	IGD	0.4664	0.0408	0.5870	0.0286	0.5688	0.0303	**0.2104**	0.0250	0.2314	0.0116
	Spacing	28.8525	4.1599	27.2492	7.0230	23.8403	4.6388	**40.7649**	19.2485	19.5849	2.9671
	GD	0.0225	0.0013	0.0283	0.0018	0.0216	0.0016	0.0115	0.0024	**0.0069**	0.0016
	MPFE	0.4271	0.0487	0.4602	0.0358	0.4958	0.0376	0.2239	0.0375	**0.1342**	0.0236
nrp-e3	HV	0.0617	0.0119	0.0428	0.0049	0.0407	0.0047	0.1718	0.0195	**0.2456**	0.0169
	Epsilon	0.4374	0.0476	0.4996	0.0293	0.5248	0.0292	**0.2869**	0.0355	0.4703	0.0576
	IGD	0.4261	0.0427	0.5178	0.0295	0.5284	0.0308	**0.2356**	0.0215	0.2569	0.0104
	Spacing	31.8915	3.8521	28.0487	4.7311	25.4828	3.3923	**34.8478**	9.8551	22.0446	3.8164
	GD	0.0188	0.0019	0.0228	0.0016	0.0170	0.0009	0.0111	0.0018	**0.0052**	0.0014
	MPFE	0.4118	0.0755	0.4294	0.0457	0.4711	0.0374	0.2599	0.0306	**0.1027**	0.0243
nrp-e4	HV	0.0594	0.0109	0.0353	0.0040	0.0388	0.0038	0.1926	0.0259	**0.2436**	0.0244
	Epsilon	0.4102	0.0324	0.4988	0.0240	0.4804	0.0197	**0.3107**	0.0598	0.4419	0.0577
	IGD	0.4458	0.0409	0.5760	0.0276	0.5585	0.0288	**0.2173**	0.0262	0.2202	0.0126
	Spacing	26.2581	3.7315	23.9731	4.5078	21.4080	4.5883	**30.9555**	16.9687	18.1514	4.9312
	GD	0.0226	0.0015	0.0285	0.0017	0.0212	0.0018	0.0122	0.0026	**0.0065**	0.0017
	MPFE	0.4186	0.0449	0.4596	0.0427	0.4854	0.0317	0.2415	0.0440	**0.1236**	0.0203

TABLE 4. Comparison of six metrics by five optimization algorithms on four instances from Gnome.

Instance	Metric	NSGA-II		DBEA		NSGA-III		eMOEA		IBEA	
		Average	St.dev.	Average	St.dev.	Average	St.dev.	Average	St.dev.	Average	St.dev.
nrp-g1	HV	0.0718	0.0091	0.0573	0.0061	0.0537	0.0043	0.1597	0.0164	**0.2369**	0.0154
	Epsilon	0.3854	0.0391	0.4414	0.0237	0.4599	0.0224	**0.3074**	0.0326	0.5037	0.0548
	IGD	0.3874	0.0437	0.4549	0.0296	0.4852	0.0219	**0.2275**	0.0203	0.2330	0.0120
	Spacing	**39.1313**	5.2991	32.3945	4.7334	27.0398	3.5062	37.6957	9.3681	23.2689	4.1955
	GD	0.0174	0.0019	0.0185	0.0012	0.0146	0.0007	0.0104	0.0013	**0.0045**	0.0013
	MPFE	0.3661	0.0645	0.3765	0.0358	0.4211	0.0353	0.2686	0.0266	**0.0925**	0.0205
nrp-g2	HV	0.0611	0.0140	0.0422	0.0049	0.0429	0.0035	0.1661	0.0174	**0.2289**	0.0220
	Epsilon	0.4201	0.0367	0.4981	0.0291	0.4781	0.0223	**0.3667**	0.0703	0.5303	0.0742
	IGD	0.4425	0.0483	0.5281	0.0300	0.5199	0.0193	0.2229	0.0214	**0.2195**	0.0113
	Spacing	27.4585	3.3214	24.2837	4.3163	21.2561	3.4877	**32.4265**	11.9229	19.0681	4.1038
	GD	0.0210	0.0022	0.0241	0.0014	0.0185	0.0007	0.0119	0.0021	**0.0060**	0.0014
	MPFE	0.3918	0.0584	0.4290	0.0317	0.4665	0.0393	0.2529	0.0305	**0.1165**	0.0199
nrp-g3	HV	0.0519	0.0079	0.0428	0.0044	0.0398	0.0046	0.1388	0.0120	**0.2195**	0.0152
	Epsilon	0.4460	0.0335	0.4866	0.0209	0.5042	0.0242	**0.3378**	0.0387	0.5473	0.0525
	IGD	0.4272	0.0371	0.4833	0.0315	0.5017	0.0263	**0.2378**	0.0202	0.2761	0.0170
	Spacing	37.0275	4.7816	30.7234	5.3785	28.0919	3.1684	**37.9204**	10.7281	24.2615	4.2260
	GD	0.0185	0.0029	0.0173	0.0010	0.0142	0.0008	0.0104	0.0013	**0.0052**	0.0012
	MPFE	0.3833	0.0729	0.3624	0.0391	0.4219	0.0470	0.2775	0.0403	**0.1016**	0.0162
nrp-g4	HV	0.0613	0.0088	0.0425	0.0039	0.0436	0.0045	0.1553	0.0180	**0.2033**	0.0160
	Epsilon	0.3796	0.0356	0.4891	0.0295	0.4687	0.0272	**0.3555**	0.0602	0.5535	0.0507
	IGD	0.3945	0.0412	0.4983	0.0271	0.4912	0.0297	**0.2241**	0.0249	0.2248	0.0074
	Spacing	25.7297	3.6031	25.3051	5.7642	20.1746	2.7866	**30.3883**	8.9940	17.6958	3.5993
	GD	0.0191	0.0018	0.0217	0.0015	0.0165	0.0006	0.0123	0.0019	**0.0048**	0.0011
	MPFE	0.3808	0.0623	0.4083	0.0354	0.4490	0.0407	0.2643	0.0413	**0.1005**	0.0213

consists of four instances; the results are shown with the average of 30 repeats and their standard deviation. Among five optimization algorithms, eMOEA and IBEA perform the best on 12 instances; NSGA-II can obtain several best values when evaluating Spacing; DBEA and NSGA-III never reach the best results during the comparison.

Table 3 shows the evaluation results on instances from Eclipse. Two algorithms eMOEA and IBEA beat all the other three algorithms. For three metrics, HV, GD, and MPFE, IBEA gets the best results of all four instances; for the other three metrics, eMOEA gets the best. As shown in Table 3, the results on four instances are stable.

Table 4 shows the evaluation results on instances from Gnome. Two algorithms eMOEA and IBEA again show the best results on all the metrics but one: NSGA-II obtains the best value when evaluating Spacing. The results from Gnome

TABLE 5. Comparison of six metrics by five optimization algorithms on four instances from Mozilla.

Instance	Metric	NSGA-II		DBEA		NSGA-III		eMOEA		IBEA	
		Average	St.dev.	Average	St.dev.	Average	St.dev.	Average	St.dev.	Average	St.dev.
nrp-m1	HV	0.0715	0.0104	0.0587	0.0047	0.0569	0.0053	0.2024	0.0221	**0.2646**	0.0266
	Epsilon	0.4111	0.0289	0.4653	0.0182	0.4699	0.0196	**0.2979**	0.0419	0.5317	0.0621
	IGD	0.4128	0.0316	0.4907	0.0201	0.4962	0.0255	**0.2486**	0.0215	0.2815	0.0142
	Spacing	**42.5381**	5.0552	37.1650	9.1365	31.7868	3.6763	37.3875	7.9698	26.5020	5.2472
	GD	0.0179	0.0018	0.0219	0.0016	0.0161	0.0011	0.0085	0.0019	**0.0047**	0.0012
	MPFE	0.3500	0.0371	0.3927	0.0335	0.4273	0.0332	0.1964	0.0350	**0.0982**	0.0163
nrp-m2	HV	0.0558	0.0096	0.0385	0.0038	0.0405	0.0034	0.2000	0.0263	**0.2414**	0.0200
	Epsilon	0.4332	0.0322	0.5027	0.0218	0.4917	0.0213	**0.3031**	0.0457	0.4843	0.0683
	IGD	0.4692	0.0416	0.5655	0.0274	0.5591	0.0234	0.2048	0.0263	**0.2017**	0.0113
	Spacing	**32.8741**	4.3874	30.5625	6.3959	26.7131	5.1740	41.1778	16.4846	22.5081	5.3236
	GD	0.0232	0.0019	0.0283	0.0020	0.0206	0.0013	0.0115	0.0026	**0.0056**	0.0014
	MPFE	0.4273	0.0466	0.4805	0.0295	0.4846	0.0304	0.2240	0.0305	**0.1175**	0.0219
nrp-m3	HV	0.0576	0.0080	0.0525	0.0037	0.0472	0.0027	0.1548	0.0167	**0.2185**	0.0142
	Epsilon	0.3991	0.0269	0.4591	0.0295	0.4686	0.0142	**0.3372**	0.0317	0.5837	0.0492
	IGD	0.4007	0.0290	0.4450	0.0245	0.4770	0.0161	**0.2343**	0.0186	0.2885	0.0169
	Spacing	**45.8165**	6.2196	35.7644	5.9564	32.7930	4.7985	41.2407	10.6846	27.9683	6.1419
	GD	0.0188	0.0023	0.0208	0.0017	0.0168	0.0008	0.0086	0.0017	**0.0051**	0.0012
	MPFE	0.3683	0.0536	0.3796	0.0298	0.4482	0.0502	0.2164	0.0267	**0.0967**	0.0162
nrp-m4	HV	0.0554	0.0115	0.0380	0.0045	0.0402	0.0030	0.1828	0.0280	**0.2255**	0.0216
	Epsilon	0.4050	0.0426	0.4942	0.0250	0.4715	0.0184	**0.3623**	0.0434	0.5472	0.0689
	IGD	0.4456	0.0489	0.5492	0.0261	0.5444	0.0257	**0.2068**	0.0290	0.2074	0.0122
	Spacing	30.1297	5.3949	26.4829	4.4128	24.1971	4.6738	**34.4085**	14.0434	20.5137	3.3458
	GD	0.0230	0.0018	0.0269	0.0019	0.0204	0.0014	0.0117	0.0027	**0.0058**	0.0014
	MPFE	0.4038	0.0486	0.4425	0.0232	0.4698	0.0344	0.2392	0.0485	**0.1106**	0.0196

are not as stable as those from Eclipse. For example, in the instance nrp-g2, IBEA can obtain the best values for four metrics; eMOEA loses the best value for the IGD metric. Two algorithms DBEA and NSGA-III again fail in reaching the best results on all the instances.

Table 5 shows the evaluation results on instances from Mozilla. IBEA obtains the best values for most metrics; eMOEA also obtains many best values; NSGA-II reaches the best Spacing values on three instances, nrp-m1, nrp-m2, and nrp-m3. Compared across Tables 3, 4, and 5, results from Mozilla are the most unstable ones.

As shown in the empirical results, NSGA-III is not suitable for solving the ManyNRP although it is designed to break the assumption of multi-objective optimization. This experiment also verifies that several typical algorithms are robust to different scenarios, such as NSGA-II and IBEA. This is to some extent consistent with existing observations [15], [29].

Finding 1. To sum up, our results empirically show that IBEA and eMOEA are two best techniques for solving the ManyNRP. If counting the number of best results of metrics, IBEA wins. If considering different metrics, IBEA is the best choice for evaluating HV, GD, and MPFE; eMOEA is the best choice for evaluating Epsilon, GD, and Spacing.

B. RQ2. IS THE COMPARISON AMONG ALGORITHMS STATISTICALLY SIGNIFICANT?

We evaluated the statistical significance among algorithm results. As mentioned in Section IV-A, Tables 3, 4, and 5 have shown that the difference among algorithms can be directly observed and stable. Therefore, we leveraged the statistical tests to support the above results.

We used the Mann-Whitney U test to calculate the statistical significance. The *Mann-Whitney U* test, also called the Wilcoxon rank-sum test, is employed to determine whether two independent set of samples are selected from the same distribution.

We selected two metrics to compare the significance: HV and Spacing. We chose HV since this metric is the most widely used one in evaluating many-objective optimization; we chose Spacing since NSGA-II can only perform the best for this metric. Tables 6 and 7 show the statistical significance between two algorithms for HV and Spacing, respectively. Only half of the matrixes are shown because the significance is symmetric.

TABLE 6. Significance of the hypervolume metric by the Mann-Whitney U test among five algorithms.

	NSGA-II	DBEA	NSGA-III	eMOEA	IBEA
NSGA-II	–				
DBEA	0	–			
NSGA-III	0	0.4758	–		
eMOEA	0	0	0	–	
IBEA	0	0	0	0	–

TABLE 7. Significance of the spacing metric by the Mann-Whitney U test among five algorithms.

	NSGA-II	DBEA	NSGA-III	eMOEA	IBEA
NSGA-II	–				
DBEA	0	–			
NSGA-III	0	0	–		
eMOEA	0	0	0	–	
IBEA	0	0	0	0	–

As shown in Table 6, most of results between each pair of algorithms are statistical significant. That is, the difference

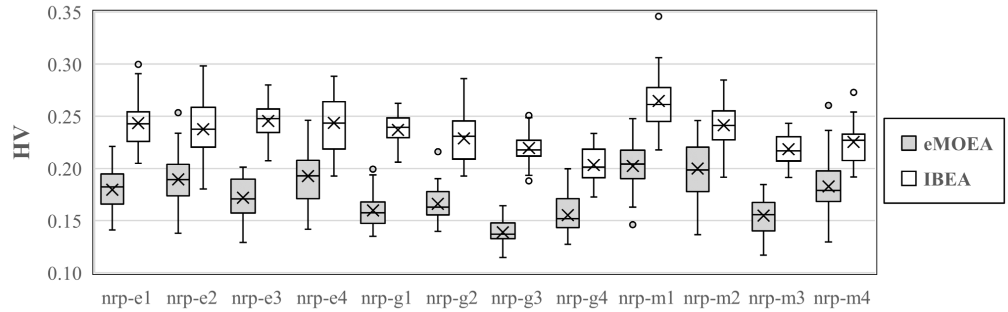


FIGURE 2. Box-plots of the hypervolume by eMOEA and IBEA.

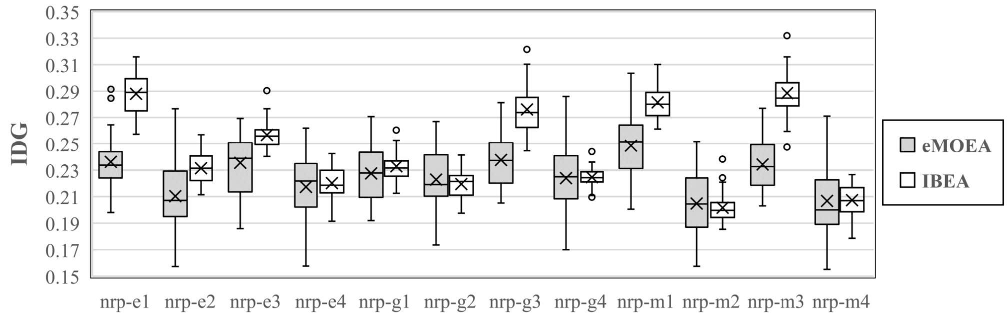


FIGURE 3. Box-plots of the IGD by eMOEA and IBEA.

between two algorithms exist and could be perceived. One exception is the pair of NSGA-III and DBEA. As mentioned in Section IV-A, both algorithms perform not well. It is possible that results by these two algorithms behave similar.

As shown in Table 7, results between each pair are statistical significant. For the Spacing metric, all pairs of results can be distinguished.

Finding 2. After calculating the statistical significance via the Mann-Whitney U test, we found that results by most of pairs of algorithms are statistically significant. That is, these algorithms can be easily distinguished.

C. RQ3. HOW DO METRIC VALUES DISTRIBUTE ON BENCHMARK INSTANCES?

To better understand the distribution of results by many-objective optimization, we leveraged the box-plots to show the quantiles of the evaluation. Figs. 2, 3, and 4 show the box-plots of two algorithms, eMOEA and IBEA, for three metrics, the hypervolume, the IDG, and the Spacing.

As shown in Fig. 2, IBEA always achieves better results than eMOEA. Among 12 instances, the second quantile (the value with the 25% ranking from low to high) of IBEA is higher than the fourth quantile (the value with the 75% ranking from low to high) of eMOEA. This result indicates that the difference of the hypervolume metric values between two algorithms is easy to be identified.

In Fig. 3, the median values by eMOEA are better than those by IBEA on 9 out of 12 instances. In 8 out of 12 instances, the middle 50% values (from 25% to 75% ranking) have an overlap between two algorithms.

As shown in Fig. 4, eMOEA outperforms IBEA. In 11 out of 12 instances, there is no overlap for the middle 50 % values between two algorithms. One exceptional instance is nrp-e4: a tiny overlap exists. Compared across Figs. 2, 3, and 4, the Spacing metric leads to more outliers than the other two metrics.

Finding 3. The distribution of metric values of eMOEA and IBEA seems stable. We observed that IBEA beats eMOEA for the hypervolume metric while eMOEA beats IBEA for the Spacing metric.

D. RQ4. IS THERE ANY CORRELATION AMONG METRIC VALUES?

As shown in Section IV-A, several metric values seem to be correlative. We explored such correlations via the Pearson correlation coefficient. Given two sets of samples Y and Z, where $y_i \in Y$ and $z_i \in Z$ $(1 \leq i \leq q)$, the Pearson correlation coefficient is defined as follows,

$$Pearson(Y, Z) = \frac{\sum_{i=1}^{q}(y_i - \bar{y})(z_i - \bar{z})}{\sqrt{\sum_{i=1}^{q}(y_i - \bar{y})^2}\sqrt{\sum_{i=1}^{q}(z_i - \bar{z})^2}},$$

where $\bar{y}$ and $\bar{z}$ denote the average values of y_i and z_i, respectively. The Pearson correlation coefficient belongs to $[-1, 1]$: a positive value indicates one set of samples increases while the other increases; a negative value indicates one set of samples decreases with the other increases [30]. A high absolute value shows that two sets of samples are highly correlative.

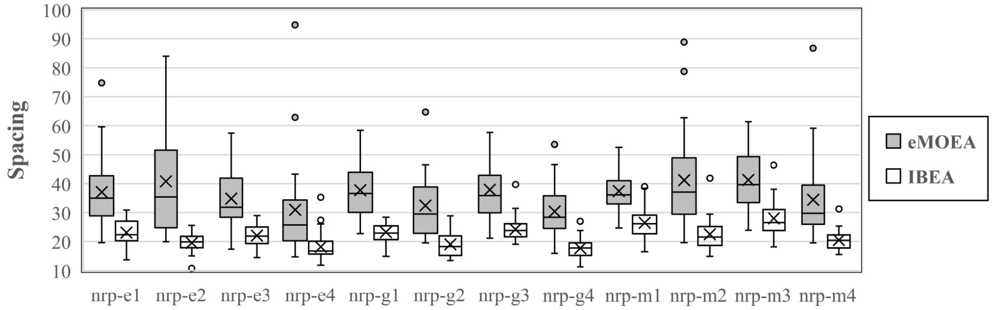


FIGURE 4. Box-plots of the Spacing by eMOEA and IBEA.

In our study, we focus on the absolute values of the Pearson correlation coefficient. We selected two algorithms eMOEA and IBEA; the correlation is calculated on three instances, nrp-e1, nrp-g1, and nrp-m1. Tables 8 and 9 show the Pearson correlation coefficient among metric values. We highlight the absolute values over 0.5000.

TABLE 8. Pearson correlation between metrics for eMOEA.

nrp-e1	HV	Epsilon	IGD	Spacing	GD	MPFE
HV	–					
Epsilon	0.0075	–				
IGD	**-0.5367**	0.1601	–			
Spacing	-0.2238	0.1793	-0.2480	–		
GD	**-0.7574**	0.1533	0.2153	0.2376	–	
MPFE	**-0.7784**	0.2753	**0.5677**	0.1558	**0.6691**	–

nrp-g1	HV	Epsilon	IGD	Spacing	GD	MPFE
HV	–					
Epsilon	0.1246	–				
IGD	**-0.8068**	-0.2268	–			
Spacing	-0.0849	-0.1022	-0.0796	–		
GD	**-0.7090**	0.2804	**0.5390**	0.0924	–	
MPFE	**-0.5747**	-0.3125	**0.6598**	0.2099	0.2701	–

nrp-m1	HV	Epsilon	IGD	Spacing	GD	MPFE
HV	–					
Epsilon	-0.4510	–				
IGD	**-0.7459**	0.1431	–			
Spacing	-0.0918	0.2566	-0.2038	–		
GD	**-0.8040**	**0.5755**	0.4390	0.1496	–	
MPFE	**-0.7000**	0.4510	**0.5268**	0.0058	**0.7700**	–

As the results of eMOEA shown in Table 8, HV, IGD, GD, and MPFE are four metrics that contain high correlations. We can speculate that given limited resources, only evaluating HV, Epsilon, and Spacing could be adequate.

As the results of IBEA shown in Table 9, HV and Epsilon contain high correlations; GD and MPFE contain high correlations. Correlation values on the instance nrp-g1 show that there are many highly correlative metrics.

Finding 4. The result of Pearson correlation coefficient shows that there indeed exist high correlations between metrics, between HV and IGD, between IGD and GD, or between HV and Epsilon. If an experiment is conducted under limited resources, evaluation with several metrics can be omitted.

TABLE 9. Pearson correlation between metrics for IBEA.

nrp-e1	HV	Epsilon	IGD	Spacing	GD	MPFE
HV	–					
Epsilon	**-0.6879**	–				
IGD	-0.2194	**0.5116**	–			
Spacing	0.3632	-0.2698	-0.3044	–		
GD	**-0.6292**	0.0406	-0.2327	-0.1611	–	
MPFE	-0.3710	-0.0527	-0.3055	0.0628	**0.8041**	–

nrp-g1	HV	Epsilon	IGD	Spacing	GD	MPFE
HV	–					
Epsilon	**-0.5847**	–				
IGD	**-0.5049**	**0.7215**	–			
Spacing	0.0249	**-0.5234**	-0.4474	–		
GD	-0.4351	-0.2123	-0.4088	**0.5576**	–	
MPFE	-0.1725	-0.4117	**-0.5222**	**0.5382**	**0.8335**	–

nrp-m1	HV	Epsilon	IGD	Spacing	GD	MPFE
HV	–					
Epsilon	**-0.8277**	–				
IGD	-0.4437	**0.6269**	–			
Spacing	0.3501	-0.4590	-0.3014	–		
GD	**-0.7297**	0.3328	-0.0174	-0.0098	–	
MPFE	-0.4795	0.1901	-0.1550	0.1178	**0.7742**	–

V. THREATS TO VALIDITY

We present the threats to the validity of our study in four categories.

A. MORE INSTANCES CAN HELP

In our study, 12 instances from three projects are used to evaluate five optimization algorithms. Our study aims to exhibit the evidence on relatively-realistic requirements data. Using these 12 instances is a common way for the evaluation on real-world data of requirements selection [8], [13], [17]–[20]. As mentioned in Section III-A, requirements data is private and important to software companies, especially commercial companies. These 12 instances can be viewed as a trade-off between real-world data and feasible experiments.

B. WHY OPTIMIZATION ALGORITHMS BEHAVE DIFFERENT

This study contains five optimization algorithms, including NSGA-II, DBEA, NSGA-III, eMOEA, and IBEA. Experiments in Section IV-A show that several algorithms do not behave as what they are expected. For instance, NSGA-III does not perform well. Similar to results from the community of evolutionary computation, we can speculate that NSGA-III may be not as robust as NSGA-II or IBEA. However, we cannot find any solid evidence to support why IBEA performs much better than NSGA-III. Meanwhile, to the best of our knowledge, most of studies from the community of evolutionary computation also use empirical results to compare the effectiveness of algorithms.

C. HOW MANY OBJECTIVES CAN BE CALLED "MANY"

In Section II-B, we have introduced the difference between multi-objective optimization and many-objective optimization. This does not help determine how many objectives can be called ''many''. This is a challenging problem, even in the community of evolutionary computation [11]. Since the Pareto frontier is hardly obtained for over four objectives, we keep using experiments to evaluate the solutions to the ManyNRP.

VI. RELATED WORK

The task of requirements selection can be conducted via two categories: selecting customers or selecting requirements. Due to the historical reason, the division between these two categories is a little vague. Generally, selecting customers as well as their profits falls into the category of the next release problem [2], [6], [13] while selecting requirements as well as their costs falls into the category of the release planning problem [31], [32]. As mentioned in Section II-C, selecting requirements or customers can be directly transferred into each other. For instance, several existing studies such as Sarro *et al.* [29] have not distinguished these two categories.

The research of the NRP has achieved fruitful results [3], [5], [7], [13], [18], [19]. Zhang *et al.* [22] have formulized the NRP into a multi-objective optimization problem and have solved it with several typical optimization methods. Finkelstein *et al.* [6] have proposed the fairness analysis to aid the decision of the NRP. Harman *et al.* [2] have designed an exact and scalable analysis for the sensitivity of the multi-objective NRP. This work provides detailed comparison and understanding for the application scenario of the NRP. Ren *et al.* [8] conducted a feature-based analysis to explore the hardness of solving the NRP instances.

Different from the above work, this paper aims to conduct an explorative study on the effectiveness of optimization algorithms on the NRP with five objectives. This problem falls into the category of many-objective optimization in requirements selection, which has not been studied before. We evaluated five optimization algorithms with six metrics on 12 benchmark instances.

Requirements selection has attracted much attention since this problem is practical in daily software design and development. Fricker and Schumacher [31] conducted an industrial study for the release planning with tree-structured features. Li *et al.* [33] refined the requirements that are requested by customers to formal methods. Hierons *et al.* [34] have examined and optimized software product line via feature models.

Optimization is a key topic of search-based software engineering. Zheng *et al.* [35] proposed the paradigm of multi-objective optimization for regression testing. Ferrucci *et al.* [36] studied the multi-objective optimization for software planning for overtime requirements. Ren *et al.* [37] developed the co-evolutionary technique for project staff assignments and job scheduling. Mkaouer *et al.* [38] have studied the many-objective optimization for software refactoring via multiple quality attributes. Xuan *et al.* [39] and Chi *et al.* [40] designed optimization methods to enhance the existing test suites to improve the efficiency of test execution. Recently, Chen *et al.* [41] proposed a new paradigm of optimization algorithms to replace the role of evolutionary computing techniques in search-based software engineering.

VII. CONCLUSIONS

We conducted an exploratory study on the effectiveness of finding solutions to the ManyNRP. This study has examined the results of five algorithms on 12 NRP instances via evaluating six metrics. Experimental results show that IBEA is the best among optimization algorithms under comparison. Our study suggests a general way to study the ManyNRP and provides a preliminary analysis for similar problems.

Our future work is to design a new algorithm for many-objective optimization by analyzing which factor impacts IBEA (or eMOEA) most. We also plan to propose other objectives for the practice of the NRP.

REFERENCES

[1] S. Barney, A. Aurum, and C. Wohlin, ''A product management challenge: Creating software product value through requirements selection,'' *J. Syst. Archit.*, vol. 54, no. 6, pp. 576–593, 2008, doi: 10.1016/j.sysarc.2007.12.004.

[2] M. Harman, J. Krinke, I. Medina-Bulo, F. Palomo-Lozano, J. Ren, and S. Yoo, ''Exact scalable sensitivity analysis for the next release problem,'' *ACM Trans. Softw. Eng. Methodol.*, vol. 23, no. 2, pp. 19:1–19:31, 2014, doi: 10.1145/2537853.

[3] A. J. Bagnall, V. J. Rayward-Smith, and I. M. Whittley, ''The next release problem,'' *Inf. Softw. Technol.*, vol. 43, no. 14, pp. 883–890, 2001, doi: 10.1016/S0950-5849(01)00194-X.

[4] H. Jiang, J. Xuan, and Z. Ren, ''Approximate backbone based multilevel algorithm for next release problem,'' in *Proc. Genet. Evol. Comput. Conf. (GECCO)*, Portland, OR, USA, Jul. 2010, pp. 1333–1340, doi: 10.1145/1830483.1830730.

[5] J. Ren, ''Sensitivity analysis in multi-objective next release problem and fairness analysis in software requirements engineering,'' M.S. thesis, Dept. Comput. Sci., King's College London, London, U.K., 2007.

[6] A. Finkelstein, M. Harman, S. A. Mansouri, J. Ren, and Y. Zhang, ''A search based approach to fairness analysis in requirement assignments to aid negotiation, mediation and decision making,'' *Requirements Eng.*, vol. 14, no. 4, pp. 231–245, 2009, doi: 10.1007/s00766-009-0075-y.

[7] A. A. Araújo, M. Paixão, I. Yeltsin, A. Dantas, and J. Souza, ''An architecture based on interactive optimization and machine learning applied to the next release problem,'' *Autom. Softw. Eng.*, vol. 24, no. 3, pp. 623–671, 2017, doi: 10.1007/s10515-016-0200-3.

[8] Z. Ren, H. Jiang, J. Xuan, S. Zhang, and Z. Luo, ''Feature based problem hardness understanding for requirements engineering,'' *Sci. China Inf. Sci.*, vol. 60, no. 3, p. 32105, 2017, doi: 10.1007/s11432-016-0089-7.

[9] J. J. Durillo, Y. Zhang, E. Alba, M. Harman, and A. J. Nebro, ''A study of the bi-objective next release problem,'' *Empirical Softw. Eng.*, vol. 16, no. 1, pp. 29–60, 2011, doi: 10.1007/s10664-010-9147-3.

[10] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002, doi: 10.1109/4235.996017.

[11] H. Seada and K. Deb, "A unified evolutionary optimization procedure for single, multiple, and many objectives," *IEEE Trans. Evol. Comput.*, vol. 20, no. 3, pp. 358–369, Jun. 2016.

[12] E. Park, J. H. Kim, H. S. Nam, and H.-J. Chang, "Requirement analysis and implementation of smart emergency medical services," *IEEE Access*, vol. 6, pp. 42022–42029, 2018, doi: 10.1109/ACCESS.2018.2861711.

[13] J. Xuan, H. Jiang, Z. Ren, and Z. Luo, "Solving the large scale next release problem with a backbone-based multilevel algorithm," *IEEE Trans. Softw. Eng.*, vol. 38, no. 5, pp. 1195–1212, Sep. 2012, doi: 10.1109/TSE.2011.92.

[14] A. Abbas, I. F. Siddiqui, S. U. Lee, A. K. Bashir, W. Ejaz, and N. M. F. Qureshi, "Multi-objective optimum solutions for IoT-based feature models of software product line," *IEEE Access*, vol. 6, pp. 12228–12239, 2018, doi: 10.1109/ACCESS.2018.2806944.

[15] 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 Trans. Evol. Comput.*, vol. 18, no. 4, pp. 577–601, Apr. 2013.

[16] A. L. Jaimes, C. A. Coello, and J. E. U. Barrientos, "Online objective reduction to deal with many-objective problems," in *Proc. 5th Int. Conf. Evol. Multi-Criterion Optim. (EMO)*. Berlin, Germany: Springer-Verlag, 2009, pp. 423–437, doi: 10.1007/978-3-642-01020-0_34.

[17] M. O. de Barros, "An experimental evaluation of the importance of randomness in hill climbing searches applied to software engineering problems," *Empirical Softw. Eng.*, vol. 19, no. 5, pp. 1423–1465, Oct. 2014, doi: 10.1007/s10664-013-9294-4.

[18] R. Fuchshuber and M. de Oliveira Barros, "Improving heuristics for the next release problem through landscape visualization," in *Search-Based Software Engineering*, C. Le Goues and S. Yoo, Eds. Cham, Switzerland: Springer, 2014, pp. 222–227.

[19] M. Paixao and J. Souza, "A robust optimization approach to the next release problem in the presence of uncertainties," *J. Syst. Softw.*, vol. 103, pp. 281–295, May 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S016412121400212X

[20] N. Veerapen, G. Ochoa, M. Harman, and E. K. Burke, "An integer linear programming approach to the single and bi-objective next release problem," *Inf. Softw. Technol.*, vol. 65, pp. 1–13, Sep. 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0950584915000658

[21] J. Xuan *et al.*, "Nopol: Automatic repair of conditional statement bugs in java programs," *IEEE Trans. Softw. Eng.*, vol. 43, no. 1, pp. 34–55, Jan. 2017, doi: 10.1109/TSE.2016.2560811.

[22] Y. Zhang, M. Harman, and S. A. Mansouri, "The multi-objective next release problem," in *Proc. Genet. Evol. Comput. Conf. (GECCO)*, London, U.K., Jul. 2007, pp. 1129–1137, doi: 10.1145/1276958.1277179.

[23] J. J. Durillo and A. J. Nebro, "jMetal: A Java framework for multi-objective optimization," *Adv. Eng. Softw.*, vol. 42, pp. 760–771, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0965997811001219

[24] M. Asafuddoula, T. Ray, and R. A. Sarker, "A decomposition-based evolutionary algorithm for many objective optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 3, pp. 445–460, Jun. 2015, doi: 10.1109/TEVC.2014.2339823.

[25] K. Deb, M. Mohan, and S. Mishra, "A fast multi-objective evolutionary algorithm for finding well-spread Pareto-optimal solutions," Indian Inst. Technol., Kanpur, India, KanGAL Rep. 2003002, 2003, pp. 1–18.

[26] E. Zitzler and S. Künzli, "Indicator-based selection in multiobjective search," in *Proc. 8th Int. Conf. Parallel Problem Solving Nature (PPSN)*, Birmingham, U.K., Sep. 2004, pp. 832–842, doi: 10.1007/978-3-540-30217-9_84.

[27] D. Brockhoff, T. Friedrich, and F. Neumann, "Analyzing hypervolume indicator based algorithms," in *Proc. 10th Int. Conf. Parallel Problem Solving Nature (PPSN)*, Dortmund, Germany, Sep. 2008, pp. 651–660, doi: 10.1007/978-3-540-87700-4_65.

[28] Y. Yang, K. Wei, R. Kang, and S. Wang, "Multi-objective chance constrained programming of spare parts based on uncertainty theory," *IEEE Access*, vol. 6, pp. 50049–50054, 2018.

[29] F. Sarro, F. Ferrucci, M. Harman, A. Manna, and J. Ren, "Adaptive multi-objective evolutionary algorithms for overtime planning in software projects," *IEEE Trans. Softw. Eng.*, vol. 43, no. 10, pp. 898–917, Oct. 2017, doi: 10.1109/TSE.2017.2650914.

[30] J. Xuan, H. Jiang, H. Zhang, and Z. Ren, "Developer recommendation on bug commenting: A ranking approach for the developer crowd," *Sci. China Inf. Sci.*, vol. 60, no. 7, pp. 072105-1–072105-18, 2017, doi: 10.1007/s11432-015-0582-8.

[31] S. Fricker and S. Schumacher, "Release planning with feature trees: Industrial case," in *Proc. 18th Int. Working Conf. Requirements Eng., Found. Softw. Quality (REFSQ)*, Essen, Germany, Mar. 2012, pp. 288–305, doi: 10.1007/978-3-642-28714-5_26.

[32] M. Felderer, A. Beer, J. Ho, and G. Ruhe, "Industrial evaluation of the impact of quality-driven release planning," in *Proc. 8th ACM/IEEE Int. Symp. Empirical Softw. Eng. Meas. (ESEM)*, Sep. 2014, pp. 62:1–62:8, doi: 10.1145/2652524.2652579.

[33] F.-L. Li, J. Horkoff, A. Borgida, G. Guizzardi, L. Liu, and J. Mylopoulos, "From stakeholder requirements to formal specifications through refinement," in *Requirements Engineering: Foundation for Software Quality*, S. A. Fricker and K. Schneider, Eds. Cham, Switzerland: Springer, 2015, pp. 164–180.

[34] R. M. Hierons, M. Li, X. Liu, S. Segura, and W. Zheng, "SIP: Optimal product selection from feature models using many-objective evolutionary optimization," *ACM Trans. Softw. Eng. Methodol.*, vol. 25, no. 2, pp. 17:1–17:39, 2016, doi: 10.1145/2897760.

[35] W. Zheng, R. M. Hierons, M. Li, X. Liu, and V. Vinciotti, "Multi-objective optimisation for regression testing," *Inf. Sci.*, vols. 334–335, pp. 1–16, Mar. 2016, doi: 10.1016/j.ins.2015.11.027.

[36] F. Ferrucci, M. Harman, J. Ren, and F. Sarro, "Not going to take this anymore: Multi-objective overtime planning for software engineering projects," in *Proc. 35th Int. Conf. Softw. Eng. (ICSE)*, San Francisco, CA, USA, May 2013, pp. 462–471, doi: 10.1109/ICSE.2013.6606592.

[37] J. Ren, M. Harman, and M. Di Penta, "Cooperative co-evolutionary optimization of software project staff assignments and job scheduling," in *Proc. 3rd Int. Symp. Search Based Softw. Eng. (SSBSE)*, Szeged, Hungary, Sep. 2011, pp. 127–141, doi: 10.1007/978-3-642-23716-4_14.

[38] M. W. Mkaouer, M. Kessentini, S. Bechikh, M. Ó. Cinnéide, and K. Deb, "On the use of many quality attributes for software refactoring: A many-objective search-based software engineering approach," *Empirical Softw. Eng.*, vol. 21, no. 6, pp. 2503–2545, 2016, doi: 10.1007/s10664-015-9414-4.

[39] J. Xuan, B. Cornu, M. Martinez, B. Baudry, L. Seinturier, and M. Monperrus, "B-Refactoring: Automatic test code refactoring to improve dynamic analysis," *Inf. Softw. Technol.*, vol. 76, pp. 65–80, Aug. 2016, doi: 10.1016/j.infsof.2016.04.016.

[40] Z. Chi, J. Xuan, Z. Ren, X. Xie, and H. Guo, "Multi-level random walk for software test suite reduction," *IEEE Comput. Intell. Mag.*, vol. 12, no. 2, pp. 24–33, May 2017, doi: 10.1109/MCI.2017.2670460.

[41] J. Chen, V. Nair, and T. Menzies, "Beyond evolutionary algorithms for search-based software engineering," *Inf. Softw. Technol.*, vol. 95, pp. 281–294, Mar. 2018, doi: 10.1016/j.infsof.2017.08.007.

JIANGYI GENG received the M.S. degree from the International School of Software, Wuhan University, China, in 2011, where he is currently pursuing the Ph.D. degree in computer software and theory with the School of Computer Science.

From 2014 to 2017, he was a Research Assistant with Wuhan University.

His current research interests include search-based software engineering, requirement engineering, and mining software repositories.

SHI YING received the Ph.D. degree in computer software and theory from Wuhan University, China, in 1999. He is currently a Professor with the School of Computer Science, Wuhan University.

His current research interests include search-based software engineering, mining software repositories, machine learning-based software engineering, and cloud computing. He has published over 60 papers in international journals and conferences, including ICSE, TSE, and ISSTA.

Dr. Ying was selected by the Program of the Science Foundation for Outstanding Scholarship of Hubei Province. He is a member of the China Computer Federation.

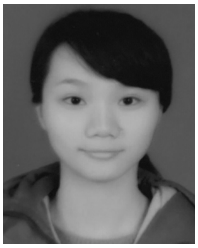
XIANGYANG JIA received the Ph.D. degree in computer software and theory from Wuhan University, China, in 2008. From 2014 to 2015, he was a Visiting Researcher with the Dependable Evolvable Pervasive Software Engineering Group, Politecnico di Milano. He is currently a Lecturer with the School of Computer Science, Wuhan University.

He has published over 30 papers in international journals and conferences. His current research interests include symbolic execution, software analysis, search-based software engineering, and mining software repositories.

Dr. Jia is a member of the China Computer Federation. He received the Hubei Science and Technology Progress Award in 2014.

TING ZHANG received the B.S. degree in computer science and the M.S. degree in communication and information technology from the University of Wollongong, Australia, in 2005 and 2007, respectively. She is currently pursuing the Ph.D. degree in computer software and theory with the School of Computer Science, Wuhan University.

Her current research interests include search-based software engineering and requirement engineering.



XUAN LIU is currently pursuing the bachelor's degree in software engineering with Wuhan University, China.

Her research interests include machine learning, data mining, and automated testing.

LANQING GUO is currently pursuing the bachelor's degree with the School of Computer Science, Wuhan University, China.

Her research interests include machine learning and search-based software engineering.

JIFENG XUAN (M'17) received the B.S. and Ph.D. degrees from the Dalian University of Technology, China.

He was a Post-Doctoral Researcher at the INRIA Lille–Nord Europe, France. He is currently a Professor with the School of Computer Science, Wuhan University, China. He has published over 40 papers in international journals and conferences, including TSE, TKDE, ICSE, and FSE. His research interests include software testing and debugging, software data analysis, and search-based software engineering. He is a reviewer of many journals and conferences, including TSE, TOSEM, ICSE, and FSE.

Dr. Xuan is a member of the ACM and the China Computer Federation (CCF). He was a recipient of the ACM SIGSOFT Distinguished Paper Award and the Outstanding Doctoral Dissertation Award by the CCF. He was elected by the Young Talent Development Program by the China Association for Science and Technology.