Search Based Requirements Optimisation: Existing Work & Challenges

Yuanyuan Zhang [†], Anthony Finkelstein [‡], and Mark Harman [†]

[†] King's College London
Strand, London
WC2R 2LS, UK

[‡]University College London
Malet Place, London
WC1E 6BT, UK

Abstract. In this position paper, we argue that search based software engineering techniques can be applied to the optimisation problem during the requirements analysis phase. Search based techniques offer significant advantages; they can be used to seek robust, scalable solutions, to perform sensitivity analysis, to yield insight and provide feedback explaining choices to the decision maker. This position paper overviews existing achievements and sets out future challenges.

1 Introduction

Once an initial set of requirements has been gathered by requirements elicitation, there is a business-level analysis problem: choices have to be made to identify optimal choices and trade–offs for decision makers. For example, one important goal is to select near optimal subsets from all possible requirements to satisfy the demands of customers, while at the same time making sure that there are sufficient resources to undertake the selected tasks.

To illustrate, Figure 1 demonstrates a possible spread of equally optimal requirements optimisation results. Two competing objectives are considered: cost to the provider and estimated satisfaction rating achieved by a solution. Each circle on the represents an equally optimal solution. That is, each circle denotes a solution for which no better solution (subset of requirements) can be found that offers better customer satisfaction without increasing cost. The set of possible solutions form what is known as a Pareto front. Pareto fronts show a solution space of candidate solutions, from which the decision maker can select. As will be seen later, Pareto fronts also yield insights into the structure of the problem.

This requirement selection problem is one example of the way in which requirements decisions can be formulated as optimisation problems. Other examples include ordering requirements to achieve earliest satisfaction, balancing each customer's needs against the others and balancing tensions between system and user requirements.

Such problems are inherently complex optimisation problems that seek to balance many competing and conflicting concerns, so it would be natural to seek algorithms for decision support. Sadly is often infeasible to apply precise analytic algorithms, because the problems are typically NP hard. To overcome this difficultly, Search Based Software Engineering (SBSE) uses metaheuristic optimisation algorithms that explore and solve complex, multi-objective, highly constrained problems in Software Engineering [5]. This paper argues that Requirements Optimisation can be viewed as an application area for SBSE.

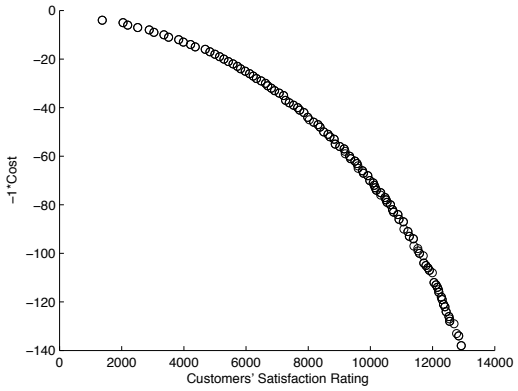

Fig. 1. Fictitious Data: 15 customers; 40 requirements. Adapted from Zhang et al. [13]. Each circle represents an equally optimal candidate solution that balances the objective of minimising supplier cost against the objective of maximising customer satisfaction. See Figure 2 for a comparison to real world requirements data from Motorola.

2 Background: Requirements Optimisation

Previous work on Requirements Optimisation has shown that metaheuristic optimisation techniques can be used to search for a balance between costs and benefits associated with sets of requirements. This has come to be known as the 'Next Release Problem (NRP)' [2]. In the NRP, as formulated by Bagnall et al., the goal is to find the ideal set of requirements that balance customer requests within resource constraints.

In this formulation the problem is a constrained single objective optimisation problem. Bagnall et al. applied a variety of techniques to a set of synthetic data to demonstrate the feasibility of SBSE for this problem. Greer and Ruhe also studied the NRP [4], proposing an iterative Genetic Algorithm and presenting results for real world requirements problems. Their approach balances the resources required for all reeases; assessing and optimizing the extent to which the ordering conflicts with stakeholder priorities.

More recently, there has been work on multi objective formulations of the NRP [11, 13]. In the multi-objective formulation, each of the objectives to be optimised is treated as a separate goal in its own right; the multiple objectives are not combined into a single (weighted) objective function. This allows the optimisation algorithm to explore the Pareto front of non-dominated solutions. Each of these non-dominated solutions denotes a possible assignment of requirements that maximizes all objectives without compromising on the maximization of the others.

Zhang et al. [13] considered the twin objectives of cost to the provider and estimated satisfaction rating for the customer, while Ruhe and Omolade [11] considered the two objectives that balance the tension between user-level and system-level requirements.

3 Advantages of the Search Based Approach

This section describes some of the ways in which SBSE techniques have proved to be effective in Requirements Optimisation and closely related problems.

Robustness. Software engineering problems are typically 'messy' problems in which the available information is often incomplete, sometimes vague and almost always subject to a high degree of change (including unforeseen change). Requirements change frequently, and small changes in the initial stages often lead to large changes to the solutions, affecting the solution complexity and making the results of these initial stages potentially fragile.

An important contribution of SBSE techniques is the way in which they can take changing factors and constraints into account in solution construction. They can, for example, provide near optimal solutions in the search space which remain near-optimal under change, rather than seeking optimal but fragile solutions [7]. This better matches the reality of most software projects, in which robustness under change is often as valuable as any other objective.

Sensitivity Analysis. In SBSE, human effort is partly replaced by meta-heuristic search. Nevertheless, the numerical data upon which the automated part of the process depends come from expert domain knowledge. In the case of requirements engineering, the decision maker is forced to rely on estimates of these crucial inputs to the requirements optimisation process. Sensitivity analysis helps the developer build confidence in the model by studying the uncertainties that are often associated with parameters in models. It aims to identify how 'sensitive' the model is to change. This allows the decision maker to pay additional attention to estimates for which the model is particularly sensitive.

Insight. Requirements Optimisation problem instances have structure. That is, the data have implicit characteristics that the decision maker needs to expose in order to inform decision making. For any non-trivial problem, however, the number of requirements, customers, their interactions and dependencies make these implicit properties far from obvious. No human could be expected to simply look and see all the implications and important features of a problem instance. For example, the search may make it easier to see that satisfaction of one customer tends to lead to dissatisfaction of another or that requirement R_i is always in generated solutions in which R_j is present.

In order to show how SBSE can yield insight in Requirements Optimisation, we now apply the cost-satisfaction formulation of Zhang et al. [13] to real requirement data from Motorola. The results are shown in Figure 2. These results have been anonymised to prevent disclosure of sensitive information.

Compare the real world results of Figure 2 with the smooth Pareto front in Figure 1. There is an 'elbow point' in Figure 2's Pareto front which reveals a potential for optimisation: The customers' satisfaction can be increased from 200 to approximately 1,200 at cost 2,000. This would be more attractive that the increase in satisfaction from 1,200 to 1,500, which would cost almost 3 times as much. The search has revealed a very attractive elbow point at cost 1,200. This kind of insight is very hard to achieve without automated optimisation, like that provided by such a search based approach.

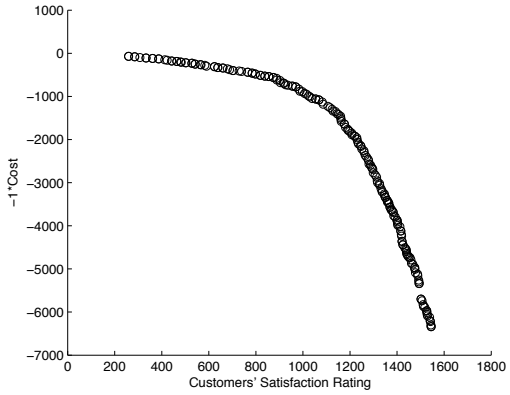

Fig. 2. Motorola mobile device requirements: 4 customers; 35 requirements. The optimisation produces a Pareto front of candidate solutions which reveal an important elbow point at cost 1,200.

Requirements Prioritisation. In the NRP, the decision maker not only selects the optimal or near optimal subset of requirements, but also the priority ordering of requirements. This method offers the potential for risk reduction. That is, circumstances vary and resources may become insufficient to fulfill all requirements. Prioritisation ensures that the most important requirements will be released first so that the maximum attainable benefits of the new system are gained earliest.

Fairness in Requirements Assignment. In the Requirements Optimisation process, it may be helpful to explore the extent to which the obtainable solutions can be said to be fair. Of course, fairness can come in different forms: should we spend the same amount on each customer or give each the same number or value of fulfilled requirements? Each notion of fairness can also be treated as a separate objective, for which a Pareto optimal search seeks non-dominated solutions [10]. In this way it becomes possible to automatically investigate the extent to which several notions of fairness can simultaneously be satisfied.

4 Challenges

This section describes some of the challenges for Search Based Requirements Optimisation.

Scalability. In Requirements Optimisation, problems arise not merely because of the number of requirements, customers and other participating factors, but also because of complexity arising from constraints and dependencies.

Currently, the Requirements Optimisation process, where it is practiced at all, is a highly labour-intensive activity. Search based techniques have the potential to handle large scale problems because they are naturally parallelisable [3, 12]. However, despite this potential, there remains a need for more work on scalability of Search Based Requirements Optimisation.

Solution Representation. Visualisation plays an important role in all optimisation problems [9]. It illustrates the solution quality and help the decision maker to understand the results. This can be easily and directly achieved using scatterplots when there are only 2 or 3 objective dimensions. Visualisation of higher dimensionality remains an open problem in the visualisation community. Requirements Optimisation solutions need to be presented in a manner that is equally intuitive to engineers and to users alike. This represents an additional degree of challenge. There are several visualisation methods for higher dimensional spaces that may be useful, for example Heatmaps [9], Self Organizing Maps (SOM) [8], and Distance and Distribution Charts [1]. However, these remain to be evaluated for Requirements Optimisation.

Feedback and Explanation of Results. In Requirements Optimisation, an additional problem arises when solutions are found: how do developers explain the solution to the customer? Of course, the customer expects to get the highest interest from the solution and they are likely to want to know, not merely the results, but also why a certain set of features was chosen or why some excluded requirements were those for which they had a particular care.

Feedback to the customer should form a part of the solution obtained by the optimisation process. This will maximize each customers' satisfaction and make explicit their participation in the optimisation process. In some cases involving politically sensitive choices, solution justification and explanation may be as important as the solution itself.

Fitness Function Definition. At the heart of SBSE is the fitness function, which guides the search by capturing the properties that make one solution preferable to another. In software engineering applications, fitness functions can be thought of as metrics [6]. These metrics translate constraints such as quality constraints (usability, reliability), organizational constraints (scalability), and environmental constraints (security, privacy) into some measurable attribute of a candidate solution.

Unfortunately, these constraints, often misleadingly termed non-functional requirements, may not be defined precisely in the early stages of the software life cycle. Therefore, techniques are required for iterative update of fitness function definition. It is possible that fitness-measure and solution-generation may need to co-evolve as part of an overall Requirements Optimisation process.

Algorithm Selection. Search Based Requirements Optimisation is based on experimental results from empirical studies. There is, however, currently little theoretical understanding as to when, how and why one metaheuristic algorithm works better than another. Once the nature of Search Based Requirements Optimisation is better understood empirically, it will be important to generalise these results and to augment them with theoretical analysis of search landscape characteristics. This will support a more formal and rigorous analysis of potential algorithmic complexity, thereby motivating the choice of algorithm to apply.

Requirements Dependencies. In the requirement analysis process, requirements are seldom independent of each other. There are two major problems related to requirement dependencies: one is how to identify and model them, the

other is the extent to which these dependencies influence and interact with the software systems level. Ruhe and Omolade [11] show how search based optimisation can track dependencies from user requirements into their impact on system components. Though this is promising, more work is required to handle fuzzy incomplete, multi-way, implicit and temporal requirements dependences.

Partial Requirement Fulfillment. Requirements have varying representations: *discrete variable requirements* which are either fulfilled completely or not fulfilled at all and *continuous variable requirements* which can be fulfilled to a certain extent, for example sever response time in web-based or distributed systems. Existing work on Search Based Requirements Optimisation has treated requirements as being entirely discrete. More work is required to extend these results to handle continuous requirements.

References

1. Kiam Heong Ang, Gregory Chong, and Yun Li. Visualization Technique for Analyzing Non-Dominated Set Comparison. In *Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution and Learning (SEAL'02)*, volume 1, page 36.
2. AJ. Bagnall, VJ. Rayward-Smith, and I.M. Whittley. The Next Release Problem. *IEE Proceedings - Software*, 43(14):883–890, Dec 2001.
3. Y. Collette and P. Siarry. *Multiobjective Optimization: Principles and Case Studies*. Springer, 2004.
4. Des Greer and Günther Ruhe. Software release planning: an evolutionary and iterative approach. *Information & Software Technology*, 46(4):243–253, 2004.
5. Mark Harman. The Current State and Future of Search Based Software Engineering. In *29th Int. Conference on Software Engineering (ICSE 2007), Future of Software Engineering (FoSE)*, Minneapolis, USA, 20-26,May 2007.
6. Mark Harman and John Clark. Metrics are fitness functions too. In 10^{th} *International Software Metrics Symposium (METRICS 2004)*, pages 58–69, Los Alamitos, California, USA, September 2004. IEEE Computer Society Press.
7. Mark Harman, Stephen Swift, and Kiarash Mahdavi. An empirical study of the robustness of two module clustering fitness functions. In *ACM Genetic and Evolutionary Computation Conference (GECCO 2005)*, Washington, D.C., USA, June 25-29 2005.
8. S. Obayashi and D. Sasaki. Visualization and data mining of pareto solutions using self-organizing map. In *Evolutionary Multi-Criterion Optimization, Proceedings, Lecture Notes in Computer Science*, pages 822–835, 2003.
9. Andy Pryke, Sanaz Mostaghim, and Alireza Nazemi. Heatmap visualisation of population based multi objective algorithms. Technical Report CSR-06-14, University of Birmingham, School of Computer Science, December 2006.
10. Jian Ren. Sensitivity analysis in multicobjective next release problem and fairness analysis in software requirements engineering. Master's thesis, DCS/PSE, King's College London, London, England, 2007.
11. Moshood Omolade Saliu and Guenther Ruhe. Bi-objective release planning for evolving software systems. In *Proceedings of ESEC/SIGSOFT FSE 2007*, 2007.
12. F. Szidarovsky, M. E. Gershon, and L. Dukstein. *Techniques for multiobjective decision making in systems management*. Elsevier, New York, 1986.
13. Yuanyuan Zhang, Mark Harman, and S. Afshin Mansouri. The multi-objective next release problem. In *GECCO'07: Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1129–1136. ACM Press, 2007.