Making the Case for MORTO: Multi Objective Regression Test Optimization

Mark Harman
University College London, CREST Centre, Malet Place, London, WC1E 6BT, UK.

***Abstract*—This paper argues that regression test optimization problems such as selection and prioritization require multi objective optimization in order to adequately cater for real world regression testing scenarios. The paper presents several examples of costs and values that could be incorporated into such a Multi Objective Regression Test Optimization (MORTO) approach.**

I. Introduction

Regression test optimization includes the two important related problems of selecting a subset of test cases that give maximum cost-value benefit and ordering test cases such that early attainment of this cost-value trade off is achieved. Both selection and prioritization problems have previously been studied as optimization problems, with a long history dating from 1977 [5]. However, much of the previous work has focussed on the problem of maximizing the attainment of *value* without adequately taking into account the *cost*.

It was not until 2001 that detailed empirical study was presented taking cost into account [3]. This was a single objective formulation that incorporated cost by treating the overall optimization function (the *fitness function*) to be a ratio: $\frac{\text{value}}{\text{cost}}$, where value in this case, was coverage and cost was execution time. As has been shown, when there are only the twin objectives of cost and value, it is possible to solve the selection problem using the greedily computed prioritization solution as a first (and fast) approximation [28].

However, the incorporation of cost into a two objective selection formulation in this way is only possible because we can place the two objectives into a ratio. If we seek to optimize for three objectives (or more), then this approach is not possible. It was not until 2007 that an approach was introduced that could handle more than two objectives simultaneously using pareto optimality for regression test selection/minimization [26].

Unfortunately, single objective Regression Test Optimization (RTO) is unlikely to be practical, because testers typically have many different objectives. Even if we manage to combine cost and value into a single objective and thereby 'squeeze two into one', we may have to consider different competing notions of cost. For example, execution time, price of data, costs due to risks. There are also many different notions of value, such as technical (fault coverage, code coverage, requirements coverage), social (human preferences) and business/commercial ('important feature' coverage). We may also have constraints that need to be factored into the overall RTO process, such as the various forms of dependence between test cases. We must ask ourselves whether there really are many real world testing scenarios in which the tester is concerned merely with a single objective? A recent survey of regression testing [29] found that

> "*Out of the 159 papers [surveyed], only 31 papers list a member of industry as an author or a co-author. More importantly, only 12 papers consider industrial software artefacts as a subject of the associated empirical studies.*"

Part of the blame for this lack of industrial application may lie with the overly restrictive nature of the 'Single Objective Assumption' which is implicit in much of the previous work on RTO. To challenge this historical orthodoxy, this paper presents a manifesto for MORTO: Multi Objective Regression Test Optimization.

Since 2007, there has been interest in MORTO [6], [8], [17], [21], [24]. The uptake of MORTO, by the Regression Testing and Search Based Software Engineering (SBSE) [9] community may lead to greater industrial interest in and use of regression testing optimization techniques. It will certainly lead to many interesting research problems, which pose the challenges of managing competing and potentially conflicting optimization objectives and constraints.

II. Regression Testing as an Optimization Problem

This section briefly reviews regression test case selection, minimization and prioritization, which collectively form part of the more general topic of RTO. A more detailed discussion of RTO can be found in the surveys of Yoo and Harman [27] and of Engström et al. [4]. This section formalises the relationship between minimization, prioritization and selection, arguing that, from an optimization perspective, there really are only two problems: selection and prioritization.

Definition 1 (Test Case Minimization): Let T be a test case and R be a test objective. M is a *test case minimization* of T with respect to R if and only if M is a subset of T that maximizes $R(T)$.

Traditionally, R is defined with respect to a set $\{r_1, \ldots, r_n\}$ of syntactic elements of the program under test to be covered by the test suite, such that $R(X)$ is defined to be the number of elements in $\{r_1, \ldots, r_n\}$ covered by X. Typically, we seek a set M that is smaller than T, one that is, therefore, a proper subset of T. Ideally, M will be minimal; no other minimized test suite will be smaller. In some approaches, we may seek to cover all elements of $\{r_1, \ldots, r_n\}$, rather than merely maximizing their coverage. When R is defined in terms of coverage, test case minimization becomes an instance of the set cover optimization problem.

As has been previously[1] noted [27] the so-called 'test set selection problem' can be formulated in terms of a test set

[1] A more recent version of this paper [27] is to appear in the Journal of Software Testing, Verification and Reliability [29].

minimization problem, so this too is an example of a minimal set cover optimization problem. More formally, Definition 2 defines the traditional test case selection problem in terms of Definition 1 of test case minimization.

Definition 2 (Test Case Selection): Let P' be a modified version of P and let T be a test suite. Let $\mathcal{C}_A$ be a function which takes a pair of programs and reports the set of program elements of the form A that are different in P' compared to P. In this context, A plays the role of test adequacy criterion, substituted by values such as <branch adequacy> and <statement adequacy>. Let $\mathcal{R}_A(X, P, P')$ be the number of elements of $\mathcal{C}_A(P, P')$ covered when P' is executed on X. A test set T' is an *A-adequate test case selection* of T with respect to (P, P') if and only if T' is a test case minimization with respect to $\mathcal{R}_A(T', P, P')$.

In the RTO literature, 'minimization' is typically taken to refer to the once-and-for-all elimination of 'redundant' test cases, while 'selection' is taken to refer to the temporary selection of test cases to be used on the next release of the system. However, these details are comparatively unimportant from an optimization point of view.

The definitions above are consistently named with the RTO literature. However, in terms of optimization there are simply selection problems and prioritization problems. In selection problems we choose elements from a set that best meet the optimization objectives, while in prioritization we seek to order a set so that elements meet the objectives better than those that follow them in the order. Hereinafter, we shall refer to both traditional minimization and traditional selection as 'Test Case Selection'; they are both selection problems from an optimization point of view.

Definition 3 (Test Case Prioritization): Let T be a test suite containing n elements and $T' = <t'_1, \ldots, t'_n>$ be a sequence on T. Let F be a function from test suite elements to some domain on which the relation $\geq$ imposes a total order. T' is a *test case prioritization* of T with respect to F if and only if $\forall i.1 \leq i \leq n-1$. $F(t'_i) \geq F(t'_{i+1})$. That is, F is monotonic over T.

These formalizations of Test Case Selection and Test Case Prioritization are based on the standard single objective formulations that can be found in the literature [1], [7], [15], [18], [19], [20]. Multi objective formulations can be defined in several ways, as is common with other work on multi objective SBSE [9]. A set of n objectives, $\{f_1, \ldots, f_n\}$, can be combined into a single weighted objective, $WO(x) = w_1 \times f_1(x) + \ldots + w_n \times f_n(x)$ for a set of weights $\{w_1, \ldots, w_n\}$. This is the simplest approach because WO can be used in place of R in Definition 1 or in place of F in Definition 3.

However, this 'weighted sum' approach is only applicable when a set of weights $\{w_1, \ldots, w_n\}$ is available. Often, RTO scenarios have many objectives for which it is unclear what the appropriate values of weights should be. In order to define weights we need to be prepared to determine how much benefit can be sacrificed per unit reduction in cost. When costs and benefits are complex amalgamations of different goals and stakeholder wishes this will not be possible. Fortunately, where weights are not obvious we can use a pareto optimal approach to manage the tensions between our competing objectives [9].

III. Outline of a MORTO Research Agenda

Regression testing objectives can be categorised into those which fall on the 'value' side (those to be maximized) and those which fall on the 'cost' side (those to be minimized). Of course, any minimization objective can usually be inverted to turn it into a maximization objective, but it is helpful to think of these as 'cost-' and 'value-' based objectives, since this establishes those that are most likely to be in tension.

For some objectives we have natural fitness functions derived from testing metrics [11]. There are also constraints that delimit the set of valid test cases. In search based optimization, it is possible to treat any constraint as also being an objective, so long as the degree to which the constraint is satisfied can be measured meaningfully. Where such a measurement is possible we can simply seek to minimise the degree to which the constraint is violated. This is only of use where the constraint is, to some extent a 'soft constraint'. Fortunately, such soft constraints are not uncommon, particularly for software testing, where the goal, rightly or wrongly, is often 'do the best we can with the resources available'.

In most realistic regression testing scenarios there will be at least one cost- and one value- based objective, though there may be many others (and also constraints to respected). As such, new and potentially interesting and important MORTO research challenges are to be found in almost any arbitrary combination of these sets of objectives, by choosing at least one from each of the cost and value categories.

A. Cost-based Objectives

Execution Time: Execution time is a natural candidate for an RTO. It has already been studied widely in the literature, following the work of Elbaum et al. [3] on cost cognizant prioritization. It was also used as a secondary objective in prioritization by Walcott et al. [23]. It was also used in regression test selection by Yoo and Harman, who introduced the idea of pareto optimization as a way to balance competing constraints in Regression Testing [26]. Execution time is clearly a pressing concern for a regression tester, given the short build cycles within which they will typically have to perform the regression test activities. However, there are many other costs to be taken into account:

Data Access Costs: In some systems, where access to databases determines the coverage of the application under test, the population of the data base will significantly affect the effectiveness of testing. We shall prefer realistic test cases, rather than a 'mocked up' version of the data base, into which data may be systematically, but nonetheless synthetically added. Such synthetic data can lead to many false positives, because integrity constraints are not handled by the automated synthetic test generation algorithm. It can also lead to many false negatives. For example, awkward corner cases that humans tend to use may be missed, while the synthetic generator design may inherit the same (wrong) assumptions that undermine the implementation.

Unfortunately, real test cases can cost real money. There may be many costs involved in populating a database with either *realistic* data (the generation of which costs human time)

or with *real* data (which has a monetary cost of access payable to the data provider). These cost minimization objectives may be balanced against other cost drivers, such as execution time, using a MORTO testing approach. MORTO will allow the regression tester to select and order their regression tests so that all forms of cost are minimized. Without MORTO, one cost may be unacceptably high, even though another is minimized, leading to the decision maker rejecting both the solutions and the approach that suggests them.

Third party Costs: Some systems interact with third parties, creating significant testing costs. For example, when testing service oriented systems [2], there may be a price for accessing the systems of a third party. However, without such third party service access, it may not be possible to test the system fully. In these scenarios the tester will wish to minimize third party costs while maximising value.

Technical Resources Costs: Some systems, such as embedded systems, are tightly coupled with their environment. These systems may consume resources that have a non-trivial cost. However, testing *in situ* or on a test rig may be required in order to run certain regression tests. The tester will want to reduce the consumption of such resources, while testing the system as fully as possible.

Setup Costs: In complex systems there may be set up costs associated with certain test cases. For instance, the test case may require devices, services, files and other aspects of the overall system to be configured in a particular way before the test can be executed. The execution of the test case itself may take negligible time. However, the set up costs for the test case may be significant in time and other costs. Such set up costs may also introduce dependencies, leading to an interaction between objectives and constraints.

Simulation Costs: Some systems are so expensive to test that a simulation is required. This is common in the automotive industry, for example, where automated testing and search based test data generation have been used [12], [25]. In these situations, the effort of developing or deploying a simulation of the real system behaviour may constitute a significant cost to be borne during testing. Where such costs can be estimated, then they can also form an objective for minimization.

B. Value-based Objectives

Code Based Coverage: There are many code-based aspects to be covered in testing, such as statement, branch and mutation coverage. These have been widely studied from the outset, with some penetration into industrial use [29]. As the literature develops, additional code-based features will emerge for which regression testing may seek coverage. Novel languages and programming paradigms will lead to their own regression testing coverage goals. All of these goals reflect the positive aims of RTO; those things that the tester seeks to maximize because they offer potential benefit. Most MORTO approaches are likely to include one such objective, if only for historical compatibility.

Non-Code-Based Coverage: Compared to code-based coverage, non-code-based objectives have been less thoroughly studied in the RTO literature, though there has been work on prioritization using system models [13], [14], [16] and requirements [22]. As models are increasingly used in Software Engineering they will come to represent the system ever more faithfully. It has been argued that all such descriptions will increasingly adopt the characteristics of source code according to the 'law of tendency towards executability' [10]. This migration towards executable models and specifications is likely to create additional coverage-based objectives.

Fault Model Sensitive: A regression testing approach should be sensitive to a fault model. If it is known that certain faults are likely, then this can be incorporated into RTO so that those tests that reveal more likely categories of faults are more likely to be selected or prioritized.

Fault History Sensitive: One particular kind of fault model is a fault history. There is no guarantee that past fault revelation confers an on-going value to test cases. Nevertheless, such previously fault-revealing test cases may be regarded as 'proven performers'. As a result, the inclusion of historical fault sensitivity into RTO may be important. Indeed, it has formed an objective in previous work [26].

Human Sensitive: Most testing scenarios involve sociological and subjective issues. The tester, manager and customer are all important stakeholders, each with their own opinions on testing priorities. These preferences may be hard to quantify, but their successful incorporation will certainly prove to be an important determinant of acceptance. Previous work has demonstrated that these 'human' issues, though subjective, can be accounted for in regression test proiroitization [30].

Business Sensitive: As well as the sociological (and highly subjective) factors, there are also more quantifiable business objectives. These may favour test cases according to their ability to test features that are directly related to revenue generation or target-market acquisition. As all testers know: not all features are equal.

C. MORTO Optimization Constraints

Precedence: Some test cases have to be performed before others because they establish a system state in which the subsequent test cases become possible, or for which these later tests perform better in some way. We may treat this as an objective where the constraint is soft, seeking to respect constraints where possible. In other situations these constraints may be 'hard constraints' that must be satisfied by the test process because subsequent tests simply will not apply unless the previous test cases have been executed.

Conjunction Constraint: Tests may be conjoined such that executing one, entails executing another. Such constraints may be soft (it is *advantageous* to the tester to test these two together) or hard (these two *must* be executed together).

Exclusivity Constraint: Two tests may be mutually exclusive. For instance, if one test completely exhausts a resource that is required by another, then these two tests cannot both be performed. Once again, these constraints may be soft or hard, but where they are present, the RTO process must take them into account. Otherwise, the test case selection and prioritization results produced by RTO may not be viable.

Dependence Constraint: Inclusion of one test may affect the cost of another. For instance, if we undertake the work required by a complex set up process for a certain test case, the same

set up may be re-usable by other test cases. In this way there may be a dependence between the cost of one test case and the costs of others. If we include one of the test cases, then the cost of all those that remain will be reduced; they share the same set up procedure and the costs associated with it.

IV. Conclusion

Testing is complex. There is no direct measure of fault-revelation likelihood and there are many different types of cost involved. This complex interplay between cost and value is further compounded by the many additional validity constraints, making test case selection and prioritization problems that naturally involve multi objective optimization. This paper argues that such a Multi Objective Regression Test Optimization (MORTO) approach is long overdue.

Acknowledgements I am most grateful to Per Runeson for the invitation to present this position paper at the **Regression 2011 workshop**. I am also very grateful to Shin Yoo for our many interesting and thought-provoking discussions on RTO topics, which have undoubtedly helped to shape the views I express in this position paper (though all errors remain my sole responsibility).

References

[1] David Wendell Binkley. The application of program slicing to regression testing. *Information and Software Technology Special Issue on Program Slicing*, 40(11 and 12):583–594, 1998.

[2] Mustafa Bozkurt, Mark Harman, and Youssef Hassoun. Testing web services: a survey. Technical Report TR-10-01, Department of Computer Science, King's College London, April 2010.

[3] Sebastian Elbaum, Alexey G. Malishevsky, and Gregg Rothermel. Incorporating varying test costs and fault severities into test case prioritization. In *Proceedings of the 23rd International Conference on Software Engineering (ICSE-01)*, pages 329–338. IEEE Computer Society, May 2001.

[4] Emelie Engström, Per Runeson, and Mats Skoglund. A systematic review on regression test selection techniques. *Information & Software Technology*, 52(1):14–30, 2010.

[5] K. Fischer. A test case selection method for the validation of software maintenance modifications. In *International Computer Software and Applications Conference (COMPSAC '77)*, pages 421–426. IEEE Computer Society Press, 1977.

[6] Sapna P. G. and Hrushikesha Mohanty. Automated test scenario selection based on levenshtein distance. In Tomasz Janowski and Hrushikesha Mohanty, editors, 6^{th} *Distributed Computing and Internet Technology (ICDCIT'10)*, volume 5966 of *Lecture Notes in Computer Science (LNCS)*, pages 255–266. Springer-Verlag (New York), Bhubaneswar, India, February 2010.

[7] Todd Graves, Mary Jean Harrold, Jung-Min Kim, Adam Porter, and Gregg Rothermel. An empirical study of regression test selection techniques. In *Proceedings of the 20th International Conference on Software Engineering*, pages 188–197. IEEE Computer Society Press, April 1998.

[8] Qing Gu, Bao Tang, and DaoXu Chen. Optimal regression testing based on selective coverage of test requirements. In *International Symposium on Parallel and Distributed Processing with Applications (ISPA 10)*, pages 419 – 426, September 2010.

[9] Mark Harman. The current state and future of search based software engineering. In Lionel Briand and Alexander Wolf, editors, *Future of Software Engineering 2007*, pages 342–357, Los Alamitos, California, USA, 2007. IEEE Computer Society Press.

[10] Mark Harman. Why source code analysis and manipulation will always be important. In 10^{th} *IEEE International Working Conference on Source Code Analysis and Manipulation*, Timisoara, Romania, 2010. keynote paper.

[11] 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.

[12] Mark Harman and Joachim Wegener. Getting results with search–based software engineering: Tutorial. In 26^{th} *IEEE International Conference and Software Engineering (ICSE 2004)*, pages 728–729, Los Alamitos, California, USA, 2004. IEEE Computer Society Press.

[13] Bogdan Korel, Mark Harman, S. Chung, P. Apirukvorapinit, and R. Gupta. Data dependence based testability transformation in automated test generation. In 16^{th} *International Symposium on Software Reliability Engineering (ISSRE 05)*, pages 245–254, Chicago, Illinios, USA, November 2005.

[14] Bogdan Korel, Luay Tahat, and Mark Harman. Test prioritization using system models. In 21^{st} *IEEE International Conference on Software Maintenance*, pages 559–568, Los Alamitos, California, USA, 2005. IEEE Computer Society Press.

[15] Zheng Li, Mark Harman, and Rob Hierons. Meta-heuristic search algorithms for regression test case prioritization. *IEEE Transactions on Software Engineering*, 33(4):225–237, 2007.

[16] Alireza Mahdian, Anneliese Amschler Andrews, and Orest Pilskalns. Regression testing with UML software designs: A survey. *Journal of Software Maintenance*, 21(4):253–286, 2009.

[17] Gustavo H. L. Pinto and Silvia Regina Vergilio. A multi-objective genetic algorithm to test data generation. In 22^{nd} *IEEE International Conference on Tools with Artificial Intelligence (ICTAI'10)*, pages 129–134. IEEE Computer Society, 2010.

[18] Gregg Rothermel, Mary Jean Harrold, Jeffery von Ronne, and Christie Hong. Empirical studies of test suite reduction. *Software Testing, Verification, and Reliability*, 4(2):219–249, December 2002.

[19] Gregg Rothermel, Roland Untch, Chengyun Chu, and Mary Jean Harrold. Prioritizing test cases for regression testing. *IEEE Transactions on Software Engineering*, 27(10):929–948, October 2001.

[20] Gregg Rothermel, Roland H. Untch, Chengyun Chu, and Mary Jean Harrold. Test case prioritization: An empirical study. In *Proceedings; IEEE International Conference on Software Maintenance*, pages 179–188, Los Alamitos, California, USA, 1999. IEEE Computer Society Press.

[21] Jerffeson Teixeira de Souza, Camila Loiola Maia, Fabrcio Gomes de Freitas, and Daniel Pinto Coutinho. The human competitiveness of search based software engineering. In *Proceedings of* 2^{nd} *International Symposium on Search based Software Engineering (SSBSE 2010)*, pages 143 – 152, Benevento, Italy, 2010. IEEE Computer Society Press.

[22] Hema Srikanth and Laurie Williams. On the economics of requirements-based test case prioritization. *ACM SIGSOFT Software Engineering Notes*, 30(4):1–3, 2005.

[23] Kristen R. Walcott, Mary Lou Soffa, Gregory M. Kapfhammer, and Robert S. Roos. Time aware test suite prioritization. In *International Symposium on Software Testing and Analysis (ISSTA 06)*, pages 1 – 12, Portland, Maine, USA., 2006. ACM Press.

[24] Huai Wang, W. K. Chan, and T. H. Tse. On the construction of context-aware test suites. Technical Report TR-2010-01, Hong Kong University, 2010.

[25] Joachim Wegener and Oliver Bühler. Evaluation of different fitness functions for the evolutionary testing of an autonomous parking system. In *Genetic and Evolutionary Computation Conference (GECCO 2004)*, pages 1400–1412, Seattle, Washington, USA, June 2004. LNCS 3103.

[26] Shin Yoo and Mark Harman. Pareto efficient multi-objective test case selection. In *International Symposium on Software Testing and Analysis (ISSTA'07)*, pages 140 – 150, London, United Kingdom, July 2007. Association for Computer Machinery.

[27] Shin Yoo and Mark Harman. Regression testing minimisation, selection and prioritisation: A survey. Technical Report TR-09-09, King's College London, 2009.

[28] Shin Yoo and Mark Harman. Using hybrid algorithm for pareto efficient multi-objective test suite minimisation. *Journal of Systems and Software*, 83(4):689–701, 2010.

[29] Shin Yoo and Mark Harman. Regression testing minimisation, selection and prioritisation: A survey. *Journal of Software Testing, Verification and Reliability*, 2011. To appear.

[30] Shin Yoo, Mark Harman, Paolo Tonella, and Angelo Susi. Clustering test cases to achieve effective and scalable prioritisation incorporating expert knowledge. In *ACM International Conference on Software Testing and Analysis (ISSTA 09)*, pages 201–212, Chicago, Illinois, USA, 19th – 23rd July 2009.