Automated Test Data Generation using Search Based Software Engineering

Mark Harman
CREST,
King's College London,
Strand, London, WC2R 2LS
United Kingdom

Abstract

Generating test data is a demanding process. Without automation, the process is slow, expensive and error-prone. However, techniques to automate test data generation must cater for a bewildering variety of functional and non-functional test adequacy criteria and must either implicitly or explicitly solve problems involving state propagation and constraint satisfaction. This talk will show how optimisation techniques associated with Search Based Software Engineering (SBSE) have been used to automate test data generation. The talk will survey the area and present the results of recent work on characterising, transforming and eliding test data search landscapes.

1. Introduction

Search based optimization has been applied to a wide variety of Software Engineering application areas including requirements engineering, project planning, cost estimation, maintenance, reverse engineering, refactoring, program comprehension, service-oriented software engineering and quality assessment. This area has come to be known as Search Based Software Engineering [14]. A recent survey of SBSE was presented at ICSE 07 as part of the 'Future of Software Engineering' track [12].

The origins of this recent explosion of SBSE work can be traced back to early work on the application of search based optimization for test data generation [32]. The past five years have witnessed a dramatic rise in interest in Search Based Testing [22].

Testing is a natural target for SBSE work; the input space within which test data is sought is typically large, but well defined, while the test goal can often be expressed as a fitness function. The generic nature of the solution makes Search Based Testing very attractive. Simply by changing the input space and fitness function, the approach can be adapted to different test data generation problems.

SBSE has been applied to a wide variety of testing goals and issues, including testing concerned with structural [9, 19, 23, 25, 29], functional [30], non functional [31] and state–based properties [8, 11]; testing for robustness [26], stress [5, 10] and security [7]; mutation [2, 34], regression [21, 24, 28, 33], interaction [6], integration testing [4] and exception testing [27]. This keynote talk will present a summary of existing work, together with recent results from the analysis of performance of difference search algorithms for structural test data generation [15] and approaches to transforming and reducing search spaces [1, 3, 13, 20].

2. Acknowledgements

The author's work is currently funded by the EPSRC projects, SEBASE, ConTRACTs and ASTReNet and by the EU project EvoTest. This paper draws on these projects and from many fruitful disucssions with their members and collaborators. More specifically, the keynote also draws on material from previous keynotes and tutorials prepared in collaboration with Joachim Wegener [16, 17, 18].

References

[1] A. Baresel, D. W. Binkley, M. Harman, and B. Korel. Evolutionary testing in the presence of loop–assigned flags: A testability transformation approach. In *International Symposium on Software Testing and Analysis (ISSTA 2004)*, pages 108–118, July 2004.

[2] B. Baudry, F. Fleurey, J.-M. Jézéquel, and Y. L. Traon. From genetic to bacteriological algorithms for mutation-based testing. *Software Testing, Verification and Reliability*, 15(2):73–96, 2005.

[3] D. W. Binkley and M. Harman. Analysis and visualization of predicate dependence on formal parameters and global variables. *IEEE Transactions on Software Engineering*, 30(11):715–735, 2004.

[4] L. C. Briand, J. Feng, and Y. Labiche. Using genetic algorithms and coupling measures to devise optimal integration test orders. In *Software Engineering and Knowledge Engineering (SEKE 02)*, pages 43–50, 2002.

[5] L. C. Briand, Y. Labiche, and M. Shousha. Stress testing real-time systems with genetic algorithms. In *Genetic and Evolutionary Computation Conference, GECCO 2005, Proceedings, Washington DC, USA, June 25-29, 2005*, pages 1021–1028. ACM, 2005.

[6] M. B. Cohen, P. B. Gibbons, W. B. Mugridge, and C. J. Colbourn. Constructing test suites for interaction testing. In *Proceedings of the 25th International Conference on Software Engineering (ICSE-03)*, pages 38–48, Piscataway, NJ, May 3–10 2003. IEEE Computer Society.

[7] C. Del Grosso, G. Antoniol, E. Merlo, and P. Galinier. Detecting buffer overflow via automatic test input data generation. *Computers and Operations Research (COR) focused issue on Search Based Software Engineeering*.

[8] K. Derderian, R. Hierons, M. Harman, and Q. Guo. Automated Unique Input Output sequence generation for conformance testing of FSMs. *The computer Journal*, 49(3):331–344, 2006.

[9] R. Ferguson and B. Korel. The chaining approach for software test data generation. *ACM Transactions on Software Engineering and Methodology*, 5(1):63–86, Jan. 1996.

[10] C. D. Grosso, G. Antoniol, M. D. Penta, P. Galinier, and E. Merlo. Improving network applications security: a new heuristic to generate stress testing data. In *GECCO 2005: Proceedings of the 2005 conference on Genetic and evolutionary computation*, volume 1, pages 1037–1043, Washington DC, USA, 25-29 June 2005. Association for Computer Machinery.

[11] Q. Guo, R. M. Hierons, M. Harman, and K. Derderian. Constructing multiple unique input/output sequences using evolutionary optimisation techniques. *IEE Proceedings — Software*, 152(3):127–140, 2005.

[12] M. Harman. The current state and future of search based software engineering. In L. Briand and A. Wolf, editors, *Future of Software Engineering 2007*, Los Alamitos, California, USA, 2007. IEEE Computer Society Press. To appear.

[13] M. Harman, L. Hu, R. M. Hierons, J. Wegener, H. Sthamer, A. Baresel, and M. Roper. Testability transformation. *IEEE Transactions on Software Engineering*, 30(1):3–16, Jan. 2004.

[14] M. Harman and B. F. Jones. Search based software engineering. *Information and Software Technology*, 43(14):833–839, Dec. 2001.

[15] M. Harman and P. McMinn. A theoretical and empirical analysis of evolutionary testing and hill climbing for structural test data generation. In *International Symposium on Software Testing and Analysis (ISSTA'07)*, London, United Kingdom, July 2007. Association for Computer Machinery. To Appear.

[16] M. Harman and J. Wegener. Evolutionary testing: Tutorial. In *Genetic and Evolutionary Computation (GECCO)*, Chicago, July 2003.

[17] M. Harman and J. 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.

[18] M. Harman and J. Wegener. Search based testing. In 6^{th} *Metaheuristics International Conference (MIC 2005)*, Vienna, Austria, Aug. 2005. To appear.

[19] B. Jones, H.-H. Sthamer, and D. Eyres. Automatic structural testing using genetic algorithms. *The Software Engineering Journal*, 11:299–306, 1996.

[20] B. Korel, M. 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, Nov. 2005.

[21] Z. Li, M. Harman, and R. Hierons. Meta-heuristic search algorithms for regression test case prioritization. *IEEE Transactions on Software Engineering*. To appear.

[22] P. McMinn. Search-based software test data generation: A survey. *Software Testing, Verification and Reliability*, 14(2):105–156, June 2004.

[23] C. Michael, G. McGraw, and M. Schatz. Generating software test data by evolution. *IEEE Transactions on Software Engineering*, (12):1085–1110, Dec. 2001.

[24] Nashat Mansour, Rami Bahsoon and G. Baradhi. Empirical comparison of regression test selection algorithms. *Systems and Software*, 57(1):79–90, 2001.

[25] R. P. Pargas, M. J. Harrold, and R. R. Peck. Test-data generation using genetic algorithms. *The Journal of Software Testing, Verification and Reliability*, 9:263–282, 1999.

[26] A. Schultz, J. Grefenstette, and K. Jong. Test and evaluation by genetic algorithms. *IEEE Expert*, 8(5):9–14, 1993.

[27] N. Tracey, J. Clark, K. Mander, and J. McDermid. Automated test-data generation for exception conditions. *Software Practice and Experience*, 30(1):61–79, 2000.

[28] K. R. Walcott, M. L. Soffa, G. M. Kapfhammer, and R. 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.

[29] J. Wegener, A. Baresel, and H. Sthamer. Evolutionary test environment for automatic structural testing. *Information and Software Technology*, 43(14):841–854, 2001.

[30] J. Wegener and O. Bühler. Evaluation of different fitness functions for the evolutionary testing of an autonomous parking system. In *Genetic and Evolutionary Computation Conference (GECCO 2004)*, Seattle, Washington, USA, June 2004. To Appear.

[31] J. Wegener and M. Grochtmann. Verifying timing constraints of real-time systems by means of evolutionary testing. *Real-Time Systems*, 15(3):275 – 298, 1998.

[32] S. Xanthakis, C. Ellis, C. Skourlas, A. Le Gall, S. Katsikas, and K. Karapoulios. Application of genetic algorithms to software testing. In 5^{th} *International Conference on Software Engineering and its Applications*, pages 625–636, Toulouse, France, 1992.

[33] S. Yoo and M. Harman. Pareto efficient multi-objective test case selection. In *International Symposium on Software Testing and Analysis (ISSTA'07)*, London, United Kingdom, July 2007. Association for Computer Machinery. To Appear.

[34] Y. Zhan and J. A. Clark. Search-based mutation testing for simulink models. In H.-G. Beyer and U.-M. O'Reilly, editors, *Genetic and Evolutionary Computation Conference (GECCO 05)*, pages 1061–1068, Washington DC, USA, June 2005. Association for Computer Machinery.