Editorial to the special issue of Computers & Operations Research on Search-Based Software Engineering

This focused issue of Computers and Operations Research contains seven excellent papers on the application of optimization and metaheuristic search techniques to problems in software engineering. The focused issue arises from a call for papers, for which a total of 18 papers were received. After careful and detailed refereeing, a total of 7 papers were finally selected for inclusion in the focussed issue.

Optimization and search based techniques have been applied to a wide range of software engineering problems, using a large number of techniques associated with OR. This area of research has come to be known as Search Based Software Engineering (SBSE) [5, 10]. The past five years have witness a dramatic growth in conferences, workshop, tracks, and special issues on SBSE. Previous work on SBSE spans the spectrum of software engineering problems from requirements engineering [2], project planning and cost estimation [1, 8, 12], through test data generation [13], to automated maintenance [3, 9, 14, 15, 16], service-oriented software engineering [4], compiler optimization [6, 7] and quality assessment [11].

The seven papers in this issue cover a wide ranging and diverse set of software engineering activities, including project planning, component selection, security and testing. This diversity shows the healthy state of search based software engineering research and practice. The papers also cover a wide range of optimization and search techniques, including classical optimization approaches such as linear programming and branch and bound, but also adaptive search techniques such as tabu search, evolution strategies and genetic algorithms.

The article by Diaz et al. deals with the question how to achieve a sufficient degree of test coverage by an automatic tool for structural testing. Several meta-heuristic approaches have already been applied in this context, in particular meta-heuristics from the evolutionary algorithms family. However, little experience is available concerning the use of tabu search for this purpose, although tabu search has been proven as one of the most successful meta-heuristic techniques in other areas of application. In the present article, tabu search is shown to be effective and efficient also for the generation of white-box test cases, reaching 100 % branch coverage by a moderate number of test cases for some example programs with rather difficult control flow graphs and branching conditions.

Barreto et al. devote their article to a hard problem in software project

management, that of staffing a software project. Explicit reference is made to different "characteristics" (e.g., skills, capabilities, experiences or roles) of developers. The goal is to allocate available personnel to the activities arising in different phases of the software development process, respecting different types of constraints. Barreto et al. formulate the allocation problem as a constraint satisfaction problem and solve it by backtracking-type techniques and by branch-and-bound. Several possible objective functions are discussed. A final experimental investigation compares the solutions obtained in this way to those provided by participants in an experiment.

The decision whether to build or to buy software components has already been addressed by other authors, but the article by Cortellessa et al. raises the quantitative treatment of this problem to a new level of generality and realism. Several relevant factors such as cost, delivery time or product reliability are integrated within a unified framework. Contrary to prior approaches, the authors take into account the possibility of in-house development; an additional degree of freedom can be exploited by a decision on the amount of testing to be invested. The resulting optimization model formulations are non-linear and mixed-integer. To make larger problem instances solvable, linear approximations for quality and reliability functions are derived.

In the work by Ahmed and Hernadi, a standard genetic algorithm approach to the automatic test data generation for achieving path coverage is extended by the modification of targeting not at a single path per genetic algorithm run, but simultaneously at a set of paths. The idea behind this approach is that while trying to find test data for the execution of a certain path, suitable data for that of another still uncovered path can be detected as a by-product. Of course, the approach requires an appropriate extension of the fitness function definitions for the genetic algorithm. The authors discuss and test several fitness function concepts, and they compare their method to two prior approaches by Lin and by Pei, respectively, in an experiment based on six "classical" test programs.

The contribution by Del Grosso et al. addresses the question how to detect buffer overflow problems by means of testing. Buffer overflow attacks can have serious consequences and they form a primary cause of software vulnerability. This motivates the development of testing techniques aimed specifically at this problem. The proposed method relies on a genetic algorithm based approach to generate test input data. To guide the search in an efficient way, a particular fitness function combining diverse weighted metrics is applied. The weights are optimized by solving a linear program. The authors show by experiments on two sets of C applications that the presented approach outperforms a previously proposed variant. The combination of

a meta-heuristic techniques with linear programming also imbues a special methodological interest; this is a combination that is uncommon in previous SBSE work.

Buehler and Wegener start their article with an introduction into the application of the evolutionary testing paradigm to one of the most important classes of testing: functional testing. They show by two specific examples from the automotive industry—the testing of automated parking systems and that of brake assistance systems—how to use the concept in practice. A crucial issue in evolutionary functional testing is the definition of suitable objective functions. The complex properties of the the applications considered make this a highly nontrivial task. An experimental evaluation of the results for the brake assistance system is presented, including a comparison both with manually and randomly generated test cases. The evaluation clearly demonstrates the advantages of the approach.

Alba and Chicano investigate the condition coverage achieved in automatic test data generation by two evolutionary algorithms, namely evolutionary strategies (ES) and genetic algorithms (GA). For both variants, both a panmictic version (working on a single, unstructured population) and a decentralized version (using a partition into sub-populations with migration in a ring topology) are studied. The algorithms are applied to twelve test programs. It turns out that, contrary to expectations from results in other application fields, the decentralized versions have no statistically significant advantages over their panmictic counterparts. However, ES shows significantly better performance than GA in both versions.

We would like to warmly thank the authors and the anonymous reviewers for the time and expertise in helping to produce this focussed issue. We would also like to extend our sincere thanks to both the outgoing editor, Gilbert Laporte, and the incoming editor, Stefan Nickel for their support for this focussed issue. Special thanks are also due to Zheng Li, who provided valuable technical and administrative support, during the reviewing process.

Walter J. Gutjahr
University of Vienna, Austria
walter.gutjahr@univie.ac.at

Mark Harman
King's College, London
Mark.Harman@kcl.ac.uk

Guest Editors

References

[1] G. Antoniol, M. D. Penta, and M. Harman. Search-based techniques applied to optimization of project planning for a massive maintenance project. In 21^{st} *IEEE International Conference on Software Maintenance*, pages 240–249, Los Alamitos, California, USA, 2005. IEEE Computer Society Press.

[2] A. Bagnall, V. Rayward-Smith, and I. Whittley. The next release problem. *Information and Software Technology*, 43(14):883–890, Dec. 2001.

[3] S. Bouktif, G. Antoniol, E. Merlo, and M. Neteler. A novel approach to optimize clone refactoring activity. In *GECCO 2006: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, volume 2, pages 1885–1892, Seattle, Washington, USA, 8-12 July 2006. ACM Press.

[4] G. Canfora, M. D. Penta, R. Esposito, and M. L. Villani. An approach for qoS-aware service composition based on genetic algorithms. In H.-G. Beyer and U.-M. O'Reilly, editors, *Genetic and Evolutionary Computation Conference, GECCO 2005, Proceedings, Washington DC, USA, June 25-29, 2005*, pages 1069–1075. ACM, 2005.

[5] J. Clark, J. J. Dolado, M. Harman, R. M. Hierons, B. Jones, M. Lumkin, B. Mitchell, S. Mancoridis, K. Rees, M. Roper, and M. Shepperd. Reformulating software engineering as a search problem. *IEE Proceedings — Software*, 150(3):161–175, 2003.

[6] M. Cohen, S. B. Kooi, and W. Srisa-an. Clustering the heap in multi-threaded applications for improved garbage collection. In *GECCO 2006: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, volume 2, pages 1901–1908, Seattle, Washington, USA, 8-12 July 2006. ACM Press.

[7] K. D. Cooper, P. J. Schielke, and D. Subramanian. Optimizing for reduced code space using genetic algorithms. In *Proceedings of the ACM Sigplan 1999 Workshop on Languages, Compilers and Tools for Embedded Systems (LCTES'99)*, volume 34.7 of *ACM Sigplan Notices*, pages 1–9, NY, May 5 1999. ACM Press.

[8] J. J. Dolado. A validation of the component-based method for software size estimation. *IEEE Transactions on Software Engineering*, 26(10):1006–1021, 2000.

[9] D. Fatiregun, M. Harman, and R. Hierons. Search-based amorphous slicing. In 12^{th} *International Working Conference on Reverse Engineering (WCRE 05)*, pages 3–12, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA, Nov. 2005.

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

[11] T. M. Khoshgoftaar, L. Yi, and N. Seliya. A multiobjective module-order model for software quality enhancement. *IEEE Transactions on Evolutionary Computation*, 8(6):593– 608, December 2004.

[12] C. Kirsopp, M. Shepperd, and J. Hart. Search heuristics, case-based reasoning and software project effort prediction. In *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1367–1374, San Francisco, CA 94104, USA, 9-13 July 2002. Morgan Kaufmann Publishers.

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

[14] B. S. Mitchell and S. Mancoridis. On the automatic modularization of software systems using the bunch tool. *IEEE Transactions on Software Engineering*, 32(3):193–208, 2006.

[15] M. O'Keeffe and M. OCinneide. Search-based software maintenance. In *Conference on Software Maintenance and Reengineering (CSMR'06)*, pages 249–260, Mar. 2006.

[16] O. Seng, M. Bauer, M. Biehl, and G. Pache. Search-based improvement of subsystem decompositions. In H.-G. Beyer and U.-M. O'Reilly, editors, *Genetic and Evolutionary Computation Conference, GECCO 2005, Proceedings, Washington DC, USA, June 25-29, 2005*, pages 1045–1051. ACM, 2005.