Summary of "Semi-Automatic Search-based Test Generation"

Nikhil Satish Pai Nikhil Anand

18th October 2015

Important Keywords:

ii1. Manual Testing: Process carried out manually to execute test cases without the use of automation tools.

ii2. search-based testing: Search-based testing casts the testing problem as a search problem and applies efficient algorithms to find inputs that can serve as suitable tests. The search space is the input domain of the test object. A set of inputs to that test is a potential solution to this object. The search is guided by a fitness function that estimates how close a candidate solution is to satisfying a coverage goal.

ii3.test case generation: essential part of software engineering used in the generation of data sets to test the code coverage of a software application. They generate input data-set to traverse the code paths that need to be tested.

ii4. dynamic symbolic execution: Symbolic execution works by analyzing a program to determine what inputs cause each part of a program to execute. Dynamic symbolic execution is a combination of concrete execution with symbolic execution. It works on the principle of automatically exploring the program execution space and reason about a program path-by-path which is an advantage over reasoning about a program input-by-input as other testing paradigms.

Brief notes:

iii1.Motivational Statement

Automated search based testing uses meta heuristic techniques to determine the various inputs in order to achieve high code coverage. But if the fitness function is unable to provide the required degree of guidance (especially in object oriented programs) then the search behaves like a random search where optimal solutions may not be found. The paper proposes an integrated approach where the search algorithm covers as much of the code as possible and when the further mutations of the solutions do not show any improvement the output is show to a user who will modify the solution in order to achieve higher coverage.

iii2.Future Work: there are many Future works mentioned in the paper, the most prevalent one being improving the search(as most of the problems are due to limitations in the search operators or fitness function) such that higher coverage can be achieved automatically, thus reducing the manual input necessary

iii4.Baseline Results :

The paper tests the result of the semi-automated testing algorithm with Evosuite, which is automatic test case generator for Java. The code coverage for both was measured on 20 random projects from SF100, which is a random set of the 100 most popular open source projects on sourceforge. Also the number of edits (measured in terms of lines added) in manual testing and the other 2 approaches.

iii3.Related work:

*Xiao et al [12] talks about approach to report problems that a testing tool encounters during test generation to the user. They call out external method calls and object creation as the main problems in test generation based on dynamic symbolic execution.

**Fraser and Arcuri [4] talks about a study on an unbiased sample pointing that in practice environmental dependencies like file access are a major inhibitor towards high code coverage and object creation and complex string-based calculations can also be for search-based testing.

Area of Improvement:

iv1. The paper does not consider the experience, program comprehension and familiarity of the manual tester into account when measuring the various parameters like coverage, the experiment needs to be repeated to see how these affect the final results.

iv2. The algorithm needs to be tested on larger code projects to determine its efficiency against automated testing techniques

*[12] X. Xiao, T. Xie, N. Tillmann, and J. de Halleux. Precise identification of problems for structural test generation. In Proceedings of the 33rd International Conference on Software Engineering, ICSE ’11, pages 611–620, New York, NY, USA, 2011. ACM.

**[4] G. Fraser and A. Arcuri. Sound empirical evidence in software testing. In ACM/IEEE International Conference on Software Engineering (ICSE), 2012. To appear.

Relation to Original paper :

The current paper proposes a new technique for automated test generation for high code coverage in addition to existing techniques like random testing, dynamic symbolic execution, search based software testing and the combination of dynamic symbolic execution and search based proposed in the original paper.

Reference to the Paper:

Y. Pavlov and G. Fraser, “Semi-automatic Search-Based Test Generation,” in 5th International Workshop on Search-Based Software Testing (SBST’12) at ICST’12, 2012, pp. 777-784.