Abstract

Software debugging is both challenging and expensive. It consists of two main tasks: localization and correction of bugs. Between these two tasks, the task of fault localization is usually much more expensive. Any improvements to this activity can significantly reduce the total software debugging expenses. The main motive of this research is to improve upon the fault localization (FL) techniques that make use of available execution information from the test cases, in a way that is effective, efficient and scalable and also able to handle programs with multiple faults.

To meet the identified objective, first we present a Fisher's Test based statistical fault localization technique. Our proposed Fisher's Test based technique, which we have named FTFL, uses statement coverage information and test execution results to compute the suspiciousness of the executable statements. Experimental results show that FTFL technique requires 34.61% less statement examination than existing FL techniques.

Subsequently, we present two hierarchical approaches to FL. We first compute the suspiciousness scores of all functions. After that, most suspicious functions are analyzed at statement level to localize the fault. For prioritizing the functions, we have define different function-level features, viz., function execution ratio, function complexity metric, and function dependency relations. Using our proposed hierarchical approaches, we could obtain at least 50% reduction in the candidate statements while localizing bugs as compared to the existing techniques such as DStar, Tarantula, Crosstab, Ochiai, BPNN and RBFNN.

We also report an extension to our proposed hierarchical approaches for localization of multi-fault programs based on the idea of parallel debugging. It involves partitioning of the failed test cases into different clusters such that each cluster targets a different fault. We combine all the passed test cases with each failed test case clusters. The resultant test suites are used to localize different faults.

After gathering all the changes detected by failed test case clusters, the program is retested. If any test case fails, the same process is repeated again. This process continues until all the test cases pass.

We finally use an ensemble of different fault localization techniques viz., spectrum-based, mutation-based, and neural network-based techniques for FL. Our ensemble based approach, which we have named EBFL, classifies the statements into two classes: *suspicious* and *non-suspicious* and helps to reduce the search space. EBFL model performs, on an average, 58% more effectively compared to the existing FL techniques and at least 16.95% more effectively than our other proposed techniques, FTFL, HieDeep, FDMLN, and DD*.
Keywords: Fault Localization, Debugging, Program Analysis.