**CBFL: Improving Software Fault Localization by Analyzing Statement Complexity**

**Abstract**

Software fault localization, which is an important software quality assurance technology, provides the location of the faults in software to improve the efficiency of debugging and repairing. In previous research, software fault localization techniques, such as spectrum-based**,** mutation-based, and program slicing, have been widely used and achieved good results. However, many statements could have same suspicious values by using these techniques, which will consume large amount of manual effort to confirm and affect the accuracy of fault localization. For example, using Ochiai or DStar to locate 395 faulty versions of 6 projects in Defects4J, nearly 70% of the faulty versions have more than one suspicious statement are ranked as tied 1. To address the above problem, this paper proposes a complexity-based fault localization (CBFL) technique to improve the precision of fault localization. First, it uses traditional techniques such as spectrum-based fault localization (SBFL) and mutation-based fault localization (MBFL) to locate faults, and gets suspicious values of statements in the program; then, it performs complexity analysis for the suspicious statements ranked are top tied 1, extracts features to build the model of statements; finally, it trains a classification model of fault statements based on complexity features of statements, and predicts the position of faults in the statements that are more likely to have faults. Based on the CBFL, this paper implements a fault localization tool, and conducts simulation experiments on the Defects4J dataset. In comparing with the DStar and MUSE, the simulation results show that the precision of CBFL outperforms DStar and MUSE in terms of *Einspect*@*n* and *Exam*.