MUTATION TESTING

Type : Learning Oriented

Avinash Thadamatha Lokesh Sannuthi

CSC 8350: Advanced Software Engineering Department of Computer Science, Georgia State University
Spring, 2017

Abstract:

Mutation testing is used to design new software tests and evaluate the quality of existing software tests. Mutation testing involves modifying a program. Mutation testing evaluates the quality of existing software tests. The idea is to modify (mutate) code covered by tests in a small way and check whether the existing test set will detect and reject the change [MUTTES]. If it doesn’t, it means the tests do not match the code’s complexity and leave one or more of its aspects untested.

Mutation testing plays an important role in the software development as it brings a whole new kind of errors to the developer's attention. It is the most powerful method to detect hidden defects, which might be impossible to identify using the conventional testing techniques. Debugging and Maintaining the product would be more easier than ever.

In this review, we will be focusing on Faster mutation testing.

Faster Mutation Testing is a group of techniques to reduce the cost of mutation testing by prioritizing and reducing tests to more quickly determine the set of systemize mutants. Faster mutation testing involves,

• Coverage-based initial test ordering
• Power based adaptive testing ordering
• Test Prioritization
• Test reduction
**Expected outcome:** Mutation testing is not widely used because of lack of awareness and inability to successfully integrate into the software development process, but the increased use of object oriented programming languages and unit testing frameworks has led to the creation of mutation testing tools for many programming languages as a way to test individual portions of an application. So, we would like to introduce this topic and bring awareness as it could be a potential for expansion of vast testing mechanisms.

**References:**


[4]. Elbaum, Sebastian; Malishevsky, Alexey G.; and Rothermel, Gregg, "Prioritizing Test Cases for Regression Testing" (2000). CSE Technical reports. 27.


**Other References:**