Finding a cheaper alternative to mutation score as test case quality indicator

School of Information and Computer Science
University of California, Irvine

1. Thesis

This research aims to find whether there is a proxy to mutation score as an indicator of test case quality. We aim to investigate the correlation between mutation score and test smell for this purpose. If there is a high correlation between mutation score and test smells, we can use test smells, which are less resource-intensive to calculate, as a proxy of mutation score.

2. Purpose

a. Why mutation score?

Mutation analysis [1] is a simple way to verify the correctness of the test code and the testing process. The mechanism of mutation testing is relatively simple. A unit-tested code is chosen to introduce different mutations. The degree of mutation applied to the code may vary, but generally changes should be kept minimal to avoid affecting the code's goals. Common mutations include making changes to logical operators, such as changing '+' to '-' or "==" to "!=" in code. After applying the mutations, the test cases are executed on the original code and mutated code. After comparing the output of the original code and each mutant, the mutants' outputs that are different from the original code are tagged as "killed mutants". Then the ratio of total mutants generated and mutants killed is used to calculate mutation score.

b. Why find correlation between test smell?

Just like regular code, test code is subject to inappropriate or imprecise programming practices that may contain bugs or degrade code quality. This can result in code that is harder to understand and maintain, and more prone to bugs. Test smells[2] can be used to describe this potentially problematic test code. Smells do not refer to bugs in the code, but to the presence of features in the code that reduce the quality of the code, such as redundant assertion, exception handling, etc. Therefore, for the test smells detection [3], it is not necessary to run the test code or modify the test code, but to judge the structure and content of the code. So testing smells is fast for testing code quality. On the contrary, although mutation analysis can check the correctness of the code very accurately, it also has the disadvantage of high cost. Mutation analysis is labor-intensive and time consuming. Therefore, the aim of this research is to find the correlation between the mutation score and the test smells. When the mutation score is higher, test whether the smell is also more severe in code. If there is a correlation between mutation scores and test smells, then whether test smells are consistently less expensive for testing code to reflect the correctness of the test code.

c. How to find correlation?

First we need to find a lot of Java and Python unit test code for mutation analysis and test smell detection. After selecting a suitable test code set, tools for mutation analysis and code smells are selected separately. Tools currently targeted include mutation tools: Universal Mutator[4], Stryker Mutator [5], etc. Also, testing smells detection tools: Test Smell detector [6], Pynose[7], etc. We can then compare the Mutation Scores with the test smells results to analyze the correlation among them.

3. Objective

- Get familiar with Mutation Analysis and Test Smell Detection
- Find Java and Python unit test codes
- Find Mutation Tools and Test smell detectors
- Select proper tools and detectors
- Apply tools and detectors to code set
- Compare the correlation between mutation scores and test smells

4. Responsibility

My current responsibility is to understand the mechanism of mutation analysis and test smell detection. I will read the paper “Chapter six - mutation testing advances: An analysis and survey.” [1] and “On the Distribution of ‘Simple Stupid Bugs’ in Unit Test Files: An Exploratory Study.”[2].Then I'll be ready to start collecting Java and Python test code. Next, I will read the paper "Test Smell Detection Tools: A Systematic Mapping Study." [3] and select the proper analysis tools. I will also make the weekly report with my mentor professor Iftekhar Ahmed for the progress of the research.

5. Timeline

Table 1: Proposed Timeline

Week 1-2	a. Get familiar with Mutation Analysis and Test Smell Detection b. Read the papers about Mutation Analysis and Test Smell Detection
Week 3-4	a. Find Java and Python unit test codes
Week 5-6	a. Find Mutation Tools and Test smell detectors b. Select proper tools and detectors
Week 7-9	a. Apply tools and detectors to code set
Week 10-11	a. Complete analysis and results. b. Complete a report for the goal of this research that finds whether there is a correlation between mutation score and test smell for unit test code.

6. Itemized Budget

The basic needs for this research projects are:

- Software Components: Licenses for software
- Hardware Components: Solid State Drives and Memory for Desktop
- Others: Articles, Books needs for research

Table 2: Item Price

Item	**Price**
512GB Solid State Drives	$100
16GB Memory	$60
Articles, Books needs for research	$300
Potential purchase: printing	$50
Total	$510

7. Reference

[1] Papadakis M, Kintis M, Zhang J, Jia Y, Traon YL, Harman M. “Chapter six - mutation testing advances: An analysis and survey.” *In Advances in computers, vol. 112. Elsevier,* 2019. p. 275–378

[2] Anthony Peruma and Christian D. Newman. “On the Distribution of ‘Simple Stupid Bugs’ in Unit Test Files: An Exploratory Study.” *In Proceedings of the 18th International Conference on Mining Software Repositories (MSR '21),* 2021

[3] Wajdi Aljedaani, Anthony Peruma, Ahmed Aljohani, Mazen Alotaibi, Mohamed Wiem Mkaouer, Ali Ouni, Christian D. Newman, Abdullatif Ghallab, and Stephanie Ludi. 2021. “Test Smell Detection Tools: A Systematic Mapping Study.” *In EASE '21: The International Conference on Evaluation and Assessment in Software Engineering, June 21–23, 2021*

[4] https://github.com/agroce/universalmutator

[5] https://stryker-mutator.io/

[6] https://github.com/TestSmells/TestSmellDetector/releases

[7] https://github.com/jetbrains-research/pynose