Abstract: On the Use of Commit-Relevant Mutants

Miloš Ojdanić
University of Luxembourg
Luxembourg
milos.ojdanic@uni.lu

Wei Ma
University of Luxembourg
Luxembourg
wei.ma@uni.lu

Thomas Laurent
University College Dublin
Ireland
thomas.laurent@ucdconnect.ie,

Thierry Titcheu Chekam
Société Européenne des Satellites
Luxembourg
thierry.titcheu.chekam@ses.com

Anthony Ventresque
University College Dublin
Ireland
anthony.ventresque@ucd.ie

Mike Papadakis
University of Luxembourg
Luxembourg
michail.papadakis@uni.lu

I. Reference

This work is under consideration concerning publication as an article for the Special issue of the Empirical Engineering (EMSE) journal of ICSME2020 (36^{th} International Conference on Software Maintenance and Evolution).

II. Introduction

Software systems are subject to regular modification during their life-cycle. Modifications are usually made in order to maintain/improve the software while automated testing is used to establish confidence that the modifications did not break functionalities.

In such scenarios, developers often assume that the previous (operational) version of the system was stable and correct. Therefore, they are interested in testing only the behaviour delta of the changes they performed from pre- to post-commit system version. For such cases developers need metrics quantifying the extend to which they have tested the error-prone program behaviours affected by their changes. Unfortunately, little research has been devoted in forming such change-aware test criteria. Change-aware test criteria would offer a viable, from an economic perspective, way of dealing with the continuous software modifications as one would only focus on the particular program changes or commits.

Mutation testing has long been established as one of the strongest test criteria. Testers can use mutants to design strong test cases, likely to be fault revealing and to perform test assessment as it effectively quantifies the test suites' strengths.

However, mutation testing research assumes a static nature of software. This strategy has the unfortunate effect of blindly using all possible mutants without considering their relevance to the task or to the most recent changes in question. To allow such focused testing, one should use only what we call commit-relevant mutants, i.e., mutants interacting with the changed program behaviours. These mutants form the change-relevant requirements and can be used to judge whether test suites are adequate and, if not, to provide guidance in improving test suites (by creating tests that kill commit-relevant mutants).

Our study argues that covering all interactions between unmodified and modified code is particularly important because problematic regression issues arise from such unforeseen interactions. Yet, using the entire set of mutants or those on the modified code leads to an imprecise solution. It either includes a large volume of noise (irrelevant mutants) or is insufficient to cover all possible interactions. Hence we show that the majority of the altered program behaviours are captured by commit-relevant mutants located on unmodified code parts.

Our results reveal that commit-relevant mutants significantly differ from the other classes of mutants and that there is a relatively weak correlation between commit-aware and traditional mutation scores, indicating the need for a commit-relevant test assessment metric. Besides other contributions, perhaps, more importantly, our results demonstrate that commit-relevant mutants have 30% more chances to reveal faults (real faults) than traditional mutation when analyzing the same number of mutants (putting approximately the same amount of effort).

III. Contribution

- The definition of the commit-relevant mutants and the related commit-relevant mutation-based test assessment.
- Demonstration of significant cost-effective guidance to accurately and adequately test particular program changes when targeting commit-relevant mutants over other classes of mutants.
- Demonstration of the fault-revealing potential of commit-relevant mutants.
- A test case prioritization scenario when using commit-aware mutation testing as a metric.

IV. Talk outline

The talk outline will be the following:

- Introducing definition of relevant mutants and commit-aware mutation testing.
- Empirical findings when targeting commit-relevant mutants over different categories of mutants.
- Empirical results of fault detection.
- Application of a test case prioritization scenario when using commit-relevant mutants as a metric.