An Empirical Study of Mutation-Based Test Case Clustering Prioritization and Reduction Technique

Longbo Li, Yanhui Zhou, Yong Yu, Feiyuan Zhao
Southwest University, Chongqing, China

Shenghua Wu and Zhe Yang
Meiyun Zhi Number Technology Co., Ltd.
**Mutation analysis** is also called mutation testing, which is a method for measuring the quality of test suite, using mutants (artificially injected faults) generated from the original program using special rules (mutation operators)[1].

Due to the expansion of the software scale, a large number of test cases are generated in regression testing.

We need to prioritize and reduce test cases during the regression testing.
We usually prioritize and reduce test cases usually using **code coverage criterion**.

1. **Statement coverage**[1]
2. **Branch coverage**[2]

A test case can kill the mutation program, not only cover code information, also need to kill the mutation program.


Clustering algorithm belong to unsupervised learning algorithm. Clustering algorithm can reveal the intrinsic properties and laws of data. Hierarchical clustering algorithm (HCA) to achieve good results based on code coverage information in test case prioritization[1].

We use HCA to explore mutation-based test case prioritization and reduction technique.

1. Every mutation operator can generate many mutants in mutation analysis. If a mutant and original program return different returns for the same test, then the test **kills** the mutant.
2. The priorities of the mutants are the same in previous study[^1][^2][^3].

Mutants have diverse attributes, and each mutant has a different priority.

**Mutation Program Unit ( MPU)** Definition:

We call mutation programs generated by the same mutation operator a mutation program unit. A MPU contains many mutants. The smallest unit in MPU is a mutant. In this paper, we use the smallest mutation program unit.
Why do we use the MPU?

1. A mutation program that is hard to kill by a test case can usually trigger a real fault\(^1\).

2. Because of the large number of mutation programs generated, mutation analysis consumes a lot of computing resources, and the use of high-priority mutation programs can reduce computational overhead.

After the mutation analysis, we first generate the mutation program unit kill matrix, which is then transformed into the mutation program unit priority matrix.
Average Percentage of Fault Detection (APFD)\(^{[1]}\)

\[
APFD = 1 - \frac{TF_1 + \cdots + TF_n}{nm} + \frac{1}{2n}
\]

Average Percentage of Weight Fault Detection (APWFD)

\[
APWFD = 1 - \frac{W_1 \times TF_1 + \cdots + W_n \times TF_n}{\sum_{j=1}^{n} W_i \times n} + \frac{1}{2n}
\]

Hierarchical clustering algorithm can control how many clusters are generated. K represents the number of clusters.
Is the different distance calculation method have different effects on test case prioritization and reduction?
Does our approach have an impact on the capability of fault detection test cases?

<table>
<thead>
<tr>
<th>Program</th>
<th>Analysis index</th>
<th></th>
<th></th>
<th></th>
<th>Analysis index</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>APFD</td>
<td>Cluster time(s)</td>
<td>Test case</td>
<td>APFWD</td>
<td>Cluster time(s)</td>
<td>Test case</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Chart</td>
<td>20.0210</td>
<td>2.0764</td>
<td>5693</td>
<td>18.7334</td>
<td>0.1189</td>
<td>3577</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Closure</td>
<td>123.9185</td>
<td>1649.2548</td>
<td>440296</td>
<td>123.1317</td>
<td>758.1702</td>
<td>182155</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lang</td>
<td>47.6511</td>
<td>10.0078</td>
<td>11338</td>
<td>48.4177</td>
<td>1.5700</td>
<td>6480</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Math</td>
<td>91.4900</td>
<td>165.7274</td>
<td>22688</td>
<td>88.2791</td>
<td>63.9586</td>
<td>9941</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time</td>
<td>25.5815</td>
<td>122.3370</td>
<td>70239</td>
<td>25.2988</td>
<td>71.4638</td>
<td>17935</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average</td>
<td>0.8819</td>
<td>5.5697</td>
<td>-</td>
<td>0.8681</td>
<td>2.5579</td>
<td>-</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
We have proposed an novel definition to Mutation Program Unit.

We have proposed mutation-based test case prioritization and reduction method.

Our method can reduce the number of test cases by 40%, and the loss of fault detection capability is only 1.38%.
We will systematically explain the theoretical mutation program unit priority we propose.

We will consider reducing the complexity of algorithms.

We will explore the impact of our approach on test cases that trigger real faults.
THANKS

lilongbo@email.swu.edu.cn