Assessing the Influence of Multiple Test Case Selection on Mutation Experiments

Marcio E. Delamaro* and Jeff Offutt†

*Computer Systems Department
Universidade de São Paulo, São Carlos, SP, Brazil
Email: delamaro@icmc.usp.br
†Software Engineering
George Mason University, Fairfax, VA, USA
Email: offutt@gmu.edu

Abstract—Mutation testing is widely used in experiments. Some papers experiment with mutation directly, while others use it to introduce faults to measure the effectiveness of tests created by other methods. There is some random variation in the mutation score depending on the specific test values used. When generating tests to use in experiments, a common, although not universal practice, is to generate multiple sets of tests to satisfy the same criterion or according to the same procedure, and then to compute their average performance. Averaging over multiple test sets is thought to reduce the variation in the mutation score. This practice is extremely expensive when tests are generated by hand (as is common) and as the number of programs increase (a current positive trend in software engineering experimentation).

The research reported in this short paper asks a simple and direct question: do we need to generate multiple sets of test cases? That is, how do different test sets influence the cost and effectiveness results? In a controlled experiment, we generated 10 different test cases to be adequate for the Statement Deletion (SSDL) mutation operator for 39 small programs and functions, and then evaluated how they differ in terms of cost and effectiveness. We found that averaging over multiple programs was effective in reducing the variance in the mutation scores introduced by specific tests.

Index Terms—Software testing; Mutation testing; Test set selection

I. INTRODUCTION

When conducting experimental studies with mutation testing, researchers usually need to solve some problems that are seldom automated. This includes identifying equivalent mutants and generating adequate test sets as part of experimental setup.

In a recent paper [1] the authors faced the problem of selecting adequate test sets to evaluate the cost and effectiveness of individual mutation operators. We used the approach pioneered by Frankl [2] of first generating a “universe” of test cases that were adequate for the complete set of mutants, and then selected subsets that were adequate for the specific mutants being studied. The cost and effectiveness of each operator can be measured by assessing some characteristics of the adequate test sets, such as their sizes and their mutation scores when run against the complete set of mutants.

Initially, only one adequate test set was used for each mutation operator. This procedure was criticized because for a given operator Op there can be many different Op-adequate subsets of the universe test set, and thus the results could vary depending on which specific tests are selected. Although intuitively appealing, to the best of our knowledge this concern has never been experimentally verified. So, the question addressed in this paper is whether, and to what extent, the selection of different adequate test sets can influence the experimental results.

The results of this paper indicate two interesting findings. The first is that different test sets can give different results on individual programs. The second is somewhat contradictory, when averaged over the set of 39 programs that we studied, statistically this difference all but disappears. These facts can be useful to future researchers in this area. The second finding indicates that when researchers are interested in averages over programs, one test set is enough. The first, however, brings up a practical question: if the mutation score for the complete set of mutants is sensitive to the specific test cases generated for a reduced set of mutants, will this impact the effectiveness of such test sets against “real world” faults? Moreover, is there a way to identify and select the best (most effective) adequate test sets from among a set of tests that all satisfy the same criterion?

The next section of this short paper presents the experimental setup. Section III presents the results obtained. Section IV presents threats to validity, and Section V presents conclusions and recommendations.

II. EXPERIMENTAL SETUP

This section describes the subject programs, the tool used, and the experimental procedure.

a) Subject Programs: Thirty nine programs written in C of varying sizes and from different domains were used as experimental subjects. These programs were extracted from the Siemens program suite [3], text books [4], and the software testing literature. The subject programs varied in size from one to 20 functions, and from seven to 390 lines of code, totaling 189 functions and 2853 lines of code. When needed, we refer to an individual subject as $P_i$.

Table I summarizes the subject programs. For each, the table shows the number of functions, the number of lines of code, the number of mutants generated by all operators, the number
of equivalent mutants, and the number of tests in the mutation-
adequate test set.

b) Supporting Tool: This study used the C language
mutation tool ProD to generate a comprehensive set of faulty programs that were
used to evaluate the effectiveness of the test sets. This paper
uses the mutation score as a proxy for effectiveness. The
set of mutation operators implemented into Proteum follows,
as closely as possible, the original C operators defined by
Agrawal et al. [6]. The set of all mutants for program \( P_i \) is
referred as \( M_i \).

c) Adequate Test Sets: For each subject program, an
adequate test set was constructed to kill all the mutants
generated by Proteum. We refer to the test sets for program
\( P_i \) as \( T_i \). The size of each \( T_i \) is shown in Table I.

Most of the tests were generated by hand by the first author.
Some test cases were provided with the Siemens programs\(^1\)
and we generated additional tests by hand to kill all non-
equivalent mutants. For the remaining programs, we also
generated tests by hand to kill all non-equivalent mutants.
Equivalent mutants were marked by hand analysis.

After all mutants were killed or marked as equivalent, test
cases that did not contribute to killing at least one mutant were
removed.

d) Data Collection: For each \( P_i \) we selected a reduced
set of mutants \( R_i \). These mutants were generated by a single
mutation operator, Statement Deletion (SSDL). We chose
SSDL for two reasons: (1) it is universally applicable to all
programs; and (2) it has been studied in recent papers as
an alternative to using the complete set of mutation operators [1, 7, 8].

For each \( R_i \) we built 10 test sets that were adequate
for SSDL. \( TR_{i,1} \) through \( TR_{i,10} \). Each \( TR_{i,j} \) was built by
selecting test cases from the universe set, \( T_i \), until an adequate
test set was obtained. Starting with an empty set, a test case
was considered and added if and only if it killed at least one
additional mutant. For the first set, \( TR_{i,1} \), tests were
considered in the same order as they appeared in \( T_i \). For
the remaining \( TR_{i,2} \) through \( TR_{i,10} \) tests were considered
in random order, starting with a new random seed for that
test set.

For each \( TR_{i,j} \) two metrics were collected: its size and
its mutation score when executed against the complete set
of mutants \( M_i \). Then, we collected statistics of distribution
and central tendency for each \( TR_{i,j} \), including mean, median,
minimum and maximum values, and the standard deviation
(SD). These numbers are analyzed in the next section.

e) Research Questions: We have two primary questions
in this experiment:

RQ1: If different SSDL-adequate test sets are selected from
the universe of test cases, how different would they
be in terms of effectiveness, as measured by the
mutation score against the complete set of mutants?

RQ2: If different SSDL-adequate test sets are selected from
the universe of test cases, how different would they
be in terms of cost, as measured by their sizes?

III. RESULTS

Table II shows the effectiveness results for the 39 programs.
Each row corresponds to program \( P_i \), and the columns show
statistics over the 10 SSDL-adequate test sets. For program
\( P_{1} \) (boundedQueue), the lowest mutation score across the
10 test sets was 0.9207 and the highest score was 0.9902.
The average (mean) mutation score of all 10 test sets is 0.9600,
with a standard deviation of 0.0216.

The differences between the lowest and highest mutation
scores of a given program vary a lot among the programs.
For example, if the high and low scores for \( P_{4} \) and \( P_{19} \) differ
by more than 20%, whereas \( P_{28} \) and \( P_{31} \) differ by less than
1%. So, if researchers rely on results for individual programs,
using only one adequate test set might skew the result.
For example, if \( P_{4} \)’s lowest scoring test set was used, SSDL would
be considered to have only .4752 effectiveness. However, if
\( P_{4} \)’s highest scoring test set was used, its effectiveness would
be found to be 0.7723.

Since several of the smallest programs had particularly high
SD, we calculated the Spearman rank correlation [9] between
the LOC and SD and between the LOC and Max-Min
difference. Two series of numbers are perfectly correlated if the
value is 1 or -1, and not at all correlated if the value is 0. The
LOC and SD were (negatively) correlated with a value of -.65,
and the LOC and Max-Min difference with a value of -.63.
Thus, we conclude there is a strong correlation between size
and spread, that is, the differences in effectiveness among
the test sets are smaller with bigger programs. This is significant
because as the programs get bigger, we do not need to use
as many test sets in our experiment. This is good news for
experimentalists, because creating 10 test sets for a 10 line
program is one thing, but creating 10 test sets for a 1000 line
program is quite another!

The last row of Table II shows the averages over the 39
programs. The average Min is .9093 and the average Max
is .9338, so the SD is very small, only .0071. Running
a one-way ANOVA between the 10 sequences of test sets
for the 39 subjects we can see that there were no statistically
significant differences between mutation score means
\( F(9, 380) = 0.312; p = 0.971 \).

The results for the number of test cases in each adequate test
set are shown in Table III. Since each program has different
universes of adequate test cases, with different sizes, we
analyze the ratio between the largest and the smallest adequate
sets for each program, rather than the difference.

The ratio ranges from a low of 1 (on programs \( P_{4}, P_{3},
P_{15}, \) and \( P_{38} \)) all the way to a high of 7 on \( P_{17} \). Not
surprisingly, larger programs usually need more tests. This
time, larger programs tended to have a higher SD over the
10 sets of tests, but Spearman’s correlation between LOC and
SD was only .56. The correlation is positive, indicating that

---

\(^1\)Programs print_tokens, print_tokens2, replace, schedule,
schedule2, tcas and totInfo.
TABLE I

<table>
<thead>
<tr>
<th>Program</th>
<th>Prog. Name</th>
<th>Functions</th>
<th>LOC</th>
<th>Mutants</th>
<th>Equiv.</th>
<th>Test Cases</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>boundedQueue</td>
<td>6</td>
<td>49</td>
<td>1121</td>
<td>99</td>
<td>13</td>
</tr>
<tr>
<td>P2</td>
<td>cal</td>
<td>1</td>
<td>18</td>
<td>891</td>
<td>71</td>
<td>8</td>
</tr>
<tr>
<td>P3</td>
<td>Calculation</td>
<td>7</td>
<td>46</td>
<td>1118</td>
<td>107</td>
<td>13</td>
</tr>
<tr>
<td>P4</td>
<td>checkIt</td>
<td>1</td>
<td>9</td>
<td>104</td>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>P5</td>
<td>CheckPalindrome</td>
<td>1</td>
<td>10</td>
<td>166</td>
<td>20</td>
<td>8</td>
</tr>
<tr>
<td>P6</td>
<td>countPositive</td>
<td>1</td>
<td>9</td>
<td>151</td>
<td>9</td>
<td>5</td>
</tr>
<tr>
<td>P7</td>
<td>date-plus</td>
<td>3</td>
<td>132</td>
<td>2421</td>
<td>160</td>
<td>44</td>
</tr>
<tr>
<td>P8</td>
<td>DigitReverser</td>
<td>1</td>
<td>17</td>
<td>496</td>
<td>43</td>
<td>5</td>
</tr>
<tr>
<td>P9</td>
<td>findLast</td>
<td>1</td>
<td>10</td>
<td>198</td>
<td>17</td>
<td>6</td>
</tr>
<tr>
<td>P10</td>
<td>findVal</td>
<td>1</td>
<td>7</td>
<td>190</td>
<td>18</td>
<td>7</td>
</tr>
<tr>
<td>P11</td>
<td>Gaussian</td>
<td>6</td>
<td>23</td>
<td>1086</td>
<td>19</td>
<td>21</td>
</tr>
<tr>
<td>P12</td>
<td>Heap</td>
<td>7</td>
<td>41</td>
<td>1079</td>
<td>98</td>
<td>8</td>
</tr>
<tr>
<td>P13</td>
<td>InversePermutation</td>
<td>1</td>
<td>15</td>
<td>576</td>
<td>61</td>
<td>12</td>
</tr>
<tr>
<td>P14</td>
<td>jday-jdate</td>
<td>2</td>
<td>49</td>
<td>2821</td>
<td>81</td>
<td>27</td>
</tr>
<tr>
<td>P15</td>
<td>lastZero</td>
<td>1</td>
<td>9</td>
<td>173</td>
<td>9</td>
<td>5</td>
</tr>
<tr>
<td>P16</td>
<td>LRS</td>
<td>5</td>
<td>51</td>
<td>1132</td>
<td>258</td>
<td>8</td>
</tr>
<tr>
<td>P17</td>
<td>MergeSort</td>
<td>3</td>
<td>32</td>
<td>991</td>
<td>48</td>
<td>18</td>
</tr>
<tr>
<td>P18</td>
<td>numZero</td>
<td>1</td>
<td>10</td>
<td>151</td>
<td>17</td>
<td>5</td>
</tr>
<tr>
<td>P19</td>
<td>oddOnPos</td>
<td>1</td>
<td>9</td>
<td>361</td>
<td>71</td>
<td>7</td>
</tr>
<tr>
<td>P20</td>
<td>pcal</td>
<td>8</td>
<td>204</td>
<td>6419</td>
<td>779</td>
<td>49</td>
</tr>
<tr>
<td>P21</td>
<td>power</td>
<td>1</td>
<td>11</td>
<td>268</td>
<td>12</td>
<td>9</td>
</tr>
<tr>
<td>P22</td>
<td>print_tokens</td>
<td>17</td>
<td>349</td>
<td>4322</td>
<td>542</td>
<td>34</td>
</tr>
<tr>
<td>P23</td>
<td>print_tokens2</td>
<td>18</td>
<td>275</td>
<td>4734</td>
<td>664</td>
<td>27</td>
</tr>
<tr>
<td>P24</td>
<td>printPrimes</td>
<td>2</td>
<td>35</td>
<td>715</td>
<td>64</td>
<td>7</td>
</tr>
<tr>
<td>P25</td>
<td>Queue</td>
<td>6</td>
<td>64</td>
<td>460</td>
<td>25</td>
<td>12</td>
</tr>
<tr>
<td>P26</td>
<td>quicksort</td>
<td>1</td>
<td>23</td>
<td>1026</td>
<td>82</td>
<td>13</td>
</tr>
<tr>
<td>P27</td>
<td>RecursiveSort</td>
<td>1</td>
<td>17</td>
<td>555</td>
<td>45</td>
<td>8</td>
</tr>
<tr>
<td>P28</td>
<td>replace</td>
<td>20</td>
<td>390</td>
<td>11,100</td>
<td>2062</td>
<td>142</td>
</tr>
<tr>
<td>P29</td>
<td>schedule</td>
<td>18</td>
<td>213</td>
<td>2108</td>
<td>221</td>
<td>45</td>
</tr>
<tr>
<td>P30</td>
<td>schedule2</td>
<td>16</td>
<td>195</td>
<td>2626</td>
<td>411</td>
<td>41</td>
</tr>
<tr>
<td>P31</td>
<td>Stack</td>
<td>6</td>
<td>56</td>
<td>460</td>
<td>49</td>
<td>11</td>
</tr>
<tr>
<td>P32</td>
<td>stats</td>
<td>1</td>
<td>19</td>
<td>884</td>
<td>101</td>
<td>7</td>
</tr>
<tr>
<td>P33</td>
<td>sum</td>
<td>1</td>
<td>7</td>
<td>165</td>
<td>11</td>
<td>6</td>
</tr>
<tr>
<td>P34</td>
<td>tcas</td>
<td>8</td>
<td>63</td>
<td>2384</td>
<td>428</td>
<td>62</td>
</tr>
<tr>
<td>P35</td>
<td>testPad</td>
<td>1</td>
<td>24</td>
<td>629</td>
<td>57</td>
<td>14</td>
</tr>
<tr>
<td>P36</td>
<td>totInfo</td>
<td>7</td>
<td>214</td>
<td>6693</td>
<td>678</td>
<td>49</td>
</tr>
<tr>
<td>P37</td>
<td>trashAndTakeOut</td>
<td>2</td>
<td>19</td>
<td>599</td>
<td>26</td>
<td>12</td>
</tr>
<tr>
<td>P38</td>
<td>twoPred</td>
<td>1</td>
<td>10</td>
<td>246</td>
<td>24</td>
<td>10</td>
</tr>
<tr>
<td>P39</td>
<td>UnixCal</td>
<td>4</td>
<td>119</td>
<td>4852</td>
<td>339</td>
<td>27</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td></td>
<td>189</td>
<td>2853</td>
<td>66480</td>
<td>7829</td>
</tr>
<tr>
<td>Min</td>
<td></td>
<td></td>
<td>1</td>
<td>7</td>
<td>104</td>
<td>3</td>
</tr>
<tr>
<td>Max</td>
<td></td>
<td></td>
<td>20</td>
<td>390</td>
<td>11100</td>
<td>2062</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td>4.85</td>
<td>73.15</td>
<td>1704.62</td>
<td>200.74</td>
</tr>
</tbody>
</table>

The spread grows as the number of tests grows. Again, this is not surprising and is possibly purely a function of size.

The ratio between the largest and smallest test set sizes, however, only has a Spearman’s correlation value of -.28 with LOC. Thus, we conclude that differences in test set size is not significantly correlated with program size.

Considering only the averages for the 39 programs, as we did with the mutation score, the smallest mean is 4.79 and the largest is 6.00. Running a one-way ANOVA, we can see that there were no statistically significant differences between test set sizes means ($F(9, 380) = 0.222; p = 0.991$).

As with the mutation scores, there is very little difference between the minimum and maximum number of test cases when averaged over all 39 programs.

IV. THREATS TO VALIDITY & LIMITATIONS

A usual threat in this kind of experimental work is the fact that, no matter which programs we use in the experiment, the results can never be generalized to all programs. To deal with this problem we tried to select a large number of programs from different sources and in different domains. Also in this sense, the small sizes of most programs may represent a threat, but analyzing a large number of large programs is practically impossible.

The creation of the universe of test cases, adequate to all the C mutants of each programs was done manually. Using different tests or a different tester could result in different results. However the construction of such sets is extremely time consuming, so it would be impractical to create more than one test set. It is the same for the identification of equivalent mutants. It was done by the first author and, besides his experience with mutation testing and the C language, no other measure was taken to guarantee the quality of such process.

Finally, we used SSDL as the operator for test case selection. Other operators, and indeed, any other test criterion could yield different results. The important issue here is not the actual values obtained, but the analysis of the methodological procedures taken. Thus we see no reason why using other
This paper evaluates the effect of using multiple test cases in experimental research. Previous researchers have assumed that selecting only one adequate test set could interfere in the results of cost and effectiveness for mutation operators, and thus created multiple test sets. However, this assumption was made without evidence.

Our results show that there can be significant differences for individual subject programs among different test sets chosen for the same adequacy criterion. These differences were observed for both effectiveness (mutation score) and cost (number of tests). This result makes a case for choosing multiple test sets during experimentation.

However, we found that the differences in effectiveness among different test sets was less with larger programs. Perhaps more importantly, the differences tended to disappear when effectiveness is averaged over a collection of programs (39 in our study). This is not surprising since the difference in the results of the test sets for each program is probably due to the chances of selecting specific test case values. If we had insignificant differences among the 10 test set for any single subject, then that would indicate the practice of using multiple sets is not necessary. The fact that a large number of subjects reduces the individual errors is not particularly related to mutation. It is only a good experimental practice.

Thus, we recommend using multiple test sets if only a few subjects are chosen, but if many subjects are used, using multiple test sets may not increase the accuracy of the experimental work. It is important to note that the lack of statistical significance does not mean that the averages do not differ. A two percentage points difference in the averages, as found in this experiment, may or may not be significant, depending how the researcher needs to use it. In addition, we used an “unweighted average” in which each program is weighted equally in the final mean. If other statistics are used, for instance a weighted mean, the results may differ slightly. The weighted mean of the effectiveness can be computed by summing up the total number of killed mutants on all 39 programs and dividing that sum by the total number of non
equivalent mutants.

We did not vary the number of tests; we always used 10. So we are not able to provide insight as to how many test sets are “enough.”

Analyzing these results raises an interesting question that might be of interest for researchers and practitioners. What are the consequences of choosing a given test case in terms of effectiveness, considering not only mutation operators but also real faults? If a tester is presented with two test sets adequate to a set of mutants (or to any other testing criterion), which should she/he choose? Is there a way to predict which should be more effective in terms of a larger set of faults, either those defined by mutant operators or real faults? In a quick check we could verify, for instance, that there is no strong correlation between the size of the SSDL-adequate test sets and their effectiveness against the complete set of mutants.

ACKNOWLEDGMENTS

Prof. Marcio Delamaro’s research is supported by FAPESP (Fundação de Amparo a Pesquisa do Estado de São Paulo), process number 2012/16950-5.

REFERENCES


