Are Mutants a Valid Substitute for Real Faults in Software Testing?

René Just*, Dariush Jalali*, Laura Inozemtseva†, Michael D. Ernst*, Reid Holmes†, Gordon Fraser‡

*University of Washington
†University of Waterloo
‡University of Sheffield

November 20, 2014
How good is my test suite?

A good test suite detects real faults

Test quality metric is necessary in many areas:

- Test generation, minimization, prioritization, ...
How good is my test suite?

A good test suite detects real faults

Test quality metric is necessary in many areas:

- Test generation, minimization, prioritization, ...

Problem: Set of real faults is unknowable

Solution: Use a proxy metric for test quality

- Code coverage ratio
- Mutant detection rate
How good is my test suite?

A good test suite detects real faults

Test quality metric is necessary in many areas:
  ▶ Test generation, minimization, prioritization, ...

Problem: Set of real faults is unknowable

Solution: Use a proxy metric for test quality
  ▶ Code coverage ratio
  ▶ Mutant detection rate

Mutant detection rate $\approx$ Real fault detection rate?
Mutation analysis: Overview

Program

Test suite
Mutation analysis: Overview

Program

Generate mutants

Mutants

Test suite
**Mutation analysis: Overview**

Each mutant contains one small syntactic change.
Mutation analysis: Overview

```java
public float avg(float[] data) {
    float sum = 0;
    for (float num : data) {
        sum += num;
    }
    return sum / data.length;
}
```
Mutation analysis: Overview

```
public float avg(float[] data) {
    float sum = 0;
    for (float num : data) {
        sum += num;
    }
    return sum / data.length;
}
```
Mutation analysis: Overview

Program

Generate mutants

Mutants

Test suite

Execute test suite

Mutant detection rate
Mutation analysis: How it is used

- Design new testing approach (generation, minimization, ...)
- Compare mutant detection rate to previous work
- Higher?
  - no
Mutation analysis: How it is used

1. Design new testing approach (generation, minimization, ...)
2. Compare **mutant detection rate** to previous work
3. Higher?
   - **no**
   - **yes**
     - Claim approach is better for real faults
     - Publish paper
Mutation analysis: How it is used*

Design new testing approach (generation, minimization, ...)

Compare mutant detection rate to previous work

Higher?

no

yes

Claim approach is better for real faults

Publish paper

*in hundreds of papers
Mutation analysis: How it is used*

- Design new testing approach (generation, minimization, ...)
- Compare mutant detection rate

Mutant detection rate \(\approx\) Real fault detection rate?

- no Higher?  
- yes Claim approach is better for real faults

*in hundreds of papers

Claim approach is better for real faults

Publish paper
Related work

ISSTA’96\(^1\)  ICSE’05\(^2\)  FSE’14

\(^1\) Daran and Thévenod-Fosse, *ISSTA’96*.

\(^2\) Andrews et al., *ICSE’05*.
**Related work**

<table>
<thead>
<tr>
<th>ISSTA’96&lt;sup&gt;1&lt;/sup&gt;</th>
<th>ICSE’05&lt;sup&gt;2&lt;/sup&gt;</th>
<th>FSE’14</th>
</tr>
</thead>
<tbody>
<tr>
<td>KLOC</td>
<td>1</td>
<td>6</td>
</tr>
</tbody>
</table>

---

<sup>1</sup> Daran and Thévenod-Fosse, *ISSTA’96*.

<sup>2</sup> Andrews et al., *ICSE’05*.
## Related work

<table>
<thead>
<tr>
<th></th>
<th>ISSTA’96(^1)</th>
<th>ICSE’05(^2)</th>
<th>FSE’14</th>
</tr>
</thead>
<tbody>
<tr>
<td>KLOC</td>
<td>1</td>
<td>6</td>
<td>321</td>
</tr>
<tr>
<td>Faults</td>
<td>12</td>
<td>38</td>
<td>357</td>
</tr>
</tbody>
</table>

\(^1\) Daran and Thévenod-Fosse, *ISSTA’96*.  
\(^2\) Andrews et al., *ICSE’05*.  
## Related work

<table>
<thead>
<tr>
<th></th>
<th>ISSTA’96¹</th>
<th>ICSE’05²</th>
<th>FSE’14</th>
</tr>
</thead>
<tbody>
<tr>
<td>KLOC</td>
<td>1</td>
<td>6</td>
<td>321</td>
</tr>
<tr>
<td>Faults</td>
<td>12</td>
<td>38</td>
<td>357</td>
</tr>
<tr>
<td>Mutants</td>
<td>24</td>
<td>1,100</td>
<td>230,000</td>
</tr>
</tbody>
</table>

¹ Daran and Thévenod-Fosse, *ISSTA’96.*
² Andrews et al., *ICSE’05.*
## Related work

<table>
<thead>
<tr>
<th></th>
<th>ISSTA’96(^1)</th>
<th>ICSE’05(^2)</th>
<th>FSE’14</th>
</tr>
</thead>
<tbody>
<tr>
<td>KLOC</td>
<td>1</td>
<td>6</td>
<td>321</td>
</tr>
<tr>
<td>Faults</td>
<td>12</td>
<td>38</td>
<td>357</td>
</tr>
<tr>
<td>Mutants</td>
<td>24</td>
<td>1,100</td>
<td>230,000</td>
</tr>
<tr>
<td>Tests</td>
<td>generated</td>
<td>generated</td>
<td>generated &amp; developer-written</td>
</tr>
</tbody>
</table>

\(^1\) Daran and Thévenod-Fosse, *ISSTA’96.*  
\(^2\) Andrews et al., *ICSE’05.*
### Related work

<table>
<thead>
<tr>
<th></th>
<th>ISSTA’96(^1)</th>
<th>ICSE’05(^2)</th>
<th>FSE’14</th>
</tr>
</thead>
<tbody>
<tr>
<td>KLOC</td>
<td>1</td>
<td>6</td>
<td>321</td>
</tr>
<tr>
<td>Faults</td>
<td>12</td>
<td>38</td>
<td>357</td>
</tr>
<tr>
<td>Mutants</td>
<td>24</td>
<td>1,100</td>
<td>230,000</td>
</tr>
<tr>
<td>Tests generated</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
</tbody>
</table>

- Effect of code coverage considered
- Qualitative study of real faults

\(^1\) Daran and Thévenod-Fosse, *ISSTA’96.*

\(^2\) Andrews et al., *ICSE’05.*
Are mutants a valid substitute for real faults?

Research Questions

1. Do stronger test suites detect more mutants?
2. What types of real faults are not represented by mutants?
3. Is mutant detection correlated with fault detection?
Methodology: Overview

Test suites ➔ Real fault detection rates ➔ Compare results

Test suites ➔ Mutant detection rates ➔ Compare results
Methodology: Overview

- Test suites
- Real faults
  - Real fault detection rates
- Mutant detection rates
- Mutants

Compare results
Methodology: Overview

Test suites

Real faults

Real fault detection rates

Mutant detection rates

Mutants

Compare results

René Just, UW CSE Are Mutants a Valid Substitute for Real Faults in Software Testing?
Reproducible and isolated real faults

Source code $V_{\text{bug}}$  
Buggy version

Source code $V_{\text{fix}}$  
Fixed version
Reproducible and isolated real faults

Bug fix only

Source code $V_{bug}$

Source code $V_{fix}$
Reproducible and isolated real faults
Real faults from version control history
Real faults from version control history

Source code $V_{n-1}$

Source code $V_n$

Labeled as bug fix
Real faults from version control history

Candidate version pair

- Source code $V_{n-1}$
- Source code $V_n$

Labeled as bug fix

Commit

Time
Real faults from version control history

Candidate version pair

commit

commit

Source code $V_{n-1}$

Source code $V_n$

Bug fix only

time

Labeled as bug fix

René Just, UW CSE

Are Mutants a Valid Substitute for Real Faults in Software Testing?
Real faults from version control history

Candidate version pair

Source code $V_{n-1}$

Source code $V_n$

Test

commit

commit

Labeled as bug fix

René Just, UW CSE

Are Mutants a Valid Substitute for Real Faults in Software Testing?
Subject programs

5 open source Java programs

- Different application domains
- Version control and bug tracking systems
- Comprehensive test suites

<table>
<thead>
<tr>
<th>Program</th>
<th>KLOC</th>
<th>Test KLOC</th>
<th>Tests</th>
</tr>
</thead>
<tbody>
<tr>
<td>JFreeChart</td>
<td>96</td>
<td>50</td>
<td>2,205</td>
</tr>
<tr>
<td>Closure Compiler</td>
<td>90</td>
<td>83</td>
<td>7,927</td>
</tr>
<tr>
<td>Commons Math</td>
<td>85</td>
<td>19</td>
<td>3,602</td>
</tr>
<tr>
<td>Joda Time</td>
<td>28</td>
<td>53</td>
<td>4,130</td>
</tr>
<tr>
<td>Commons Lang</td>
<td>22</td>
<td>6</td>
<td>2,245</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>321</strong></td>
<td><strong>211</strong></td>
<td><strong>20,109</strong></td>
</tr>
</tbody>
</table>
Real faults

357 reproducible and isolated real faults

<table>
<thead>
<tr>
<th>Candidates</th>
<th>Compilable</th>
<th>Reproducible</th>
<th>Isolated</th>
</tr>
</thead>
<tbody>
<tr>
<td>JFreeChart</td>
<td>80</td>
<td>62</td>
<td>28</td>
</tr>
<tr>
<td>Closure Compiler</td>
<td>316</td>
<td>227</td>
<td>179</td>
</tr>
<tr>
<td>Commons Math</td>
<td>435</td>
<td>304</td>
<td>132</td>
</tr>
<tr>
<td>Joda Time</td>
<td>75</td>
<td>57</td>
<td>29</td>
</tr>
<tr>
<td>Commons Lang</td>
<td>273</td>
<td>186</td>
<td>69</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>1,179</strong></td>
<td><strong>836</strong></td>
<td><strong>437</strong></td>
</tr>
</tbody>
</table>
Real faults

357 reproducible and isolated real faults

<table>
<thead>
<tr>
<th></th>
<th>Candidates</th>
<th>Compilable</th>
<th>Reproducible</th>
<th>Isolated</th>
</tr>
</thead>
<tbody>
<tr>
<td>JFreeChart</td>
<td>80</td>
<td>62</td>
<td>28</td>
<td>26</td>
</tr>
<tr>
<td>Closure Compiler</td>
<td>316</td>
<td>227</td>
<td>179</td>
<td>133</td>
</tr>
<tr>
<td>Commons Math</td>
<td>435</td>
<td>304</td>
<td>132</td>
<td>106</td>
</tr>
<tr>
<td>Joda Time</td>
<td>75</td>
<td>57</td>
<td>29</td>
<td>27</td>
</tr>
<tr>
<td>Commons Lang</td>
<td>273</td>
<td>186</td>
<td>69</td>
<td>65</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>1,179</strong></td>
<td><strong>836</strong></td>
<td><strong>437</strong></td>
<td><strong>357</strong></td>
</tr>
</tbody>
</table>
Real faults

357 reproducible and isolated real faults

<table>
<thead>
<tr>
<th>Candidates</th>
<th>Compilable</th>
<th>Reproducible</th>
<th>Isolated</th>
</tr>
</thead>
<tbody>
<tr>
<td>JFreeChart</td>
<td>80</td>
<td>62</td>
<td>28</td>
</tr>
<tr>
<td>Closure Compiler</td>
<td>316</td>
<td>227</td>
<td>179</td>
</tr>
<tr>
<td>Commons Math</td>
<td>435</td>
<td>304</td>
<td>132</td>
</tr>
<tr>
<td>Joda Time</td>
<td>75</td>
<td>57</td>
<td>29</td>
</tr>
<tr>
<td>Commons Lang</td>
<td>273</td>
<td>186</td>
<td>69</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>1,179</strong></td>
<td><strong>836</strong></td>
<td><strong>437</strong></td>
</tr>
</tbody>
</table>
Real faults

357 reproducible and isolated real faults

<table>
<thead>
<tr>
<th>Library</th>
<th>Candidates</th>
<th>Compilable</th>
<th>Reproducible</th>
<th>Isolated</th>
</tr>
</thead>
<tbody>
<tr>
<td>JFreeChart</td>
<td>80</td>
<td>62</td>
<td>28</td>
<td>26</td>
</tr>
<tr>
<td>Closure Compiler</td>
<td>316</td>
<td>227</td>
<td>179</td>
<td>133</td>
</tr>
<tr>
<td>Commons Math</td>
<td>435</td>
<td>304</td>
<td>132</td>
<td>106</td>
</tr>
<tr>
<td>Joda Time</td>
<td>75</td>
<td>57</td>
<td>29</td>
<td>27</td>
</tr>
<tr>
<td>Commons Lang</td>
<td>273</td>
<td>186</td>
<td>69</td>
<td>65</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>1,179</strong></td>
<td><strong>836</strong></td>
<td><strong>437</strong></td>
<td><strong>357</strong></td>
</tr>
</tbody>
</table>
Real faults

357 reproducible and isolated real faults

Candidate version pair

commit

Source code $V_{n-1}$

commit

Source code $V_n$

time

~1 person year

no false positives

<table>
<thead>
<tr>
<th>Candidates</th>
<th>Compilable</th>
<th>Reproducible</th>
<th>Isolated</th>
</tr>
</thead>
<tbody>
<tr>
<td>JFreeChart</td>
<td>80</td>
<td>62</td>
<td>28</td>
</tr>
<tr>
<td>Closure Compiler</td>
<td>316</td>
<td>227</td>
<td>179</td>
</tr>
<tr>
<td>Commons Math</td>
<td>435</td>
<td>304</td>
<td>132</td>
</tr>
<tr>
<td>Joda Time</td>
<td>75</td>
<td>57</td>
<td>29</td>
</tr>
<tr>
<td>Commons Lang</td>
<td>273</td>
<td>186</td>
<td>69</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>1,179</strong></td>
<td><strong>836</strong></td>
<td><strong>437</strong></td>
</tr>
</tbody>
</table>
Methodology: Overview

- Test suites
- Real faults
  - Real fault detection rates
- Mutant detection rates
- Mutants
  - Compare results
Mutants

230,000 mutants generated by Major mutation framework

Mutation operators\textsuperscript{1,2}

- Replace operators
- Replace literals
- Delete statements
- Modify branch conditions

\textsuperscript{1}Namin et al., \textit{ICSE’08}.
\textsuperscript{2}Jia and Harman, \textit{TSE’11}.
Methodology: Overview

Test suites

- Real faults
- Real fault detection rates
- Mutant detection rates
- Mutants

Compare results
Developer-written test suites

Obtaining related test suites $T_{bug}$ and $T_{fix}$

- Source code $V_{bug}$
- Test suite $T_{bug}$
- Source code $V_{fix}$
- Test suite $T_{fix}$
Developer-written test suites

Obtaining related test suites $T_{\text{bug}}$ and $T_{\text{fix}}$

Source code $V_{\text{bug}}$

Test suite $T_{\text{bug}}$

Source code $V_{\text{fix}}$

Test suite $T_{\text{fix}}$

Triggering test only
Developer-written test suites

Obtaining related test suites $T_{\text{bug}}$ and $T_{\text{fix}}$

Source code $V_{\text{bug}}$ → Test suite $T_{\text{bug}}$
Source code $V_{\text{fix}}$ → Test suite $T_{\text{fix}}$

Triggering test only
Developer-written test suites

Obtaining related test suites $T_{\text{bug}}$ and $T_{\text{fix}}$

Source code $V_{\text{bug}}$

Source code $V_{\text{fix}}$

Test suite $T_{n-1}$

Test suite $T_n$

We cannot directly use $T_{n-1}$ and $T_n$ from version control

- $T_{n-1}$ and $T_n$ might include failing tests
- $T_n$ might include additional tests (unrelated to the fault)
Developer-written test suites

Obtaining related test suites $T_{\text{bug}}$ and $T_{\text{fix}}$

Source code $V_{\text{bug}}$

Source code $V_{\text{fix}}$

Test suite $T_{n-1}$

Test suite $T_{\text{fix}}$

Test suite $T_n$

Remove failing tests

We cannot directly use $T_{n-1}$ and $T_n$ from version control

- $T_{n-1}$ and $T_n$ might include failing tests
- $T_n$ might include additional tests (unrelated to the fault)
Developer-written test suites

Obtaining related test suites $T_{\text{bug}}$ and $T_{\text{fix}}$

We cannot directly use $T_{n-1}$ and $T_{n}$ from version control
- $T_{n-1}$ and $T_{n}$ might include failing tests
- $T_{n}$ might include additional tests (unrelated to the fault)
Developer-written test suites

Obtaining related test suites $T_{\text{bug}}$ and $T_{\text{fix}}$

Source code $V_{\text{bug}}$

Test suite $T_{\text{bug}}$

Source code $V_{\text{fix}}$

Test suite $T_{\text{fix}}$

Triggering test only
Automatically-generated test suites

EvoSuite, Randoop, and JCrasher
  ▶ Multiple configurations and test objectives

Workflow
  1. Generate tests for fixed program version
  2. Automatically remove failing tests
Test suites: Summary

Developer-written test suites

- Related test suite pairs $T_{bug}$ and $T_{fix}$
- Average statement coverage of $T_{bug}$: 90%

Automatically-generated test suites

- 35,141 test suites
- Average statement coverage: 55%
Methodology: Overview

- Test suites
- Real faults
  - Real fault detection rates
- Mutant detection rates
- Mutants
  - Compare results
Evaluation: Overview

Research Questions

1. Do stronger test suites detect more mutants?
2. What types of real faults are not represented by mutants?
3. Is mutant detection correlated with fault detection?
RQ1: Do stronger test suites detect more mutants?

Setup

- Developer-written test suite pairs $T_{bug}$ and $T_{fix}$
- Does $T_{fix}$ have a higher mutant detection rate than $T_{bug}$?
RQ1: Do stronger test suites detect more mutants?

Setup
- Developer-written test suite pairs $T_{bug}$ and $T_{fix}$
- Does $T_{fix}$ have a higher mutant detection rate than $T_{bug}$?

Results
- Mutant detection rate increased for 73% of faults
RQ1: Do stronger test suites detect more mutants?

Comparison to code coverage

**Mutant detection**
- Increased: 73%
- Unchanged: 27%

**Branch coverage**
- Increased: 50%
- Unchanged: 50%

**Statement coverage**
- Increased: 60%
- Unchanged: 40%
RQ2: What types of faults are not represented by mutants?

Setup

- Qualitative study for 27% of faults
- Weakness or general limitation?

[Pie chart showing 27% and 73%]

- Mutant detection rate increased
- Mutant detection rate unchanged
RQ2: What types of faults are not represented by mutants?

Setup
- Qualitative study for 27% of faults
- Weakness or general limitation?

Results
- Mutant detection rate increased (73%)
- Weak or missing mutation operator (10%)
- No such mutation operator (17%)
RQ2: What types of faults are not represented by mutants?

Setup
- Qualitative study for 27% of faults
- Weakness or general limitation?

Results
- Mutant detection rate increased
- Weak or missing mutation operator
- No such mutation operator

Buggy version
```
switch (x) {
    case 1: 
    ...
    case 2: 
    ...
```

Fixed version
```
switch (x) {
    case 1: 
    ...
    return false;
    case 2: 
    ...
```
RQ2: What types of faults are not represented by mutants?

**Setup**
- Qualitative study for 27% of faults
- Weakness or general limitation?

**Results**

<table>
<thead>
<tr>
<th>Mutant detection rate increased</th>
<th>Weak or missing mutation operator</th>
<th>No such mutation operator</th>
</tr>
</thead>
<tbody>
<tr>
<td>73%</td>
<td>17%</td>
<td>10%</td>
</tr>
</tbody>
</table>

**Buggy version**
```
switch (x) {
  case 1:
  ...
  case 2:
  ...
```

**Fixed version**
```
switch (x) {
  case 1:
  ...
  case 2:
  return false;
  ...
```
RQ2: What types of faults are not represented by mutants?

Setup

- Qualitative study for 27% of faults
- Weakness or general limitation?

Results

- Mutant detection rate increased
- Weak or missing mutation operator
- No such mutation operator

Buggy version

```java
if (isNumZero) {
    return INF;
}
return NaN;
```

Fixed version

```java
return NaN;
```
RQ2: What types of faults are not represented by mutants?

Setup
- Qualitative study for 27% of faults
- Weakness or general limitation?

Results

- Mutant detection rate increased: 73%
- Weak or missing mutation operator: 10%
- No such mutation operator: 17%

Mutation operator: Insert ???

Buggy version

```java
... 
if (isNumZero) {
    return INF;
}
return NaN;
... 
```

Fixed version

```java
... 
return NaN;
... 
```
RQ3: Is mutant detection correlated with fault detection?

Setup

- 35,141 automatically-generated test suites
- How well does mutant detection predict fault detection?
RQ3: Is mutant detection correlated with fault detection?

Setup

- 35,141 automatically-generated test suites
- How well does mutant detection predict fault detection?

Results

![Chart: Mutant detection vs. fault detection and Statement coverage vs. fault detection](image)

René Just, UW CSE
Are Mutants a Valid Substitute for Real Faults in Software Testing?
RQ3: Is mutant detection correlated with fault detection?

Setup
- 35,141 automatically-generated test suites
- How well does mutant detection predict fault detection?

Results

![Chart showing comparison between mutant detection vs. fault detection and statement coverage vs. fault detection. The chart indicates a correlation between the two measurements.]
RQ3: Is mutant detection correlated with fault detection?

Setup
- 35,141 automatically-generated test suites
- How well does mutant detection predict fault detection?

Results

![Chart: Mutant detection vs. fault detection and Statement coverage vs. fault detection](chart.png)
RQ3: Is mutant detection correlated with fault detection?

Setup

- 35,141 automatically-generated test suites
- How well does mutant detection predict fault detection?

Results

![Chart showing mutant detection vs. fault detection and statement coverage vs. fault detection]
Mutants are a valid substitute for most real faults

Mutant detection is positively correlated with fault detection

Mutation-based test generation is promising

Don't use code coverage for test suite minimization: You might miss up to 60% of real faults!

17% of faults cannot be represented by any mutants

Mutation results do not generalize to those faults

http://defects4j.org  http://mutation-testing.org
**Mutants are a valid substitute for most real faults**

**Mutant detection is positively correlated with fault detection**

![Chart: Mutant detection vs. fault detection](chart1.png)

- Mutant detection vs. fault detection
- Statement coverage vs. fault detection

**Mutation-based test generation is promising**

**Mutant detection is more sensitive to faults than coverage**

Don’t use code coverage for test suite minimization:
You might miss up to 60% of real faults!
Mutants are a valid substitute for most real faults

Mutant detection is positively correlated with fault detection

- Mutant detection vs. fault detection
- Statement coverage vs. fault detection

Mutation-based test generation is promising

Mutant detection is more sensitive to faults than coverage

Don’t use code coverage for test suite minimization:
You might miss up to 60% of real faults!

17% of faults cannot be represented by any mutants

Mutation results do not generalize to those faults

http://defects4j.org  http://mutation-testing.org