Mutation testing

Program
Mutation testing: mutant generation

Program

\[ \text{lhs} < \text{rhs} \rightarrow \text{lhs} \leq \text{rhs} \]

\[ \text{lhs} < \text{rhs} \rightarrow \text{lhs} \neq \text{rhs} \]

\[ \text{stmt} \rightarrow \text{no-op} \]
Mutation testing: mutant generation

Program

Mutants

\[ lhs < rhs \quad \rightarrow \quad lhs \leq rhs \]

\[ lhs < rhs \quad \rightarrow \quad lhs \neq rhs \]

\[ stmt \quad \rightarrow \quad no-op \]
Mutation testing: mutant generation

Program

Mutants

\[ \text{lhs} < \text{rhs} \quad \rightarrow \quad \text{lhs} \leq \text{rhs} \]

\[ \text{lhs} < \text{rhs} \quad \rightarrow \quad \text{lhs} \neq \text{rhs} \]

\[ \text{stmt} \quad \rightarrow \quad \text{no-op} \]
Mutation testing: mutant generation

Program

Mutants

\[ lhs < rhs \quad \rightarrow \quad lhs \leq rhs \]
\[ lhs < rhs \quad \rightarrow \quad lhs \neq rhs \]
\[ stmt \quad \rightarrow \quad no-op \]
Mutation testing

Program

Mutants

Tests

JUnit  JUnit
JUnit  JUnit
JUnit  JUnit
JUnit  JUnit
How expensive is mutation testing?

PollEv.com/renejust859
Mutation testing

- **Generating** mutants is **cheap**
- **Executing tests** on mutants is **cheap(ish)**
- **Writing tests** is **expensive**
Mutation testing

- Generating mutants is cheap
- Executing tests on mutants is cheap(ish)
- Writing tests is expensive

Is the mutation score meaningful?
Mutation testing

- Generating mutants is cheap
- Executing tests on mutants is cheap(ish)
- Writing tests is expensive

Engineers care about mutants that elicit effective tests!
Few mutants are effective
Few mutants are effective

A mutant is not effective if:
- it cannot be detected (semantically equivalent)
A mutant is not effective if:

- it cannot be detected (semantically equivalent)
- it fails for any given test (trivial)
A mutant is not effective if:

- it cannot be detected (semantically equivalent)
- it fails for any given test (trivial)
- it is dominated by other mutants (subsumed)
## Mutant subsumption

<table>
<thead>
<tr>
<th>Mutant</th>
<th>MutOp</th>
<th>Ctx</th>
<th>Tests</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m_1$: $\leftarrow \rightarrow$ !=</td>
<td>for</td>
<td>$t_1$</td>
<td><img src="#" alt="Blue Circle" /></td>
</tr>
<tr>
<td>$m_2$: $\leftarrow \rightarrow$ ==</td>
<td>for</td>
<td>$t_2$</td>
<td><img src="#" alt="Red Circle" /></td>
</tr>
<tr>
<td>$m_3$: $\leftarrow \rightarrow$ &lt;=</td>
<td>for</td>
<td>$t_3$</td>
<td><img src="#" alt="Red Circle" /></td>
</tr>
<tr>
<td>$m_4$: $\leftarrow \rightarrow$ &gt;</td>
<td>for</td>
<td>$t_4$</td>
<td><img src="#" alt="Blue Circle" /></td>
</tr>
<tr>
<td>$m_5$: $\leftarrow \rightarrow$ &gt;=</td>
<td>for</td>
<td>$t_1$</td>
<td><img src="#" alt="Blue Circle" /></td>
</tr>
<tr>
<td>$m_6$: $\leftarrow \rightarrow$ true</td>
<td>for</td>
<td>$t_2$</td>
<td><img src="#" alt="Red Circle" /></td>
</tr>
<tr>
<td>$m_7$: $\leftarrow \rightarrow$ false</td>
<td>for</td>
<td>$t_3$</td>
<td><img src="#" alt="Blue Circle" /></td>
</tr>
<tr>
<td>$m_8$: $\leftarrow \rightarrow$ !=</td>
<td>if</td>
<td>$t_4$</td>
<td><img src="#" alt="Red Circle" /></td>
</tr>
<tr>
<td>$m_9$: $\leftarrow \rightarrow$ ==</td>
<td>if</td>
<td>$t_1$</td>
<td><img src="#" alt="Red Circle" /></td>
</tr>
<tr>
<td>$m_{10}$: $\leftarrow \rightarrow$ &lt;=</td>
<td>if</td>
<td>$t_2$</td>
<td><img src="#" alt="Blue Circle" /></td>
</tr>
<tr>
<td>$m_{11}$: $\leftarrow \rightarrow$ &gt;</td>
<td>if</td>
<td>$t_3$</td>
<td><img src="#" alt="Blue Circle" /></td>
</tr>
<tr>
<td>$m_{12}$: $\leftarrow \rightarrow$ &gt;=</td>
<td>if</td>
<td>$t_4$</td>
<td><img src="#" alt="Blue Circle" /></td>
</tr>
<tr>
<td>$m_{13}$: $\leftarrow \rightarrow$ true</td>
<td>if</td>
<td>$t_1$</td>
<td><img src="#" alt="Red Circle" /></td>
</tr>
<tr>
<td>$m_{14}$: $\leftarrow \rightarrow$ false</td>
<td>if</td>
<td>$t_2$</td>
<td><img src="#" alt="Blue Circle" /></td>
</tr>
</tbody>
</table>
Mutation testing at Google

```c
int RunMe(int a, int b) {
    if (a == b || b == 1) {
        Changing this 1 line to
        if (a != b || b == 1) {
            does not cause any test exercising them to fail.

            Consider adding test cases that fail when the code is mutated to
            ensure those bugs would be caught.

            Mutants ran because goranpetrovic is whitelisted

        }
    }
}
```
Mutation testing at Google

```java
int RunMe(int a, int b) {
    if (a == b || b == 1) {
        // Changing this 1 line to
        if (a != b || b == 1) {
            does not cause any test exercising them to fail.
            Consider adding test cases that fail when the code is mutated to
            ensure those bugs would be caught.
            Mutants ran because goranpetrovic is whitelisted
        }
    }
}
```

Please fix

Not useful

Petrovic and Ivankovic, ICSE (SEIP) 2018
Killable vs. productive mutants

Status quo

- **Killable** mutants are **good**  →  **tests**
- **Equivalent** mutants are **bad**  →  **no tests**
Killable vs. productive mutants

Status quo
- **Killable** mutants are **good** \(\rightarrow\) tests
- **Equivalent** mutants are **bad** \(\rightarrow\) no tests

A more nuanced view
- **Killable vs. equivalent** is too simplistic
- **Productive mutants** elicit effective tests, but
  - Killable mutants can be unproductive
  - Equivalent mutants can be productive
Killable vs. productive mutants

Status quo
- Killable mutants are good $\iff$ tests
- Equivalent mutants are bad $\iff$ no tests

A more nuanced view
- Killable vs. equivalent is too simplistic
- Productive mutants elicit effective tests, but
  - Killable mutants can be unproductive
  - Equivalent mutants can be productive

The notion of productive mutants is fuzzy!

A mutant is **productive** if it is
1. killable and **elicits an effective test** or
2. equivalent and **advances code quality or knowledge**
Killable vs. productive mutants

PollEv.com/renejust859

Welcome to renejust859's presentation
As soon as renejust859 displays a poll, we'll update this area to give you the voting options.
Easy as pie. Just hang tight, you're ready to go.
Killable vs. productive mutants (1)

**Original program**

```java
public double getAvg(double[] nums) {
    double sum = 0;
    int len = nums.length;
    for (int i = 0; i < len; ++i) {
        sum = sum + nums[i];
    }
    return sum / len;
}
```

**Mutant**

```java
public double getAvg(double[] nums) {
    double sum = 0;
    int len = nums.length;
    for (int i = 0; i < len; ++i) {
        sum = sum * nums[i];
    }
    return sum / len;
}
```

Is the mutant is **killable**?
The mutant is **killable**, but is it **productive**?
Killable vs. productive mutants (2)

Original program

```java
public double getAvg(double[] nums) {
    int len = nums.length;
    double sum = 0;
    double avg = 0;
    for (int i = 0; i < len; ++i) {
        avg = avg + (nums[i] / len);
        sum = sum + nums[i];
    }
    return sum / len;
}
```

Mutant

```java
public double getAvg(double[] nums) {
    int len = nums.length;
    double sum = 0;
    double avg = 0;
    for (int i = 0; i < len; ++i) {
        avg = avg * (nums[i] / len);
        sum = sum + nums[i];
    }
    return sum / len;
}
```

Is the mutant killable?
The mutant is not killable, but is it unproductive?
Killable vs. productive mutants (3)

Original program

...  
Set cache = new HashSet(a * b);
...

Mutant

...  
Set cache = new HashSet(a + b);
...

Is the mutant killable?
The mutant is **killable**, but **is it productive?**
Many mutants but not all are productive

A mutant is **productive** if it is:

1. **killable** and **elicits an effective test** or
2. **equivalent** and **advances code quality or knowledge**
Existing selection strategies

Program

Mutants

Tests

JUnit

JUnit

JUnit

JUnit

Effective tests
Existing selection strategies

Program

Sample mutation operators

Mutants

Tests

Effective tests

JUnit

JUnit

JUnit
Existing selection strategies

Sample generated mutants

Effective tests

JUnit

JUnit

JUnit

JUnit
Existing selection strategies are **program-independent** and **no better than random**

*Gopinath et al., ICSE’16, *Kurtz et al., FSE’16
Mutation testing 2.0

Program → Mutants → Tests

Effective tests

JUnit