SWE 637 Software Testing

Chapter 9

In-class exercise

Dr. Brittany Johnson-Matthews
(Dr. B for short)

https://go.gmu.edu/SWE637

Adapted from slides by Jeff Offutt and Bob Kurtz
Exercise 8.3 #12

```java
class GoodFastCheap {
    boolean good = false;
    boolean fast = false;
    boolean cheap = false;

    public void makeGood () {
        good = true;
        if (fast && cheap) cheap = false;
    }

    public void makeFast () {
        fast = true;
        if (good && cheap) good = false;
    }

    public void makeCheap () {
        cheap = true;
        if (good && fast) fast = false;
    }

    public void makeBad () { good = false; }
    public void makeSlow () { fast = false; }
    public void makeExpensive () { cheap = false; }

    public boolean isSatisfactory () {
        if ((good && fast) || (good && cheap) || (fast && cheap))
            return true;
        return false;
    }

    public boolean isSatisfactoryRefactored (){
        if (good && fast) return true;
        if (good && cheap) return true;
        if (fast && cheap) return true;
        return false;
    }
}
```

Good, fast, and cheap: pick any two out of three!
Consider a mutation operator that replaces each instance of a boolean variable with the boolean literals true and false

1. How many mutants does this operator generate for method isSatisfactory()?
2. List them (just the mutated line, not the whole method) using the ∆1.. ∆N format
Mutating GoodFastCheap

Consider a mutation operator that replaces each instance of a boolean variable with the boolean literals true and false.

1. How many mutants does this operator generate for method isSatisfactory()?
2. List them (just the mutated line, not the whole method) using the \( \Delta_1.. \Delta_N \) format.
Consider a mutation operator that replaces each instance of a boolean variable with the boolean literals true and false.

How many mutants does this operator generate for method isSatisfactory()?

12 mutants
public final class GoodFastCheap {
    ...

    public boolean isSatisfactory () {
        if ((good && fast) || (good && cheap) || (fast && cheap))
            return true;
        return false;
    }
}

A Mutated GoodFastCheap

12 mutants
Strong Killing Mutants

Which mutants are strongly killed by test TTF (good=T, fast=T, cheap=F)?

<table>
<thead>
<tr>
<th>Case</th>
<th>Condition</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Δ1</td>
<td>if (good &amp;&amp; fast)</td>
<td></td>
</tr>
<tr>
<td>Δ2</td>
<td>if (false &amp;&amp; fast)</td>
<td></td>
</tr>
<tr>
<td>Δ3</td>
<td>if (good &amp;&amp; true)</td>
<td></td>
</tr>
<tr>
<td>Δ4</td>
<td>if (good &amp;&amp; false)</td>
<td></td>
</tr>
<tr>
<td>Δ5</td>
<td>if (good &amp;&amp; fast)</td>
<td></td>
</tr>
<tr>
<td>Δ6</td>
<td>if (good &amp;&amp; fast)</td>
<td></td>
</tr>
<tr>
<td>Δ7</td>
<td>if (good &amp;&amp; fast)</td>
<td></td>
</tr>
<tr>
<td>Δ8</td>
<td>if (good &amp;&amp; fast)</td>
<td></td>
</tr>
<tr>
<td>Δ9</td>
<td>if (good &amp;&amp; fast)</td>
<td></td>
</tr>
<tr>
<td>Δ10</td>
<td>if (good &amp;&amp; fast)</td>
<td></td>
</tr>
<tr>
<td>Δ11</td>
<td>if (good &amp;&amp; fast)</td>
<td></td>
</tr>
<tr>
<td>Δ12</td>
<td>if (good &amp;&amp; fast)</td>
<td></td>
</tr>
</tbody>
</table>
Strong Killing Mutants

Which mutants are strongly killed by test TTF (good=T, fast=T, cheap=F)?

2, 4

Which mutants are strongly killed by test TFT?

```
if ((good && fast) || (good && cheap) || (fast && cheap))
Δ1 if ((true && fast) || (good && cheap) || (fast && cheap))
Δ2 if ((false && fast) || (good && cheap) || (fast && cheap))
Δ3 if ((good && true) || (good && cheap) || (fast && cheap))
Δ4 if ((good && false) || (good && cheap) || (fast && cheap))
Δ5 if ((good && fast) || (true && cheap) || (fast && cheap))
Δ6 if ((good && fast) || (false && cheap) || (fast && cheap))
Δ7 if ((good && fast) || (good && true) || (fast && cheap))
Δ8 if ((good && fast) || (good && false) || (fast && cheap))
Δ9 if ((good && fast) || (good && cheap) || (true && cheap))
Δ10 if ((good && fast) || (good && cheap) || (false && cheap))
Δ11 if ((good && fast) || (good && cheap) || (fast && true))
Δ12 if ((good && fast) || (good && cheap) || (fast && false))
```
Strong Killing Mutants

Which mutants are strongly killed by test TTF (good=T, fast=T, cheap=F)?

2, 4

Which mutants are strongly killed by test TFT?

6, 8

Which mutants are strongly killed by test FFF?

<table>
<thead>
<tr>
<th>Case</th>
<th>Condition</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>good &amp;&amp; fast</td>
<td>(good &amp;&amp; cheap)</td>
</tr>
<tr>
<td>2</td>
<td>!good &amp;&amp; fast</td>
<td>(good &amp;&amp; cheap)</td>
</tr>
<tr>
<td>3</td>
<td>good &amp;&amp; true</td>
<td>(good &amp;&amp; cheap)</td>
</tr>
<tr>
<td>4</td>
<td>!good &amp;&amp; true</td>
<td>(good &amp;&amp; cheap)</td>
</tr>
<tr>
<td>5</td>
<td>good &amp;&amp; false</td>
<td>(true &amp;&amp; cheap)</td>
</tr>
<tr>
<td>6</td>
<td>!good &amp;&amp; false</td>
<td>(false &amp;&amp; cheap)</td>
</tr>
<tr>
<td>7</td>
<td>(good &amp;&amp; fast)</td>
<td>(good &amp;&amp; true)</td>
</tr>
<tr>
<td>8</td>
<td>(good &amp;&amp; fast)</td>
<td>(true &amp;&amp; false)</td>
</tr>
<tr>
<td>9</td>
<td>(good &amp;&amp; fast)</td>
<td>(false &amp;&amp; true)</td>
</tr>
<tr>
<td>10</td>
<td>(good &amp;&amp; fast)</td>
<td>(false &amp;&amp; cheap)</td>
</tr>
<tr>
<td>11</td>
<td>(good &amp;&amp; fast)</td>
<td>[false &amp;&amp; cheap]</td>
</tr>
<tr>
<td>12</td>
<td>(good &amp;&amp; fast)</td>
<td>(true &amp;&amp; cheap)</td>
</tr>
</tbody>
</table>
Strong Killing Mutants

Which mutants are strongly killed by test TTF (good=T, fast=T, cheap=F)?
2, 4

Which mutants are strongly killed by test TFT?
6, 8

Which mutants are strongly killed by test FFF?
None

Determine a test to kill each mutant. Are any of the mutants equivalent?

<table>
<thead>
<tr>
<th>Test</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Δ1</td>
<td>if ((good &amp;&amp; fast)</td>
</tr>
<tr>
<td>Δ2</td>
<td>if ((true &amp;&amp; fast)</td>
</tr>
<tr>
<td>Δ3</td>
<td>if ((false &amp;&amp; fast)</td>
</tr>
<tr>
<td>Δ4</td>
<td>if ((good &amp;&amp; true)</td>
</tr>
<tr>
<td>Δ5</td>
<td>if ((good &amp;&amp; false)</td>
</tr>
<tr>
<td>Δ6</td>
<td>if ((good &amp;&amp; fast)</td>
</tr>
<tr>
<td>Δ7</td>
<td>if ((true &amp;&amp; fast)</td>
</tr>
<tr>
<td>Δ8</td>
<td>if ((false &amp;&amp; fast)</td>
</tr>
<tr>
<td>Δ9</td>
<td>if ((true &amp;&amp; fast)</td>
</tr>
<tr>
<td>Δ10</td>
<td>if ((true &amp;&amp; fast)</td>
</tr>
<tr>
<td>Δ11</td>
<td>if ((true &amp;&amp; fast)</td>
</tr>
<tr>
<td>Δ12</td>
<td>if ((false &amp;&amp; fast)</td>
</tr>
</tbody>
</table>

If all tests are equivalent, then:

if ((good && fast) || (fast && cheap))

0
Strong Killing Mutants

Which mutants are strongly killed by test TTF (good=T, fast=T, cheap=F)?
2, 4

Which mutants are strongly killed by test TFT?
6, 8

Which mutants are strongly killed by test FFF?
None

Determine a test to kill each mutant. Are any of the mutants equivalent?

No, all mutants are strongly killed by exactly one test
m1 FTF, m2 TTF, m3 TFF, m4 TTF, m5 FFT, m6 TFT, m7 TFF, m8 TFT, m9 FFT, m10 FTT, m11 FTF, m12 FTT
Weakly Killing Mutants

Does any test kill m1 weakly but not strongly?

\[
\text{if } ((\text{good} \&\& \text{fast}) \text{ || (good} \&\& \text{cheap)} \text{ || (fast} \&\& \text{cheap)})
\]

\[
\Delta 1 \text{ if } ((\text{true} \&\& \text{fast}) \text{ || (good} \&\& \text{cheap)} \text{ || (fast} \&\& \text{cheap)}
\]

FTT weakly kills m1

\((\text{true} \&\& \text{fast})\text{ is true, so (good} \&\& \text{cheap)} \text{ and (fast} \&\& \text{cheap)}\text{ are never evaluated (which weakly kills m1 based on an error in the PC), but the result is the same as for P so m1 is not strongly killed} \)
END OF EXERCISE
public boolean isSatisfactoryRefactored() {
    if (good && fast) return true;
    if (good && cheap) return true;
    if (fast && cheap) return true;
    return false;
}

Consider a conditional operator replacement (COR) mutation operator that replaces each instance of a conditional operator { &&, || } with the other, and also replaces the entire conditional expression (the operator and both operands) with true and false.

1. How many mutants does this operator generate for method isSatisfactoryRefactored()?
2. List them (just the mutated line, not the whole method) using the $\Delta 1.. \Delta N$ format
Another Mutated GoodFastCheap

9 mutants

public final class GoodFastCheap {
    ...

    public boolean isSatisfactoryRefactored () {
        if (good && fast) return true;
        if (good || fast) return true;
        if (true) return true;
        if (false) return true;
        if (good && cheap) return true;
        if (good || cheap) return true;
        if (true) return true;
        if (false) return true;
        if (fast && cheap) return true;
        if (fast || cheap) return true;
        if (true) return true;
        if (false) return true;
        return false;
    }
}

Another Mutated GoodFastCheap

9 mutants

public final class GoodFastCheap {
    ...

    public boolean isSatisfactoryRefactored () {
        if (good && fast) return true;
        if (good || fast) return true;
        if (true) return true;
        if (false) return true;
        if (good && cheap) return true;
        if (good || cheap) return true;
        if (true) return true;
        if (false) return true;
        if (fast && cheap) return true;
        if (fast || cheap) return true;
        if (true) return true;
        if (false) return true;
        return false;
    }
}
Another Mutated Good Fast Cheap

Determine a test to strongly kill each mutant (gfc)

```java
if (good && fast) return true;

if (good || fast) return true; // Killed by:

if (true) return true; // Killed by:

if (false) return true; // Killed by:

if (good && cheap) return true;

if (good || cheap) return true; // Killed by:

if (true) return true; // Killed by:

if (false) return true; // Killed by:

if (fast && cheap) return true;

if (fast || cheap) return true; // Killed by:

if (true) return true; // Killed by:

if (false) return true; // Killed by:
```
Another Mutated **GoodFastCheap**

Determine a test to strongly kill each mutant (gfc)

```java
if (good && fast) return true;

∆1 if (good || fast) return true;  // Killed by: TFF,FTF

∆2 if (true) return true;  // Killed by: FFF,TFF,FTF,FFT

∆3 if (false) return true;  // Killed by: TFF

if (good && cheap) return true;

∆4 if (good || cheap) return true;  // Killed by:  

∆5 if (true) return true;  // Killed by:  

∆6 if (false) return true;  // Killed by:  

if (fast && cheap) return true;

∆7 if (fast || cheap) return true;  // Killed by:  

∆8 if (true) return true;  // Killed by:  

∆9 if (false) return true;  // Killed by:  
```
**Another Mutated GoodFastCheap**

Determine a test to strongly kill each mutant (gfc)

<table>
<thead>
<tr>
<th>Mutation</th>
<th>Code</th>
<th>Killer Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Δ1</td>
<td>if (good</td>
<td></td>
</tr>
<tr>
<td>Δ2</td>
<td>if (true) return true;</td>
<td>FFF,TFF,FTF,FFT</td>
</tr>
<tr>
<td>Δ3</td>
<td>if (false) return true;</td>
<td>TFF</td>
</tr>
<tr>
<td>Δ4</td>
<td>if (good</td>
<td></td>
</tr>
<tr>
<td>Δ5</td>
<td>if (true) return true;</td>
<td>FFF,TFF,FTF,FFT</td>
</tr>
<tr>
<td>Δ6</td>
<td>if (false) return true;</td>
<td>TFF</td>
</tr>
<tr>
<td>Δ7</td>
<td>if (fast</td>
<td></td>
</tr>
<tr>
<td>Δ8</td>
<td>if (true) return true;</td>
<td></td>
</tr>
<tr>
<td>Δ9</td>
<td>if (false) return true;</td>
<td></td>
</tr>
</tbody>
</table>
Determine a test to strongly kill each mutant (gfc)

if (good && fast) return true;
\[\Delta1\] if (good || fast) return true; // Killed by: TFF,FTF
\[\Delta2\] if (true) return true; // Killed by: TFF,TFF,FTF,FFT
\[\Delta3\] if (false) return true; // Killed by: FFF,FFF,FTF,FFT

if (good && cheap) return true;
\[\Delta4\] if (good || cheap) return true; // Killed by: TFF,FTF
\[\Delta5\] if (true) return true; // Killed by: FFF,TFF,FTF,FFT
\[\Delta6\] if (false) return true; // Killed by: TFT

if (fast && cheap) return true;
\[\Delta7\] if (fast || cheap) return true; // Killed by: FFF,FFF
\[\Delta8\] if (true) return true; // Killed by: FFF,TFF,FTF,FFT
\[\Delta9\] if (false) return true; // Killed by: FTF