Chapter 10

Retrospective on Unit Testing
The Test Method Pendulum
Test Case Identification Effort

- Program Graph Testing
- Basis Path Testing
- du-Path Testing
- Slice Testing
- Decision Table Testing
- Equivalence Class Testing
- Boundary Value Testing
Number of Test Cases
Traversing the Code-Based Side

- FORTRAN version of the Triangle Program
- Problems with path testing
- Basis path (McCabe) testing?
- Dataflow testing
- Slice testing
  - 80 topologically possible paths
  - Only 11 feasible paths
  - Cyclomatic complexity is 14
  - Basis path testing? (expect 14 basis paths, but only 11 paths are feasible)
  - Dataflow testing?
  - Slice testing?
FORTRAN version of the Triangle Program

Input a, b, c

Match = 0

80 topologically possible paths

1. a = b?
   Y → 2, Match = Match + 1
   N → 3, a = c?

2. Match = Match + 1

3. a = c?
   Y → 4, Match = Match + 2
   N → 5, b = c?

4. Match = Match + 2

5. b = c?
   Y → 6, Match = Match + 3
   N → 7, Match = 0?

6. Match = Match + 3

7. Match = 0?
   N → 8, a + b ≤ c?
   Y → 13, Match = 1?

8. a + b ≤ c?
   N → 9, a + c ≤ b?
   Y → 20, Equilateral
   N → 14, a + b ≤ c?
   Y → 18, Match = 3?
   N → 19, b + c ≤ a?

9. a + c ≤ b?
   N → 10, b + c ≤ a?
   Y → 11, Scalene
   N → 12, Not a Triangle

10. b + c ≤ a?

11. Scalene

12. Not a Triangle

13. Match = 1?
   N → 16, Match = 2?
   Y → 17, a + c ≤ b?

14. a + b ≤ c?
   Y → 14, a + b ≤ c?
   N → 18, Match = 3?
   Y → 19, b + c ≤ a?

15. Isosceles

16. Match = 2?
   N → 18, Match = 3?
   Y → 17, a + c ≤ b?

17. a + c ≤ b?
   Y → 17, a + c ≤ b?
   N → 16, Match = 2?

18. Match = 3?
   Y → 19, b + c ≤ a?
   N → 18, Match = 3?

19. b + c ≤ a?

20. Equilateral

Only 11 feasible paths
Program Graph

(V(G) = 14)

\[ V(G) = e - n + p \]
\[ = 33 - 20 + 1 \]
\[ = 14 \]
### Feasible Paths in the FORTRAN Version

<table>
<thead>
<tr>
<th>Path</th>
<th>Node Sequence</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>1–2–3–4–5–6–7–13–16–18–20</td>
<td>Equilateral</td>
</tr>
<tr>
<td>p2</td>
<td>1–3–5–6–7–13–16–18–19–15</td>
<td>Isosceles (b = c)</td>
</tr>
<tr>
<td>p3</td>
<td>1–3–5–6–7–13–16–18–19–12</td>
<td>Not a Triangle (b = c)</td>
</tr>
<tr>
<td>p5</td>
<td>1–3–4–5–7–13–16–17–12</td>
<td>Not a Triangle (a = c)</td>
</tr>
<tr>
<td>p6</td>
<td>1–2–3–5–7–13–14–15</td>
<td>Isosceles (a = b)</td>
</tr>
<tr>
<td>p7</td>
<td>1–2–3–5–7–13–14–12</td>
<td>Not a Triangle (a = b)</td>
</tr>
<tr>
<td>p8</td>
<td>1–3–5–7–8–12</td>
<td>Not a Triangle (a + b ≤ c)</td>
</tr>
<tr>
<td>p9</td>
<td>1–3–5–7–8–9–12</td>
<td>Not a Triangle (b + c ≤ a)</td>
</tr>
<tr>
<td>p10</td>
<td>1–3–5–7–8–9–10–12</td>
<td>Not a Triangle (a + c ≤ b)</td>
</tr>
<tr>
<td>p11</td>
<td>1–3–5–7–8–9–10–11</td>
<td>Scalene</td>
</tr>
</tbody>
</table>
Basis Path Testing?

- Problematic!
- $V(G) = 14$ suggests 14 basis paths
- Only 11 logically feasible paths
- “The Method” generates infeasible paths
Dataflow Testing?

- Moving closer to “semantic reality”
- Implications of the match variable
  - FORTRAN style to reduce decisions
  - Only values of match are 0, 1, 2, 3, and 6
- The match variable has
  - four definition nodes
  - three computation uses
  - four predicate uses
  - 28 define/use paths
- The “side” variables a, b, and c have
  - 1 definition node
  - 9 use nodes
  - 9 definition-clear paths to test
- 37 dataflow test cases
Slice Testing?

• Even closer to “semantic reality”
• Make backward slices on
  – scalene at node 11
  – notATriangle at node 12
  – isosceles at node 15
  – equilateral at node 20
Slice Test Cases

- 3 test cases for $S($scalene,11$)$
  - $(3, 4, 5)$
  - $(4, 5, 3)$
  - $(5, 3, 4)$
- 1 test case for $S($equilateral,20$)$
- 3 test cases for $S($isosceles,15$)$
- 18 test cases for $S($notATriangle, 12$)$
  - six ways to fail the triangle test
  - each way has three permutations, as for $S($scalene, 11$)$
- 25 slice-based test cases
- Compare with equivalence class test cases
Traversing the Specification-Based Side

- **Boundary Value Testing**
  - nominal values: 15 test cases
  - worst case values: 125 test cases

- **Equivalence Class Testing**
  - 11 equivalence classes
  - correspond exactly to the 11 feasible paths

- **Decision Table Testing**
  - 20 possible rules
  - 2 impossible rules
  - provides insight into the match variable
## Path Coverage of Nominal Boundary Values

<table>
<thead>
<tr>
<th>Case</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>Expected</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
<td>100</td>
<td>1</td>
<td>Isosceles</td>
<td>p6</td>
</tr>
<tr>
<td>2</td>
<td>100</td>
<td>100</td>
<td>2</td>
<td>Isosceles</td>
<td>p6</td>
</tr>
<tr>
<td>3</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>Equilateral</td>
<td>p1</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
<td>100</td>
<td>199</td>
<td>Isosceles</td>
<td>p6</td>
</tr>
<tr>
<td>5</td>
<td>100</td>
<td>100</td>
<td>200</td>
<td>Not a Triangle</td>
<td>p7</td>
</tr>
<tr>
<td>6</td>
<td>100</td>
<td>1</td>
<td>100</td>
<td>Isosceles</td>
<td>p4</td>
</tr>
<tr>
<td>7</td>
<td>100</td>
<td>2</td>
<td>100</td>
<td>Isosceles</td>
<td>p4</td>
</tr>
<tr>
<td>8</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>Equilateral</td>
<td>p1</td>
</tr>
<tr>
<td>9</td>
<td>100</td>
<td>199</td>
<td>100</td>
<td>Isosceles</td>
<td>p4</td>
</tr>
<tr>
<td>10</td>
<td>100</td>
<td>200</td>
<td>100</td>
<td>Not a Triangle</td>
<td>p5</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>100</td>
<td>100</td>
<td>Isosceles</td>
<td>p2</td>
</tr>
<tr>
<td>12</td>
<td>2</td>
<td>100</td>
<td>100</td>
<td>Isosceles</td>
<td>p2</td>
</tr>
<tr>
<td>13</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>Equilateral</td>
<td>p1</td>
</tr>
<tr>
<td>14</td>
<td>199</td>
<td>100</td>
<td>100</td>
<td>Isosceles</td>
<td>p2</td>
</tr>
<tr>
<td>15</td>
<td>200</td>
<td>100</td>
<td>100</td>
<td>Not a Triangle</td>
<td>p3</td>
</tr>
</tbody>
</table>
Path Coverage of Nominal and Worst Case Boundary Value Test Cases

<table>
<thead>
<tr>
<th></th>
<th>p1</th>
<th>p2</th>
<th>p3</th>
<th>p4</th>
<th>p5</th>
<th>p6</th>
<th>p7</th>
<th>p8</th>
<th>p9</th>
<th>p10</th>
<th>p11</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nominal</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Worst-case</td>
<td>5</td>
<td>12</td>
<td>6</td>
<td>11</td>
<td>6</td>
<td>12</td>
<td>7</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>12</td>
</tr>
</tbody>
</table>
Equivalence Class Testing

Classes from Chapter 6:

D1  = \{<a, b, c> : a = b = c\}  
D2  = \{<a, b, c> : a = b, a \neq c\}  
D3  = \{<a, b, c> : a = c, a \neq b\}  
D4  = \{<a, b, c> : b = c, a \neq b\}  
D5  = \{<a, b, c> : a \neq b, a \neq c, b \neq c\}  
D6  = \{<a, b, c> : a > b + c\}  
D7  = \{<a, b, c> : b > a + c\}  
D8  = \{<a, b, c> : c > a + b\}  
D9  = \{<a, b, c> : a = b + c\}  
D10 = \{<a, b, c> : b = a + c\}  
D11 = \{<a, b, c> : c = a + b\}

These classes will exactly execute the 11 feasible paths.
## Decision Table Testing (continued)

<table>
<thead>
<tr>
<th></th>
<th>c1. Match =</th>
<th>0</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>c2. a + b &lt;= c?</td>
<td>T</td>
<td>F!</td>
<td>F!</td>
<td>F</td>
</tr>
<tr>
<td></td>
<td></td>
<td>T</td>
<td>F!</td>
<td>F!</td>
</tr>
<tr>
<td>c3. a + c &lt;= b?</td>
<td>F!</td>
<td>T</td>
<td>F!</td>
<td>F</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F!</td>
<td>T</td>
<td>F!</td>
</tr>
<tr>
<td>c4. b + c &lt;= a?</td>
<td>F!</td>
<td>F!</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F!</td>
<td>F!</td>
<td>T</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>a1. Scalene</th>
<th>x</th>
</tr>
</thead>
<tbody>
<tr>
<td>a2. Not a Triangle</td>
<td>x</td>
</tr>
<tr>
<td>a3. Isosceles</td>
<td>x</td>
</tr>
<tr>
<td>a4. Equilateral</td>
<td>x</td>
</tr>
<tr>
<td>a5. Impossible</td>
<td></td>
</tr>
</tbody>
</table>
(Extended Entry) Decision Table Testing

<table>
<thead>
<tr>
<th>c1. Match =</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>c2. a + b &lt;= c?</td>
<td>T</td>
<td>F!</td>
<td>F!</td>
<td>F</td>
</tr>
<tr>
<td>c3. a + c &lt;= b?</td>
<td>F!</td>
<td>T</td>
<td>F!</td>
<td>F</td>
</tr>
<tr>
<td>c4. b + c &lt;= a?</td>
<td>F!</td>
<td>F!</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>a1. Scalene</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>a2. Not a Triangle</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>a3. Isosceles</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a4. Equilateral</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>a5. Impossible</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Notes on the EEDT

- the “match” variable notes which two sides (if any) are equal
- we are stuck with some reverse logic on the triangle inequalities
- there is a typo in Table 10.4. In the rule where
  - match = 1
  - c2 and c3 are F!
  - c4 = F is an isosceles triangle
- In the rule before that, the two x entries say that something is both Not a Triangle and Isosceles. (Oppss!)
Redundancy Metrics Definitions

- Given a test method $M$ that generates $m$ test cases and a coverage metric $S$ that defines $s$ test cases for $n$ structural elements...
  - The coverage of a methodology $M$ with respect to a metric $S$, denoted $C(M,S)$, is ratio of $n$ to $s$.
  - The redundancy of a methodology $M$ with respect to a metric $S$, denoted $R(M,S)$, is ratio of $m$ to $s$.
  - The net redundancy of a methodology $M$ with respect to a metric $S$, denoted $NR(M,S)$ is ratio of $m$ to $n$. 
Redundancy Metrics for the Triangle Program

<table>
<thead>
<tr>
<th>Method</th>
<th>m</th>
<th>n</th>
<th>s</th>
<th>$C(M, S) = \frac{n}{s}$</th>
<th>$R(M, S) = \frac{m}{s}$</th>
<th>$NR(M, S) = \frac{m}{n}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nominal</td>
<td>15</td>
<td>7</td>
<td>11</td>
<td>0.64</td>
<td>1.36</td>
<td>2.14</td>
</tr>
<tr>
<td>Worst-case</td>
<td>125</td>
<td>11</td>
<td>11</td>
<td>1</td>
<td>11.36</td>
<td>11.36</td>
</tr>
<tr>
<td>Goal</td>
<td>s</td>
<td>s</td>
<td>s</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Case Study

A hypothetical Insurance Premium Program computes the semi-annual car insurance premium based on two parameters: the policy holder's age and driving record:

$$\text{Premium} = \text{BaseRate} \times \text{ageMultiplier} - \text{safeDrivingReduction}$$

The ageMultiplier is a function of the policy holder's age, and the safe driving reduction is given when the current points (assigned by traffic courts for moving violations) on the policy holder's driver's license are below an age-related cutoff. Policies are written for drivers in the age range of 16 to 100. Once a policy holder has 12 points, his/her driver's license is suspended (hence there is no need for insurance). The BaseRate changes from time to time; for this example, it is $500 for a semi-annual premium.
Insurance Premium Program Data

<table>
<thead>
<tr>
<th>Age Range</th>
<th>Age Multiplier</th>
<th>Points Cutoff</th>
<th>Safe Driving Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>16 &lt;= age &lt; 25</td>
<td>2.8</td>
<td>1</td>
<td>50</td>
</tr>
<tr>
<td>25 &lt;= age &lt; 35</td>
<td>1.8</td>
<td>3</td>
<td>50</td>
</tr>
<tr>
<td>35 &lt;= age &lt; 45</td>
<td>1.0</td>
<td>5</td>
<td>100</td>
</tr>
<tr>
<td>45 &lt;= age &lt; 60</td>
<td>0.8</td>
<td>7</td>
<td>150</td>
</tr>
<tr>
<td>60 &lt;= age &lt;= 100</td>
<td>1.5</td>
<td>5</td>
<td>200</td>
</tr>
</tbody>
</table>
Insurance Premium Program Calculations, Test Method Selection

- Premium = BaseRate*ageMultiplier – safeDrivingReduction
- ageMultiplier = F1(age)  [from table]
- safeDrivingReduction = F2(age, points) [from table]
- age and safeDrivingReduction are physical variables, with a dependency in F2.
- Boundary values for age: 16, 17, 54, 99, 100
- Boundary values for safeDrivingReduction: 0, 1, 6, 11, 12
- Robust values for age and safeDrivingReduction are not allowed by business rules.
- Worst case BVA yields 25 test cases, and many gaps, some redundancy. Need something better.
Worst Case Boundary Value Test Cases
(severe gaps!)
Closer Look At One Age Interval
(severe redundancy)
Weak and Robust Normal Equivalence Classes? (not much help)
Hybrid Test Cases for One Age Class
Code-Based Testing for the Insurance Premium Program

• (Source pseudo-code and program graph on next slide)
• Cyclomatic complexity: \( V(G) = 11 \)
• There are 11 feasible paths.
Program InsurancePremium
Dim driverAge, points As Integer
Dim baseRate, premium As Real
1 Input(baseRate, driverAge, points)
2 premium = 0
3 safeDrivingReduction = 0
4 Select Case driverAge
5 Case 1: 16<= driverAge < 25
6    ageMultiplier = 2.8
7    If points < 1 Then
8        safeDrivingReduction = 50
9    EndIf
10 Case 2: 25<= driverAge < 35
11    ageMultiplier = 1.8
12    If points < 3 Then
13        safeDrivingReduction = 50
14    EndIf
15 Case 3: 35<= driverAge < 45
16    ageMultiplier = 1.0
17    If points < 5 Then
18        safeDrivingReduction = 100
19    EndIf
20 Case 4: 45<= driverAge < 60
21    ageMultiplier = 0.8
22    If points < 7 Then
23        safeDrivingReduction = 150
24    EndIf
25 Case 5: 60<= driverAge <= 100
26    ageMultiplier = 1.5
27    If points < 5 Then
28        safeDrivingReduction = 200
29    EndIf
30 Case 6: Else
31    Output("Driver age out of range")
32 End Select
33 premium = baseRate * ageMultiplier - safeDrivingReduction
34 Output(premium)
35 End
### Paths in the Insurance Premium Program

<table>
<thead>
<tr>
<th>Path</th>
<th>Node Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>1-2-3-4-5-6-7-9-32-33-34-35</td>
</tr>
<tr>
<td>p2</td>
<td>1-2-3-4-5-6-7-8-9-32-33-34-35</td>
</tr>
<tr>
<td>p3</td>
<td>1-2-3-4-10-11-12-14-32-33-34-35</td>
</tr>
<tr>
<td>p4</td>
<td>1-2-3-4-10-11-12-13-14-32-33-34-35</td>
</tr>
<tr>
<td>p5</td>
<td>1-2-3-4-15-16-17-19-32-33-34-35</td>
</tr>
<tr>
<td>p7</td>
<td>1-2-3-4-20-21-22-24-32-33-34-35</td>
</tr>
<tr>
<td>p8</td>
<td>1-2-3-4-20-21-22-23-24-32-33-34-35</td>
</tr>
<tr>
<td>p9</td>
<td>1-2-3-4-25-26-27-29-32-33-34-35</td>
</tr>
<tr>
<td>p10</td>
<td>1-2-3-4-25-26-27-28-29-32-33-34-35</td>
</tr>
<tr>
<td>p11</td>
<td>1-2-3-4-30-31-32-33-34-35</td>
</tr>
</tbody>
</table>
### Path Coverage of Spec-Based Methods

<table>
<thead>
<tr>
<th>Test Cases</th>
<th>Paths covered</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boundary value</td>
<td>p1, p2, p7, p8, p9, p10</td>
</tr>
<tr>
<td>Worst-case boundary value</td>
<td>p1, p2, p3, p4, p5, p6, p7, p8, p9, p10</td>
</tr>
<tr>
<td>Weak normal equivalence class</td>
<td>P1, p2, p3, p4, p5, p6, p7, p8, p9</td>
</tr>
<tr>
<td>Robust normal equivalence class</td>
<td>p1, p2, p3, p4, p5, p6, p7, p8, p9, p10, p11</td>
</tr>
<tr>
<td>Decision table</td>
<td>p1, p2, p3, p4, p5, p6, p7, p8, p9, p10, p11</td>
</tr>
<tr>
<td>Hybrid spec-based (extended to all age classes)</td>
<td>p1, p2, p3, p4, p5, p6, p7, p8, p9, p10, p11</td>
</tr>
</tbody>
</table>