Overview

• The most usual application of graph criteria is to program source
• Graph: Usually the control flow graph (CFG)
• Node coverage: execute every statement
• Edge coverage: execute every branch
• Loops: looping structures such as for loops, while loops, etc.

Data flow coverage: augment the CFG
  – defs are statements that assign values to variables
  – uses are statements that use variables

Data flow not covered in undergraduate course
Control Flow Graphs

• A CFG models all executions of a method by describing control structures
• Nodes: statements or sequences of statements (basic blocks)
• Edges: transfers of control
• Basic Block: A sequence of statements such that if the first statement is executed, all statements will be (no branches)

• CFGs are sometimes annotated with extra information
  – branch predicates
  – defs
  – uses
• Rules for translating statements into graphs …

CFG: The if Statement

```java
if (x < y)
    { y = 0; x = x + 1; }
else
    { x = y; }
```

Diagram:

1. (x < y)
2. y = 0
3. x = x + 1
4. x = y

Diagram:

1. if (x < y)
2. y = 0
3. x = x + 1
4. x = y

CFG : The if-Return Statement

```java
if (x < y) {
    return;
}
print (x);
return;
```

NO edge from node 2 to 3. The return nodes must be distinct.

Loops

- Loops require “extra” nodes to be added
- Nodes that do not represent statements or basic blocks
CFG : while and for Loops

```plaintext
x = 0;
while (x < y)
{
    y = f(x, y);
    x = x + 1;
}
```

```plaintext
for (x = 0; x < y; x++)
{
    y = f(x, y);
}
```

CFG : The case (switch) Structure

```plaintext
read (c);
switch (c)
{
    case 'N':
        y = 25;
        break;
    case 'Y':
        y = 50;
        break;
    default:
        y = 0;
        break;
}
print(y);
```
Example Control Flow – Stats

```java
public static void computeStats (int[] numbers) {
    int length = numbers.length;
    double med, var, sd, mean, sum, varsum;
    sum = 0;
    for (int i = 0; i < length; i++) {
        sum += numbers[i];
    }
    med = numbers[length / 2];
    mean = sum / (double) length;
    varsum = 0;
    for (int i = 0; i < length; i++) {
        varsum = varsum + ((numbers[i] - mean) * (numbers[i] - mean));
    }
    var = varsum / (length - 1.0);
    sd = Math.sqrt(var);
    System.out.println("length: " + length);
    System.out.println("mean: " + mean);
    System.out.println("median: " + med);
    System.out.println("variance: " + var);
    System.out.println("standard deviation: " + sd);
}
```

Control Flow Graph for Stats

1. int length = numbers.length;
2. double med, var, sd, mean, sum, varsum;
3. sum = 0;
4. for (int i = 0; i < length; i++)
   sum += numbers[i];
5. med = numbers[length / 2];
6. mean = sum / (double) length;
7. varsum = 0;
8. for (int i = 0; i < length; i++)
   varsum = varsum + ((numbers[i] - mean) * (numbers[i] - mean));
9. var = varsum / (length - 1.0);
10. sd = Math.sqrt(var);
11. System.out.println("length: " + length);
12. System.out.println("mean: " + mean);
13. System.out.println("median: " + med);
14. System.out.println("variance: " + var);
15. System.out.println("standard deviation: " + sd);
**Control Flow TRs and Test Paths – EC**

![Diagram of Control Flow TRs and Test Paths – EC]

**Edge Coverage**

<table>
<thead>
<tr>
<th>TR</th>
<th>Test Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.</td>
<td>[1, 2, 3, 4, 3, 5, 6, 7, 6, 8]</td>
</tr>
<tr>
<td>B.</td>
<td>[2, 3, 4, 3, 5, 6, 7, 6, 8]</td>
</tr>
<tr>
<td>C.</td>
<td>[3, 4, 3, 5, 6, 7, 6, 8]</td>
</tr>
<tr>
<td>D.</td>
<td>[3, 4, 3, 5, 6, 7, 6, 8]</td>
</tr>
<tr>
<td>E.</td>
<td>[4, 3, 5, 6, 7, 6, 8]</td>
</tr>
<tr>
<td>F.</td>
<td>[5, 6, 7, 6, 8]</td>
</tr>
<tr>
<td>G.</td>
<td>[6, 7, 6, 8]</td>
</tr>
<tr>
<td>H.</td>
<td>[6, 8]</td>
</tr>
<tr>
<td>I.</td>
<td>[7, 6]</td>
</tr>
</tbody>
</table>

**Control Flow TRs and Test Paths – EPC**

![Diagram of Control Flow TRs and Test Paths – EPC]

**Edge-Pair Coverage**

<table>
<thead>
<tr>
<th>TR</th>
<th>Test Paths</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. [1, 2, 3]</td>
<td>i. [1, 2, 3, 4, 3, 5, 6, 7, 6, 8]</td>
</tr>
<tr>
<td>B. [2, 3, 4]</td>
<td>ii. [1, 2, 3, 5, 6, 8]</td>
</tr>
<tr>
<td>C. [2, 3, 5]</td>
<td>iii. [1, 2, 3, 4, 3, 4, 3, 5, 6, 7, 6, 7, 6, 8]</td>
</tr>
<tr>
<td>D. [3, 4, 3]</td>
<td></td>
</tr>
<tr>
<td>E. [3, 5, 6]</td>
<td></td>
</tr>
<tr>
<td>F. [4, 3, 5]</td>
<td></td>
</tr>
<tr>
<td>G. [5, 6, 7]</td>
<td></td>
</tr>
<tr>
<td>H. [5, 6, 8]</td>
<td></td>
</tr>
<tr>
<td>I. [6, 7, 6]</td>
<td></td>
</tr>
<tr>
<td>J. [7, 6, 8]</td>
<td></td>
</tr>
<tr>
<td>K. [4, 3, 4]</td>
<td></td>
</tr>
<tr>
<td>L. [7, 6, 7]</td>
<td></td>
</tr>
</tbody>
</table>

**TP** | **TRs toured** | **sidetrips**
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>A, B, D, E, F, G, I</td>
<td></td>
</tr>
<tr>
<td>ii</td>
<td>A, C, E, H</td>
<td></td>
</tr>
<tr>
<td>iii</td>
<td>A, B, C, D, E, F, G, I, J, K, L</td>
<td>H</td>
</tr>
</tbody>
</table>
Control Flow TRs and Test Paths – PPC

Prime Path Coverage

<table>
<thead>
<tr>
<th>TR</th>
<th>Test Paths</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. [3, 4, 3]</td>
<td>i.  [1, 2, 3, 4, 3, 5, 6, 7, 6, 8]</td>
</tr>
<tr>
<td>B. [4, 3, 4]</td>
<td>ii. [1, 2, 3, 4, 3, 4, 3, 5, 6, 7, 6, 8]</td>
</tr>
<tr>
<td>C. [7, 6, 7]</td>
<td>iii. [1, 2, 3, 4, 3, 5, 6, 8]</td>
</tr>
<tr>
<td>D. [7, 6, 8]</td>
<td>iv. [1, 2, 3, 5, 6, 7, 6, 8]</td>
</tr>
<tr>
<td>E. [6, 7, 6]</td>
<td>v.  [1, 2, 3, 5, 6, 8]</td>
</tr>
<tr>
<td>F. [1, 2, 3, 4]</td>
<td></td>
</tr>
<tr>
<td>G. [4, 3, 5, 6, 7]</td>
<td></td>
</tr>
<tr>
<td>H. [4, 3, 5, 6, 8]</td>
<td></td>
</tr>
<tr>
<td>I. [1, 2, 3, 5, 6, 7]</td>
<td></td>
</tr>
<tr>
<td>J. [1, 2, 3, 5, 6, 8]</td>
<td></td>
</tr>
</tbody>
</table>

TP | TRs toured          | sidetrips |
---|---------------------|-----------|
 i | A, D, E, F, G      | H, I, J   |
 ii| A, B, C, D, E, F, G| H, I, J   |
 iii| A, F, H            | J         |
 iv| D, E, F, I         | J         |
v | J                   |           |

Summary

- Applying the graph test criteria to control flow graphs is relatively straightforward
  - Most of the developmental research work was done with CFGs
- A few subtle decisions must be made to translate control structures into the graph
- Some tools will assign each statement to a unique node
  - These slides and the book uses basic blocks
  - Coverage is the same, although the bookkeeping will differ