Workshop 3 (Week 4) – Structural Coverage

Sample Solution

The purpose of this workshop is to practice and develop an understanding of various control flow based structural coverage metrics.

1. Concepts

- **What is condition coverage? Give an example.**

  Executing true and false of each condition. Example: in the expression ((x>5) && (y>0)), there are two conditions: x> 5 and y>0.
  To achieve 100% condition coverage, the true and false branches of each condition should be executed at least once.

- **What is condition/decision coverage? Give an example.**

  Condition/Decision Coverage (C/DC): Combing DC and CC.
  It overcomes the limitation of Decision Coverage (DC) and Condition Coverage (CC).

```
1 begin
2    int x,y,z;
3    input(x,y);
4    if(x<0 and y<0)
5      z=foo1(x,y)
6    else
7      z=foo2(x,y);
8    output(z);
9 end
```

$T=\{t_1:<x=-3,y=-2>,t_2:<x=4,y=2>\}$

- **Decision coverage = 100%**
- **Condition coverage = 100%**

- **What is multiple condition coverage? Give an example.**

  Multiple condition coverage (MCC) reports whether every possible combination of Boolean sub-expressions (i.e., condition) occurs.
  D=(A<B) or (A>C)

	A<B	A>C	D
1	true	true	true
2	true	false	true
3	false	true	true
4	false	false	false

- **What is modified condition/decision coverage? Give an example.**

  Motivation: Effectively test important combinations of conditions, without exponential blowup in test suite size.

  “Important” combinations means: Each basic condition independently affects the outcome of each decision.

- **How do you compare these coverage metrics?**

  MCC is the strongest metric (high test effectiveness), but requires the largest number of test cases (high test effort).

  DC (Decision Coverage) and CC (Condition Coverage) require a smaller number of test cases, but has lower test effectiveness.

  C/DC achieves better test effectiveness than CC and DC, but may require more test cases than CC and DC.

  MCDC can achieve a balance between MCC and C/DC.

.

2. Coverage Analysis

Assume we want to test the following code, where A, B and C represent three atomic boolean expressions:

```
if ( (A || B) && C ) {
   /* Some code  */
}
else {
  /* Other code */
}
```

- **Design test cases that can achieve 100% statement coverage**

  A = true / B = true / C = true ---> decision is evaluated to "true"
  A = false / B = false / C = false ---> decision is evaluated to "false"

- **Design test cases that can achieve 100% branch decision coverage**

  A = true / B = true / C = true ---> decision is evaluated to "true"
  A = false / B = false / C = false ---> decision is evaluated to "false"

- **Design test cases that can achieve 100% condition coverage**

  A = true / B = true / C = true ---> decision is evaluated to "true"
  A = false / B = false / C = false ---> decision is evaluated to "false"

- **Design test cases that can achieve 100% condition/decision coverage**

A = true / B = true / C = true ---> decision is evaluated to "true"
A = false / B = false / C = false ---> decision is evaluated to "false"

- **Design test cases that can achieve 100% multiple condition coverage**

  A = true / B = true / C = true
  A = true / B = true / C = false
  A = true / B = false / C = true
  A = true / B = false / C = false
  A = false / B = true / C = true
  A = false / B = false / C = false
  A = false / B = false / C = true
  A = false / B = true / C = false

- **Design test cases that can achieve 100% modified condition/decision coverage**

  A = false / B = false / C = true ---> decision is evaluated to "false"
  A = false / B = true / C = true ---> decision is evaluated to "true"
  A = false / B = true / C = false ---> decision is evaluated to "false"
  A = true / B = false / C = true ---> decision is evaluated to "true"

3. The Compute Median Example

Consider the following function that computes the Median value:

Task 1: Design some test cases for the Median function.

Task 2: Compute test coverage (including condition, condition/decision, and multiple condition coverage).

Task 3: Design more test cases to achieve 100% condition, condition/decision, and multiple condition coverage.

Task 4: Implement your test cases as jUnit test cases and execute the test cases.

(If jUnit is not installed at your PC, install it from: https://junit.org/)

```
public static int median(int x, int y, int z){
      int median = 0;
      if(x >= y && x <= z){ // y<=x<=z
         median = x;
      } else if(x >= z && x <= y){ // z<=x<=y
         median = x;
      } else if(y >=x && y < z){ // x<=y<=z
         median = y;
      } else if(y >= z && y <= x){ // z<=y<=x
         median = y;
      } else {    // x<=z<=y or y<=z<=x
         median = z;
      }
      return median;
   }
}
```

1) 100% Condition Coverage:

x>=y	x<=z	y<z	x	y	z
T	T	T	2	1	3
F	F	F	2	3	1

2) 100% Decision Coverage

```
if(x >= y && x <= z){ // y<=x<=z
        median = x;
    }
```

True: x=2 y=1 z=3

```
else if(x >= z && x <= y){ // z<=x<=y
        median = x;
    }
```

True: x=2 y=3 z=1

```
else if(y >=x && y < z){ // x<=y<=z
        median = y;
    }
```

True: x=1 y=2 z=3

```
else if(y >= z && y <= x){ // z<=y<=x
        median = y;
    }
```

True: x=3 y=2 z=1

All FALSE Branches: x=1 y=3 z=2

3) 100% Condition/Decision Coverage (CC + DC):

x=2 y=1 z=3

x=2 y=3 z=1

x=1 y=2 z=3

x=3 y=2 z=1

x=1 y=3 z=2

4) 100% Multiple Condition Coverage:

x>=y	x<=z	y<z	x	y	z
T	T	T	2	1	3
F	F	F	2	3	1
F	T	T	1	2	3
T	F	F	3	2	1
F	T	F	1	3	2
T	F	T	3	1	2

Note that for each test case, the expected output should also be given.

4. Try the Web: Code in Game

https://www.codingame.com/ide/puzzle/the-descent