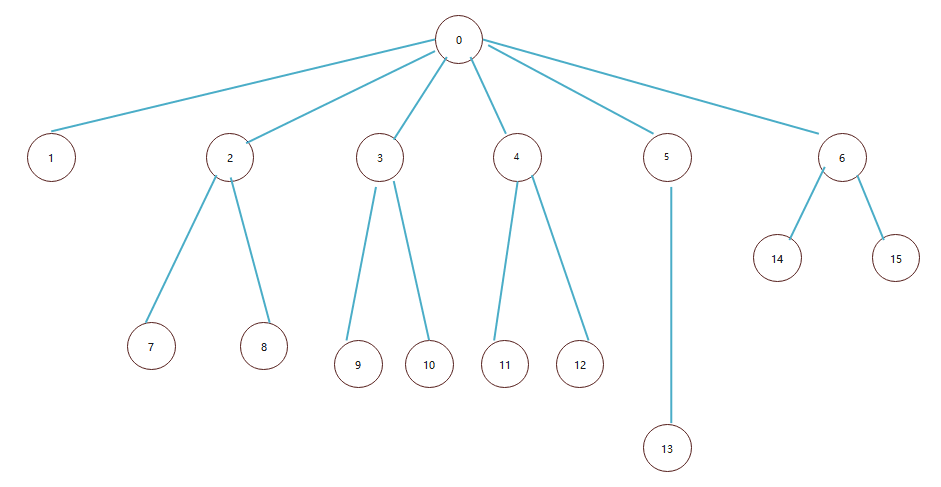
|  |  |  |  |
| --- | --- | --- | --- |
|  | A | B | C |
| 1 | 2 | 2 | 2 |
| 2 | 1 | 1 | 1 |
| 3 | 0 | 0 | 0 |
| 4 | -1 | -1 | -1 |

Each Choice Coverage

|  |  |  |  |
| --- | --- | --- | --- |
|  | A | B | C |
| 1 | 2 | 2 | 2 |
| 2 | 2 | 1 | 1 |
| 3 | 2 | 0 | 0 |
| 4 | 2 | -1 | -1 |
| 5 | 1 | 2 | 1 |
| 6 | 1 | 1 | 0 |
| 7 | 1 | 0 | -1 |
| 8 | 1 | -1 | 2 |
| 9 | 0 | 2 | 0 |
| 10 | 0 | 1 | -1 |
| 11 | 0 | 0 | 2 |
| 12 | 0 | -1 | 1 |

|  |  |  |  |
| --- | --- | --- | --- |
| 13 | -1 | 2 | -2 |
| 14 | -1 | 1 | 2 |
| 15 | -1 | 0 | 1 |
| 16 | -1 | -1 | 0 |

Pair Wise Coverage



0 triOut = 0

1 if (Side1 <= 0 || Side2 <= 0 || Side3 <= 0)  
{  
 triOut = 4;  
 return (triOut);  
}

2if (Side1 == Side2)  
 triOut = triOut + 1;

3if (Side1 == Side3)  
 triOut = triOut + 2;

4 2和3都执行了

5 2 和 3和4都执行了

6 2 ，3，4都没执行

7else if (triOut == 1 && Side1+Side2 > Side3)  
 triOut = 2;

8else triOut = 4;

9else if (triOut == 2 && Side1+Side3 > Side2)  
 triOut = 2;

10else triOut = 4;

11else if (triOut == 3 && Side2+Side3 > Side1)  
 triOut = 2;

12else triOut = 4;

13if (triOut > 3)  
 triOut = 3;

14 6的情况下

if (Side1+Side2 <= Side3 || Side2+Side3 <= Side1 ||  
 Side1+Side3 <= Side2)  
 triOut = 4;

15 6的情况下

Else triOut = 1;

Node Coverage

TR={1,2,3,4,5,6,7,8,9,10,11,12,13,14,15}

TestPaths:[0,1][0,2,7][0,2,8][0,3,9][0,3,10][0,4,11][0,4,12][0,5,13][0,6,14][0,6,15]

|  |  |  |  |
| --- | --- | --- | --- |
|  | A | B | C |
| 1 [0,1] | -1 | 1 | 1 |
| 2 [0,2,7] | 2 | 2 | 3 |
| 3 [0,2,8] | 2 | 2 | 4 |
| 4 [0,3,9] | 2 | 3 | 2 |
| 5 [0,3,10] | 2 | 4 | 2 |
| 6 [0,4,11] | 3 | 2 | 2 |
| 7 [0,4,12] | 4 | 2 | 2 |
| 8 [0,5,13] | 2 | 2 | 2 |
| 9 [0,6,14] | 2 | 3 | 6 |
| 10 [0,6,15] | 2 | 3 | 4 |

Edge Coverage

TR={[0,1],[0,2],[0,3][0,4][0,5][0,6][2,7][2,8][3,9][3,10][4,11][4,12][5,13][6,14][6,15] }

TestPaths:[0,1][0,2,7][0,2,8][0,3,9][0,3,10][0,4,11][0,4,12][0,5,13][0,6,14][0,6,15]

|  |  |  |  |
| --- | --- | --- | --- |
|  | A | B | C |
| 1 [0,1] | -1 | 1 | 1 |
| 2 [0,2,7] | 2 | 2 | 3 |
| 3 [0,2,8] | 2 | 2 | 4 |
| 4 [0,3,9] | 2 | 3 | 2 |
| 5 [0,3,10] | 2 | 4 | 2 |
| 6 [0,4,11] | 3 | 2 | 2 |
| 7 [0,4,12] | 4 | 2 | 2 |
| 8 [0,5,13] | 2 | 2 | 2 |
| 9 [0,6,14] | 2 | 3 | 6 |
| 10 [0,6,15] | 2 | 3 | 4 |

（和上面node一样的测试样例，因为test path是一样的

Prime Path Coverage

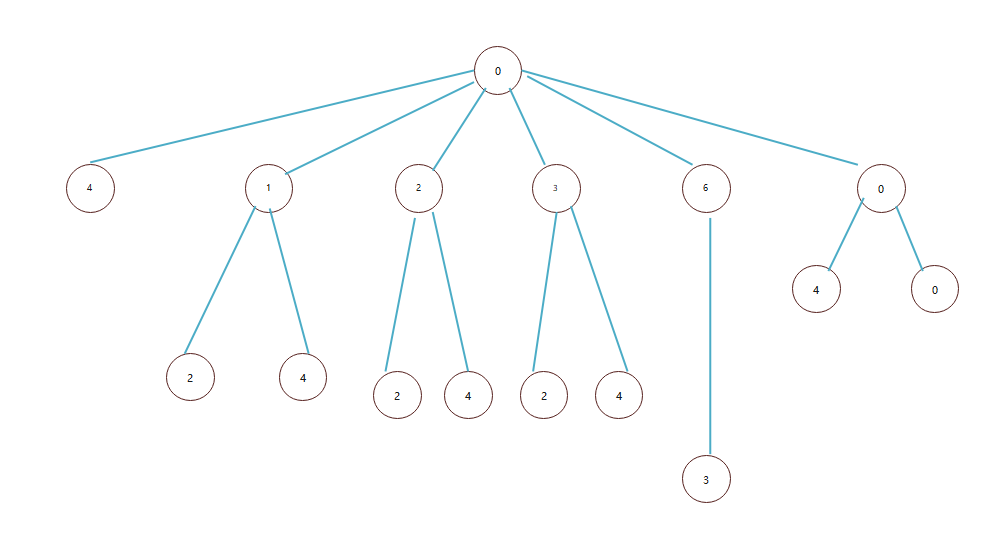
TestPaths:[0,1][0,2,7][0,2,8][0,3,9][0,3,10][0,4,11][0,4,12][0,5,13][0,6,14][0,6,15]

|  |  |  |  |
| --- | --- | --- | --- |
|  | A | B | C |
| 1 [0,1] | -1 | 1 | 1 |
| 2 [0,2,7] | 2 | 2 | 3 |
| 3 [0,2,8] | 2 | 2 | 4 |
| 4 [0,3,9] | 2 | 3 | 2 |
| 5 [0,3,10] | 2 | 4 | 2 |
| 6 [0,4,11] | 3 | 2 | 2 |
| 7 [0,4,12] | 4 | 2 | 2 |
| 8 [0,5,13] | 2 | 2 | 2 |
| 9 [0,6,14] | 2 | 3 | 6 |
| 10 [0,6,15] | 2 | 3 | 4 |

仍然和Node Path一致

Triangle这个程序由于没有循环和交叉，做起来相对简单。

All uses for triOut==4



Test Path = [0,1],[0,2,8],[0,3,10],[0,4,12],[0,6,14]

|  |  |  |  |
| --- | --- | --- | --- |
|  | A | B | C |
| 1 [0,1] | -1 | 1 | 1 |
| 2 [0,2,8] | 2 | 2 | 4 |
| 3 [0,3,10] | 2 | 4 | 2 |
| 4 [0,4,12] | 4 | 2 | 2 |
| 5 [0,6,14] | 2 | 3 | 6 |

给定逻辑表达式

((A=B)V（B<C）)^(A+B)<5

|  |  |  |  |
| --- | --- | --- | --- |
|  | A | B | C |
| 1 | 1 | 1 | 1 |
| 2 | 2 | 3 | 7 |

Predict Coverage

|  |  |  |  |
| --- | --- | --- | --- |
|  | A | B | C |
| 1 | 2 | 2 | 3 |
| 2 | 2 | 3 | 2 |

TTT

FFF

Clause Coverage

|  |  |  |  |
| --- | --- | --- | --- |
|  | A | B | C |
| 1 | 1,2 | 1,2 | 2,3 |
| 2 | 1,3 | 1,3 | 2,3 |
| 3 | 1,3 | 1,2 | 1,3 |
| 4 | 2,2 | 2,2 | 2,3 |
| 5 | 2,3 | 2,3 | 2,3 |
| 6 | 2,3 | 2,2 | 2,3 |
| 7 | 1,2 | 2,2 | 3,3 |
| 8 | 1,3 | 2,3 | 3,3 |
| 9 | 1,3 | 2,2 | 3,3 |

(TTT ,TTF)(TTT,TFF)(TTT,FTF)

(TFT,TTF)(TFT,TFF),(TFT,FTF)

(FTT,TTF)(FTT,TFF)(FTT,FTF)

CACC

对于：Mutantone:

A:2 B:2 C:4

本来应该为4，实际结果是2

对于 Mutantwo

A:2 B:2 C:4

本来应该为4，实际结果是2