

EE360T/382V Software Testing

khurshid@ece.utexas.edu

February 15, 2020

Overview

Today – Continue with logic coverage

Last time – Began logic coverage

Next time – Complete logic coverage

EE360T/382V Software Testing

khurshid@ece.utexas.edu

Chapter 3*: Logic Coverage

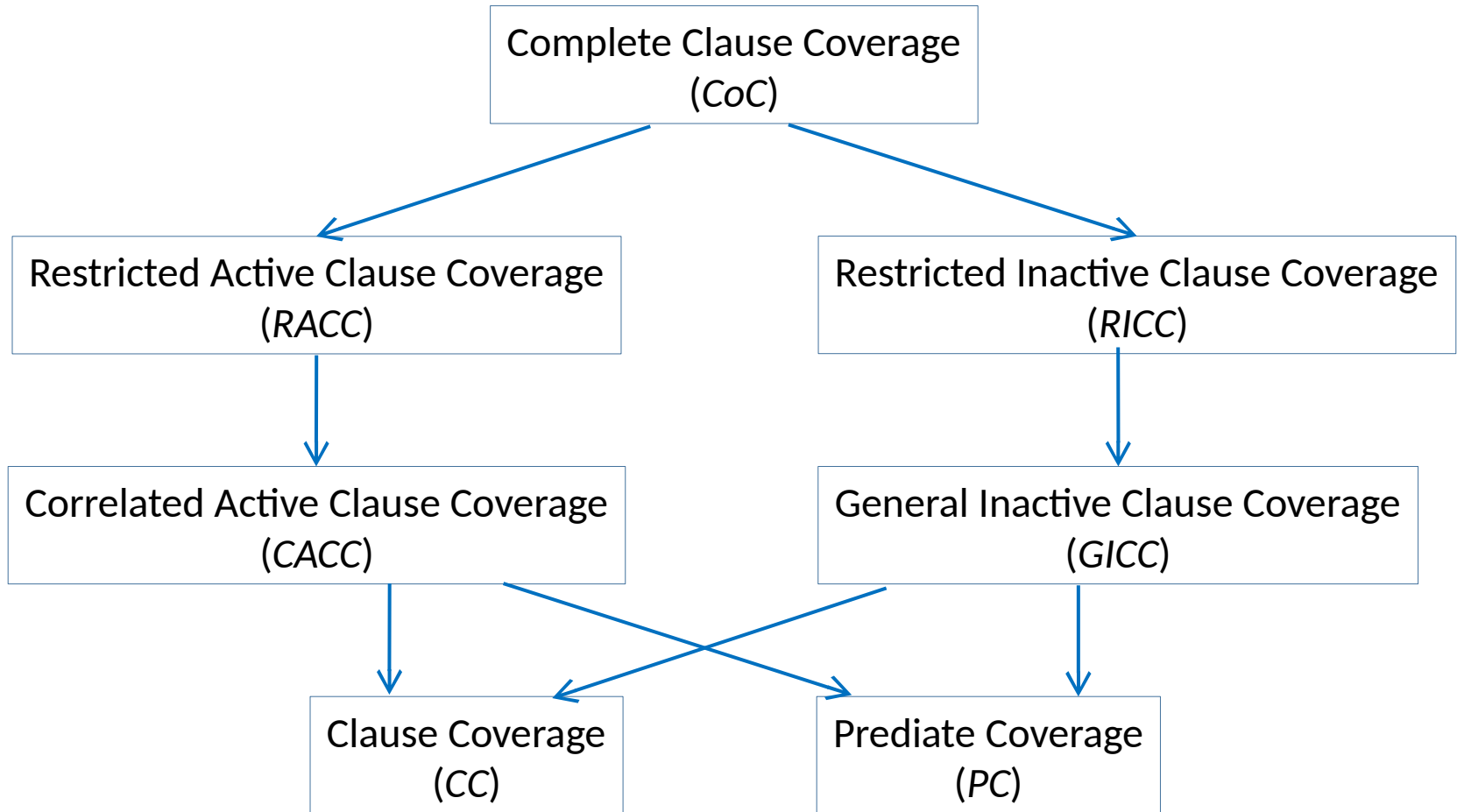
*Introduction to Software Testing by Ammann and Offutt

Criteria based on structures

The textbook focuses on four kinds of structures to define criteria:

- Graphs
 - E.g., control-flow graphs (CFGs)
- Logical expressions
 - E.g., if-conditions
- Input domain characterization
 - E.g., sorted array
- Syntactic structures
 - E.g., mutation

Subsumption



Triangle classification program

Create tests w.r.t. logic coverage criteria:

```
private static int Triang (int Side1, int Side2, int Side3)
{
    int tri_out;
    // tri_out is output from the routine
    //  Triang = 1 if triangle is scalene
    //  Triang = 2 if triangle is isosceles
    //  Triang = 3 if triangle is equilateral
    //  Triang = 4 if not a triangle

    ...
}
```

Predicates in Training method

42: Side1 <= 0 || Side2 <= 0 || Side3 <= 0

49: Side1 == Side2

51: Side1 == Side3

53: Side2 == Side3

55: tri_out == 0

59: Side1+Side2 <= Side3 || Side2+Side3 <= Side1 ||
Side1+Side3 <= Side2

70: tri_out > 3

72: tri_out == 1 && Side1+Side2 > Side3

74: tri_out == 2 && Side1+Side3 > Side2

76: tri_out == 3 && Side2+Side3 > Side1

Reachability of Triang predicates (Table 3.1 in textbook)

42: True

49: P1: $S1 > 0 \ \&\& \ S2 > 0 \ \&\& \ S3 > 0$

51: P1

53: P1

55: P1

59: P1 $\&\& \text{tri_out} == 0$

70: P1 $\&\& \text{tri_out} != 0$

72: P1 $\&\& \text{tri_out} != 0 \ \&\& \ \text{tri_out} \leq 3$

74: P1 $\&\& \text{tri_out} != 0 \ \&\& \ \text{tri_out} \leq 3 \ \&\& \ (\text{tri_out} != 1 \ || \ S1 + S2 \leq 3)$

76: ...

Possible values of tri_out at L55

```
48     tri_out = 0;  
49     if (Side1 == Side2)  
50         tri_out = tri_out + 1;  
51     if (Side1 == Side3)  
52         tri_out = tri_out + 2;  
53     if (Side2 == Side3)  
54         tri_out = tri_out + 3;  
55     if (tri_out == 0)
```

Which of 0, 1, 2, 3, 4, 5, 6 are possible?

Re-writing reachability conditions

Use only input parameters

59: P1 && tri_out == 0
(S1 > 0 && S2 > 0 && S3 > 0) && (S1 != S2 &&
S1 != S3 && S2 != S3)

70: P2: (S1 > 0 && S2 > 0 && S3 > 0) &&
(S1 == S2 || S1 == S3 || S2 == S3)


72: P2 && tri_out <= 3
P2 && (S1 != S2 || S2 != S3 || S1 != S3)

74: ...

Exercise: create tests for PC, CC, CACC

Program transformation issues

Rewriting the program, say to eliminate multi-clause predicates, may modify coverage requirements

```
if ((a && b) || c)       if (a) {  
    s1;                 if (b)  
else                    s1;  
    s2;                 else {  
                        if (c)  
                        s1;  
                        else  
                        s2;  
                    }  
                }  
            }  
        else {  
            if (c)  
                s1;  
            else  
                s2;  
        }  
    }
```

CACC on original vs. PC on modified

	<i>a</i>	<i>b</i>	<i>c</i>	$(a \ \&\& \ b) \ \ c$	CACC	PC
1	T	T	T	T		X
2	T	T	F	T	X	
3	T	F	T	T	X	X
4	T	F	F	F	X	X
5	F	T	T	T		X
6	F	T	F	F	X	
7	F	F	T	T		
8	F	F	F	F		X

PC on the modified program does not subsume CACC on the original program and vice versa

?/!