Measuring Testing Adequency

by
Ramraj S ME.,
Assistant Professor
Department of Software Engineering

2 - Mar -2016

Agenda

- Recap - Code coverage
- Simple condition coverage
- Multiple Condition Coverage

Simple condition coverage

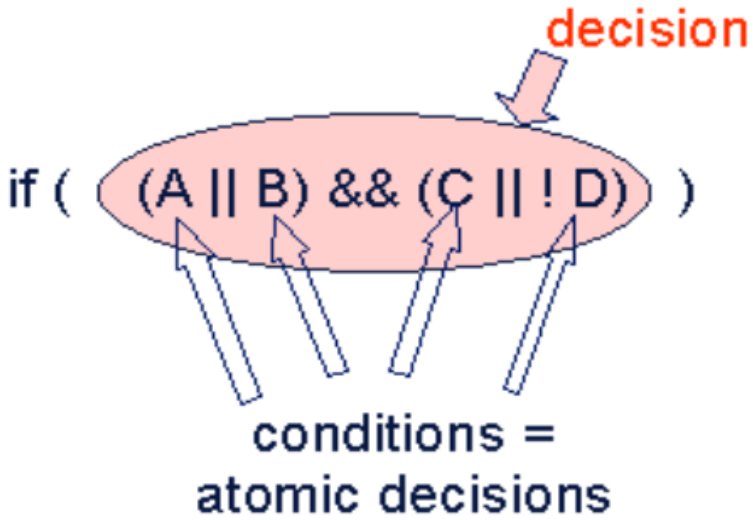


Fig. 13 A decision can be made up of conditions

Figure : Simple Condition Coverage

Complete/Incomplete Evaluation

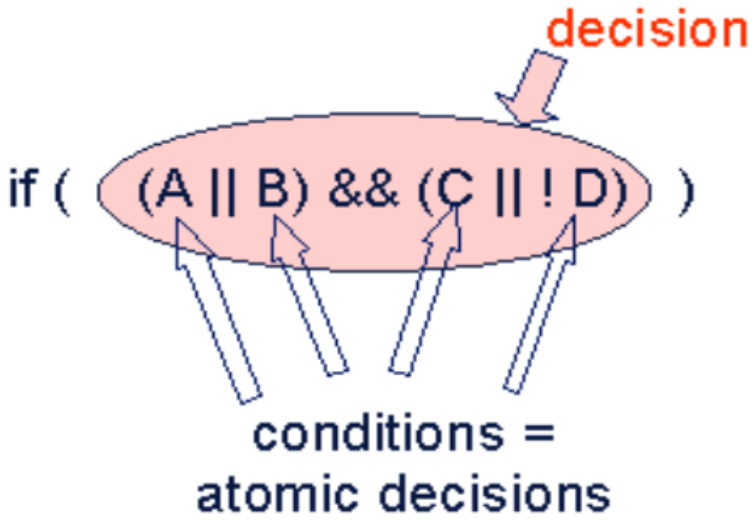


Fig. 13 A decision can be made up of conditions

Figure : Simple Condition Coverage

- In some programming languages C, C++ and Java, the compiler is allowed to stop evaluating a decision as soon as the outcome of the decision is known. This is called incomplete evaluation.
- In the above example if A is false decision value will be false (without checking the other conditions)

Simple condition coverage - Complete evaluation

```
√    IF A AND B THEN
▯          ...
     ELSE
√          ...
```

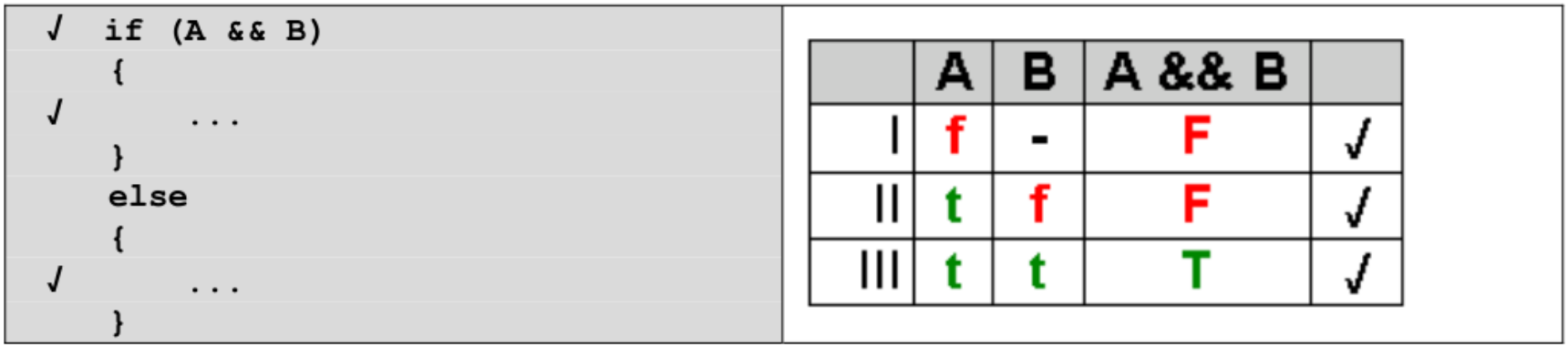
	A	B	A and B	
1	f	f	F	
2	f	t	F	√
3	t	f	F	√
4	t	t	T	

Fig. 14 A code snippet in Pascal and the four possible test cases (complete evaluation)

Figure : Simple Condition Coverage

- Test case 2 and 3 will give 100% simple condition coverage, but not taking decision into account.
- Both test cases execute the else-branch; the if-branch is not executed. This results in only 50% branch coverage

Simple condition coverage - Incomplete evaluation



```
√  if (A && B)
   {
√      ...
   }
   else
   {
√      ...
   }
```

	A	B	A && B	
I	f	-	F	√
II	t	f	F	√
III	t	t	T	√

Fig. 15 A code snippet in C and the three possible test cases (incomplete evaluation)

Figure : Simple Condition Coverage

- In testcase 1 B is not evaluated, i.e. you cannot assume a certain value for B. This is indicated by the -
- Three test cases to get 100% simple condition coverage for (A&&B).
- Gives 100 % branch coverage also.

Minimal Multiple Condition Coverage

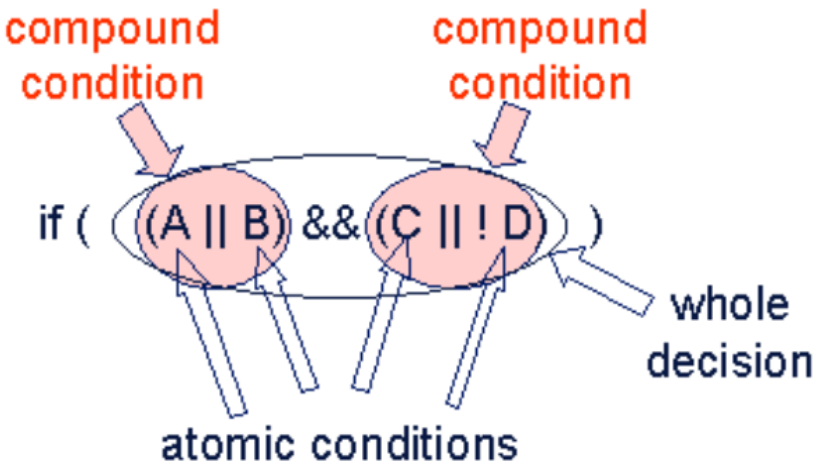


Figure : Multiple Condition Coverage

Minimal multiple condition coverage requires that

1. all atomic conditions are evaluated to both true and false, and
2. all compound conditions are evaluated to both true and false, and
3. the whole decision evaluates to both true and false.

- Minimal multiple condition coverage subsumes condition / decision coverage, because (1) of the list above subsumes simple condition coverage and (3) from the list above subsumes decision coverage.
- Minimal multiple condition coverage is also called as "modified branch condition decision testing"

- Every point of entry and exit in the program has been invoked at least once,
- every condition in a decision in the program has taken all possible outcomes at least once,
- every decision in the program has taken all possible outcomes at least once, and
- each condition in a decision has been shown to independently affect that decisions outcome.
- A condition is shown to independently affect a decisions outcome by varying just that condition while holding fixed all other possible conditions.

As a first example, we consider the decision (A || B) && (C || D), for a language with complete evaluation. There are 16 combinations for A, B, C, and D.

	A	B	C	D	A or B	C or D	(A or B) and (C or D)
1	f	f	f	f			
2	f	f	f	t	f	t	f
3	f	f	t	f			
4	f	f	t	t			
5	f	t	f	f			
6	f	t	f	t	t	t	t
7	f	t	t	f			
8	f	t	t	t			
9	t	f	f	f	t	f	f
10	t	f	f	t	t	t	t
11	t	f	t	f	t	t	t
12	t	f	t	t			
13	t	t	f	f			
14	t	t	f	t			
15	t	t	t	f			
16	t	t	t	t			

B

D

C

A

A: 2+10
B: 2+6
C: 9+11
D: 9+10

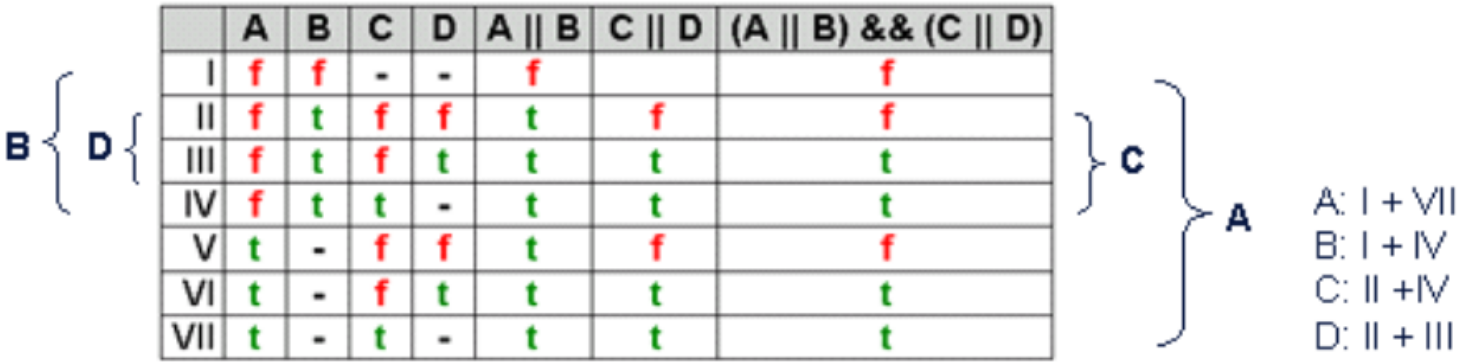
Fig. 17 The truth table and a set of pairs for 100% M/DC (complete evaluation)

Figure : Multiple Condition Coverage

Test case 9 and 10 build a valid pair for the condition D because

- they differ in the value for the condition in question (D is false in test case 9 and D is true in test case 10)
- the value of the other three conditions (A, B, and C) are identical for both test cases
- the overall outcome for the decision differs,i.e.((A or B)&&(C or D)) is false in test case 9 and true in test case 10.

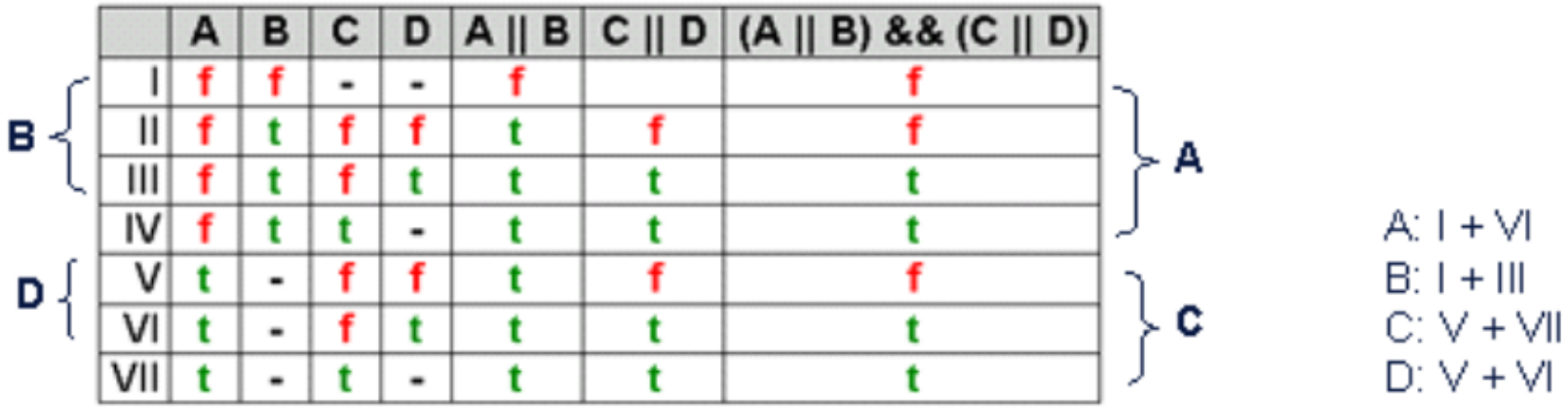
As a second example, we take the same decision, but with incomplete evaluation.



	A	B	C	D	A || B	C || D	(A || B) && (C || D)
I	f	f	-	-	f		f
II	f	t	f	f	t	f	f
III	f	t	f	t	t	t	t
IV	f	t	t	-	t	t	t
V	t	-	f	f	t	f	f
VI	t	-	f	t	t	t	t
VII	t	-	t	-	t	t	t

Fig. 18 The truth table and a set of pairs for 100% M/DC (incomplete evaluation)

Figure : Multiple Condition Coverage



	A	B	C	D	A || B	C || D	(A || B) && (C || D)
I	f	f	-	-	f		f
II	f	t	f	f	t	f	f
III	f	t	f	t	t	t	t
IV	f	t	t	-	t	t	t
V	t	-	f	f	t	f	f
VI	t	-	f	t	t	t	t
VII	t	-	t	-	t	t	t

B: II, III — A: I, II, III, IV — D: V, VI — C: V, VI, VII

A: I + VI
B: I + III
C: V + VII
D: V + VI

Fig. 19 Another set of pairs of test cases that yield 100% MC/DC for the decision

Figure : Multiple Condition Coverage

References I

[1] Torbjrn Ryber *"ESSENTIAL SOFTWARE TEST DESIGN "*, Chapter 4 Chapter 5

Thank you