White-box testing techniques

Dr. Sangharatna Godboley
Assistant Professor
Department of CSE
NIT Warangal

Slides Credits: Dr. Durga Prasad Mohapatra, Professor, Department of CSE, NIT Rourkela

White-box Testing

- Designing white-box test cases:
  - Requires knowledge about the internal structure of software.
  - White-box testing is also called structural testing.
  - In this unit we will study white-box testin

White-Box Testing Methodologies

- There exist several popular white-box testing methodologies:
  - Statement coverage
  - Branch coverage
  - Condition coverage
  - MC/DC coverage
  - Path coverage
  - Data flow-based testing
  - Mutation testing

Statement Coverage

- Statement coverage methodology:
  - Design test cases so that every statement in the program is executed at least once.

Statement Coverage

- The principal idea:
  - Unless a statement is executed,
  - We have no way of knowing if an error exists in that statement.

Statement Coverage Criterion

- Observing that a statement behaves properly for one input value:
  - No guarantee that it will behave correctly for all input values.

Statement Testing

- Coverage measurement:

$$\frac{\#\text{ executed statements}}{\#\text{ statements}}$$

- Rationale: a fault in a statement can only be revealed by executing the faulty statement

Example

```
int f1(int x, int y){
1 while (x != y){
2     if (x>y) then
3           x=x-y;
4     else y=y-x;
5 }
6 return x;          }
```

Euclid's GCD Algorithm

Euclid's GCD Algorithm

- By choosing the test set {(x=3,y=3),(x=4,y=3), (x=3,y=4)}
  - All statements are executed at least once.

Branch Coverage

- Test cases are designed such that:
  - Different branch conditions
    - Given true and false values in turn.

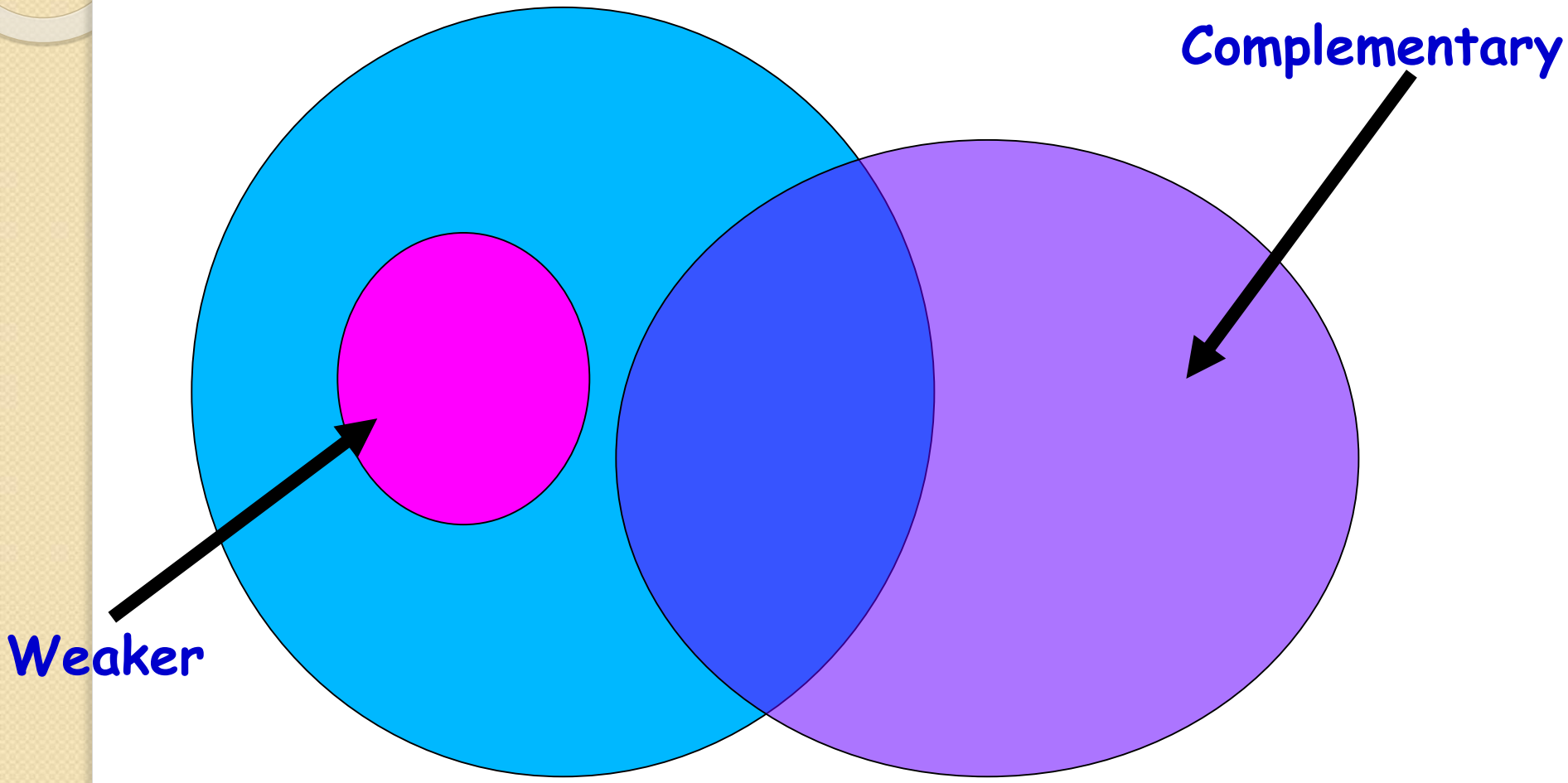
Branch Coverage

- Branch testing guarantees statement coverage:
  - A stronger testing compared to the statement coverage-based testing.

Stronger Testing

- Test cases are a superset of a weaker testing:
  - A stronger testing covers at least all the elements of the elements covered by a weaker testing.

Stronger, Weaker, and Complementary Testing



Example

```
int f1(int x,int y){
1 while (x != y){
2     if (x>y) then
3           x=x-y;
4     else y=y-x;
5 }
6 return x;          }
```

Example

- Test cases for branch coverage can be:
- {(x=3,y=3),(x=3,y=2), (x=4,y=3), (x=3,y=4)}

Branch Testing

- Adequacy criterion: Each branch (edge in the CFG) must be executed at least once
- Coverage:

$$\frac{\#\ \text{executed branches}}{\#\ \text{branches}}$$

Statements vs Branch Testing

- Traversing all edges of a graph causes all nodes to be visited
  - So test suites that satisfy the branch adequacy criterion for a program P also satisfy the statement adequacy criterion for the same program
- The converse is not true
  - A statement-adequate (or node-adequate) test suite may not be branch-adequate (edge-adequate)

Condition Coverage

- Test cases are designed such that:
  - Each component of a composite conditional expression
    - Given both true and false values.

Condition Coverage Examples

```
if(A && B)
  F1();
  else
  F2();
 if(C)
  F3()
  else
  F4();
```

Test Cases for Condition Coverage:

A=T, B=T, C=F

A=F, B=F, C=T

```
if(A && B)
  F1();
else
  F2();
if(C)
  F3()
else
  F4();
```

Test Cases for Condition Coverage:

A=T, B=F, C=F

A=F, B=T, C=T

Problem: Not decision/branch coverage achieved

Branch vs Condition testing

- Condition testing:
  - Stronger testing than branch testing.
- Branch testing:
  - Stronger than statement coverage testing.

Branch vs Condition testing

- Branch testing is the simplest condition testing strategy:
  - Compound conditions appearing in different branch statements
    - Are given true and false values.

Multiple Condition Coverage

- Multiple condition coverage reports whether every possible combination of Boolean sub-expressions occurs.
- The test cases required for full multiple condition coverage of a condition are essentially given by the logical operator truth table for the condition.

Multiple Condition Coverage cont …

- Test cases are designed such that:
  - Each component of a composite conditional expression
    - Given both true and false values.

Example

- Consider the conditional expression
  - ((c1.and.c2).or.c3):
- Each of c1, c2, and c3 are exercised at least once,
  - i.e. given true and false values.

Multiple Condition Coverage Examples

```
if(A && B)   // condition 1
   F1();
 else
   F2();
if(C || D)      // condition 2
   F3()
 else
   F4();
```

Test Cases for MCC:

For condition 1	For condition 2
A=T, B=T	C=T, D=T
A=T, B=F	C=T, D=F
A=F, B=T	C=F, D=T
A=F, B=F	C=F, D=F

MCC Testing Adequacy Criterion

- Adequacy criterion: Each basic condition must be executed at least once
- Coverage:

$$\frac{\#\text{ truth values taken by all basic conditions}}{2 * \#\text{ basic conditions}}$$

Branch Testing vs MCC Testing

- Multiple Condition Coverage testing:
  - Stronger testing than branch testing.
- Branch testing:
  - Stronger than statement coverage testing.

Need of a Feasible Testing Technique

- Consider a boolean expression having **n** components:
  - For condition coverage we require 2^n test cases. (example in next slide)
- Condition coverage-based testing technique:
  - Practical only if **n** (the number of component conditions) is small.

Consider: if (a || b && c) then …

Test	a	b	c
(1)	T	T	T
(2)	T	T	F
(3)	T	F	T
(4)	T	F	F
(5)	F	T	T
(6)	T	T	F
(7)	F	F	T
(8)	F	F	F

MCC

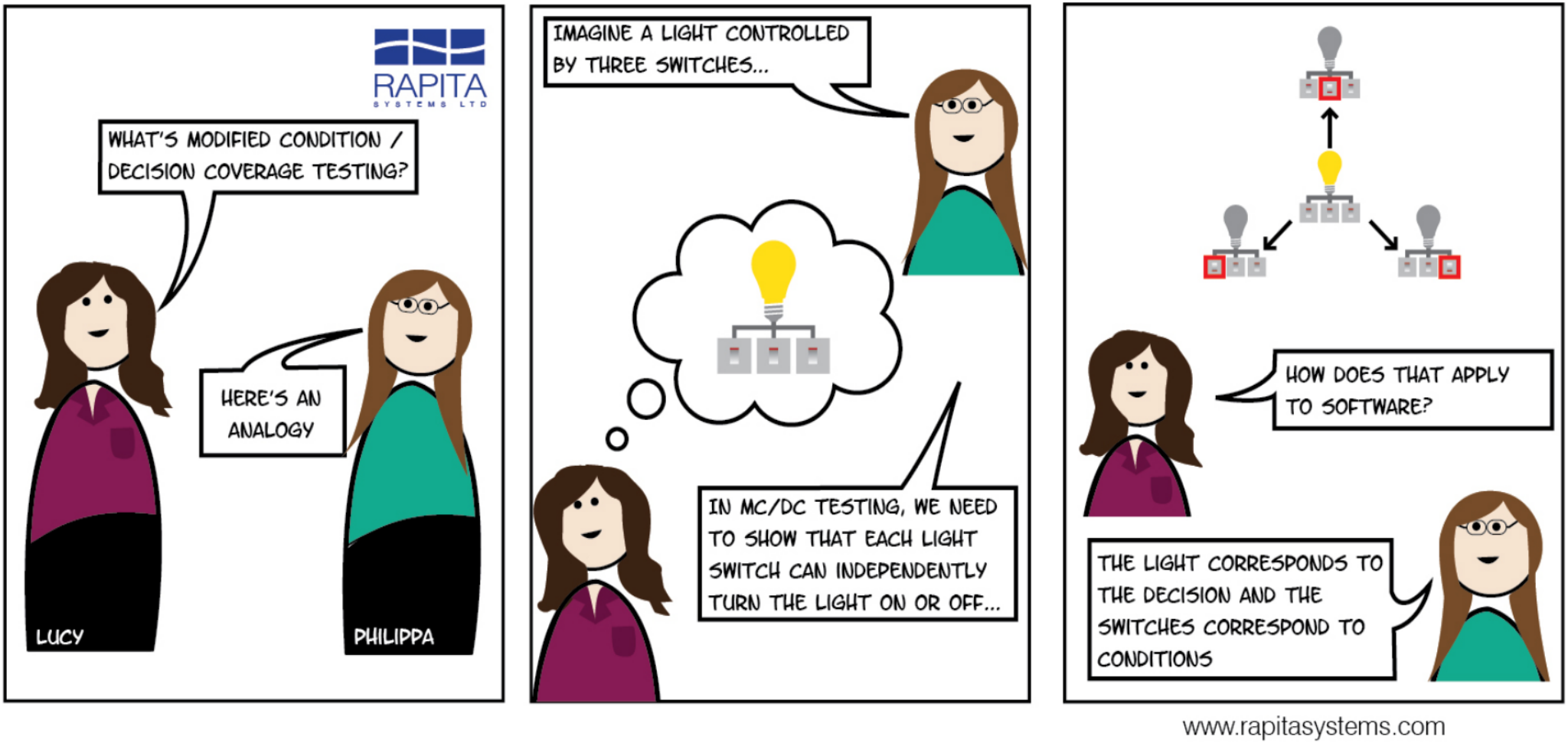
Exponential in the number of basic conditions

Modified condition/decision (MC/DC)

- Motivation: Effectively test important combinations of conditions, without exponential blowup in test suite size
  - "Important" combinations means: Each basic condition shown to independently affect the outcome of each decision
- Requires:
  - For each basic condition C, two test cases,
  - values of all *evaluated* conditions except C are the same
  - compound condition as a whole evaluates to *true* for one and *false* for the other

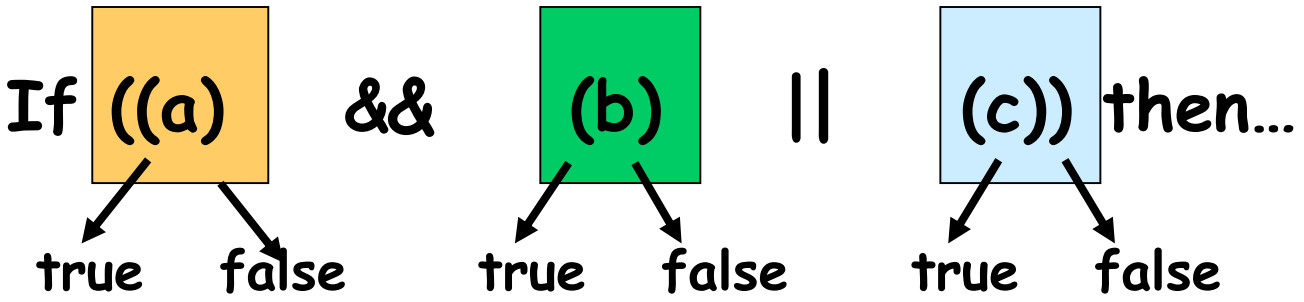
What is MC/DC?

- MC/DC stands for **M**odified **C**ondition / **D**ecision **C**overage
- A kind of Predicate Coverage technique
  - Condition: Leaf level Boolean expression.
  - Decision: Controls the program flow.
- Main idea: Each condition must be shown to independently affect the outcome of a decision, i.e. the outcome of a decision changes as a result of changing a single condition.


RAPITA
SYSTEMS LTD
WHAT'S MODIFIED CONDITION / DECISION COVERAGE TESTING?
HERE'S AN ANALOGY
LUCY
PHILIPPA
IMAGINE A LIGHT CONTROLLED BY THREE SWITCHES...
IN MC/DC TESTING, WE NEED TO SHOW THAT EACH LIGHT SWITCH CAN INDEPENDENTLY TURN THE LIGHT ON OR OFF...
HOW DOES THAT APPLY TO SOFTWARE?
THE LIGHT CORRESPONDS TO THE DECISION AND THE SWITCHES CORRESPOND TO CONDITIONS
www.rapitasystems.com

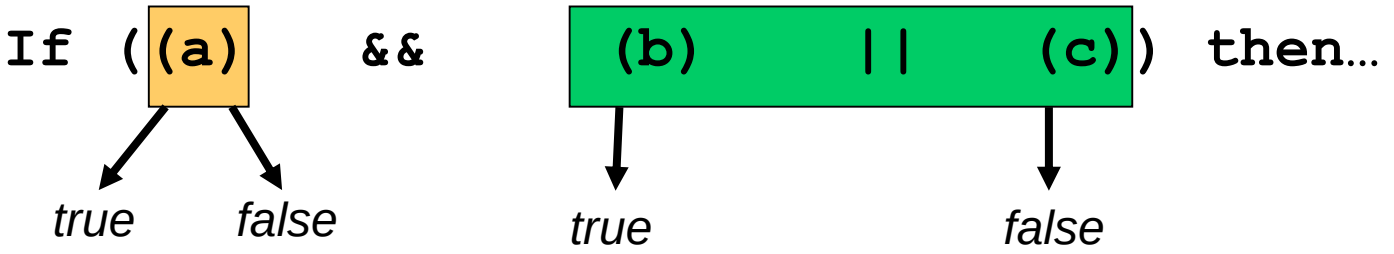
Condition Coverage

- Every condition in the decision has taken all possible outcomes at least once.
-



MC/DC Coverage

- Every condition in the decision independently affects the decision's outcome.



Change the value of each condition individually while keeping all other conditions constant.

MC/DC: linear complexity

- N+1 test cases for N basic conditions

(((a || b) && c) || d) && e

Test Case	a	b	c	d	e	outcome
(1)	true	--	true	--	true	true
(2)	false	true	true	--	true	true
(3)	true	--	false	true	true	true
(6)	true	--	true	--	false	false
(11)	true	--	false	false	--	false
(13)	false	false	--	false	--	false

- Underlined values independently affect the output of the decision
- Required by the RTCA/DO-178B standard

Comments on MC/DC

- MC/DC is
  - basic condition coverage (C)
  - branch coverage (DC)
  - plus one additional condition (M): every condition must *independently affect* the decision's output
- It is subsumed by compound conditions and subsumes all other criteria discussed so far
  - stronger than statement and branch coverage
- A good balance of thoroughness and test size (and therefore widely used)

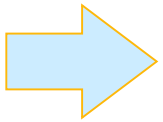
Example

If (A && B) then…

(1) create truth table for conditions.

(2) Extend truth table so that it indicated which test cases can be used to show the independence of each condition.

A B	Result
T T	T
T F	F
F T	F
F F	F



Number	A B	Result	A	B
1	T T	T	3	2
2	T F	F		1
3	F T	F	1	
4	F F	F		

Example cont …

Number	A	B	Result	A	B
1	T	T	T	3	2
2	T	F	F		1
3	F	T	F	1	
4	F	F	F		

- Show independence of **A**:
  - Take 1 + 3
- Show independence of **B**:
  - Take 1 + 2
- Resulting test cases are
  - 1 + 2 + 3
  - (T , T) + (T , F) + (F , T)

More advanced example

```
If (A && (B || C)) then...
```

Number	A B C	Result	A	B	C
1	T T T	T	5		
2	T T F	T	6	4	
3	T F T	T	7		4
4	T F F	F		2	3
5	F T T	F	1		
6	F T F	F	2		
7	F F T	F	3		
8	F F F	F			

More advanced example contd..

Note: We want to determine the MINIMAL set of test cases

Here:

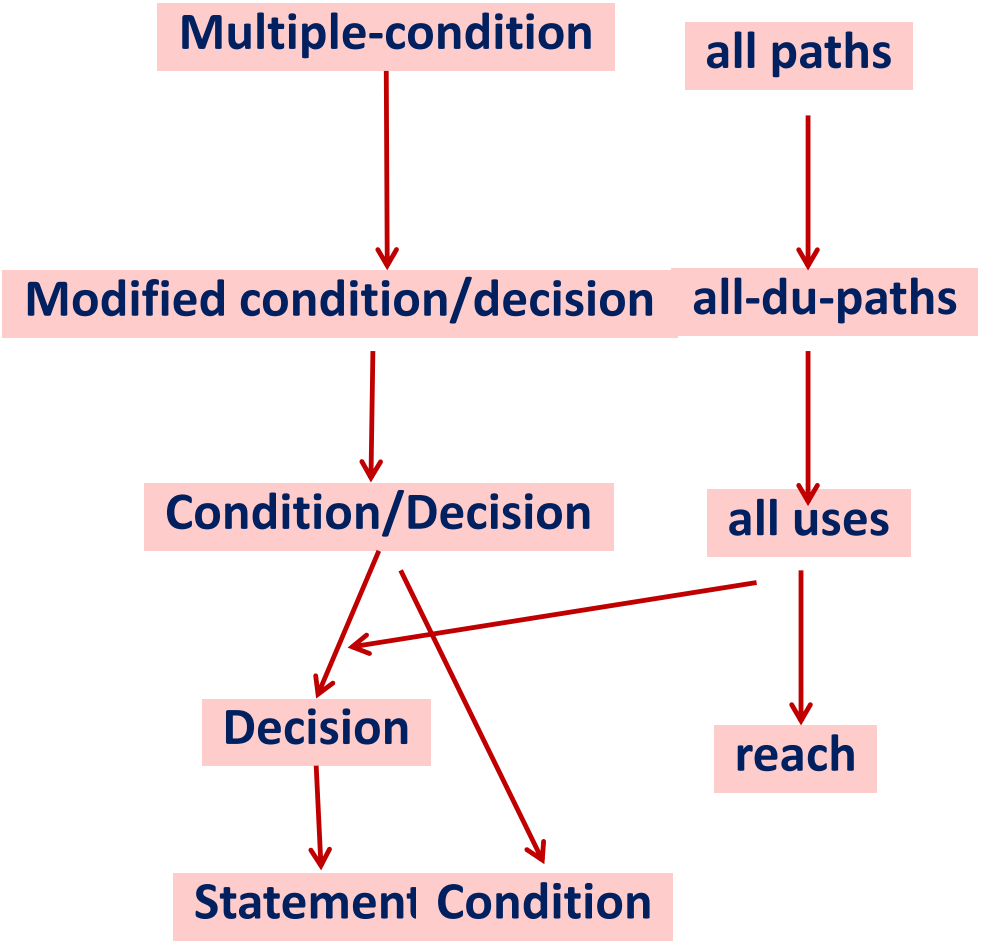
- {2,3,4,6}
- {2,3,4,7}

Non-minimal set is:

- {1,2,3,4,5}

Where does it fit in?

- The MC/DC criterion is much stronger than the condition/decision coverage criterion, but the number of test cases to achieve the MC/DC criterions still varies linearly with the number of conditions **n** in the decisions.
  - Much more complete coverage than condition/decision coverage, but
  - at the same time it is not terribly costly in terms of number of test cases.



Comparison of different coverage based Testing Strategies

Coverage Criteria	Statement Coverage	Decision Coverage	Condition Coverage	Condition/ Decision Coverage	MC/DC	Multiple Condition Coverage
Every point of entry and exit in the program has been invoked at least once		•	•	•	•	•
Every statement in the program has been invoked at least once	•					
Every decision in the program has taken all possible outcomes at least once		•		•	•	•
Every condition in a decision in the program has taken all possible outcomes at least once			•	•	•	•
Every condition in a decision has been shown to independently affect that decision's outcome					•	•[8]
Every combination of condition outcomes within a decision has been invoked at least once						•

weakest → strongest

Thank You