

**Software Engineering**  
**Prof. Rajib Mall**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 58**  
**MC/DC Testing**

Welcome to this lecture in the last lecture, we were discussing about the MC DC Testing.



(Refer Slide Time: 00:26)

**Creating MC/DC Test Cases: An Example**

- N+1 test cases required for N basic conditions
- Example:  
$$(((a>10 \ || \ b<50) \ \&\& \ c==0) \ || \ d<5) \ \&\& \ e==10)$$

Test Case	a>10	b<50	c==0	d<5	e==10	outcome
(1)	<u>true</u>	false	<u>true</u>	false	<u>true</u>	<b>true</b>
(2)	false	<u>true</u>	true	false	true	<b>true</b>
(3)	true	false	false	<u>true</u>	true	<b>true</b>
(6)	true	false	true	false	<u>false</u>	<b>false</b>
(11)	true	false	<u>false</u>	<u>false</u>	true	<b>false</b>
(13)	<u>false</u>	<u>false</u>	true	false	true	<b>false</b>

- Underlined values independently affect the output of the decision

We said that it is a very important test criterion, it is particle and also it is a it gives a thorough testing and therefore, very popular. Now, let us proceed with what we were discussing last time. We had seen the basic definitions of the MC DC testing. And, we had seen few examples about how to develop MC DC test cases given a conditional statement, but remember that there may be several conditional statements in a program and of course, we need to design for every conditional statement. Now, let us look at more systematic way to develop the MC DC test cases.

One thing need to mention that here, if a conditional expression has N sub expressions here we have 5 sub expressions 1 2 3 4 5. Then for multiple condition coverage MCC, we need 2 to the power 5 which is 32, but for MC DC test cases it will be just 6 5 plus 1 is 6. And, if there are 10 basic conditions or sub expressions, then for multiple condition coverages we need 2 to the person 10 which is a 1024 test cases, but for in MC DC

testing we need 11 test cases. We will not give any proof for N plus 1 test cases required for n basic conditions we will just take this result.

Now, let us look at this example expression. And, see how we go about creating the MC DC test cases. The first step of course, is to draw the truth table as the different sub expressions or the basic conditions change the truth values. All possible combinations of truth value we need to explore, but that will be 2 to the power 5, which is 32 rows in the truth table. But this screen is small I just drawn only the important ones I will not included all the possible combinations. I have omitted the ones which are do not I had not required for inclusion in the MC DC test suite.

Once, we draw the complete truth table that is all possible combinations of the truth values of the basic conditions and the corresponding outcome. We need to check for every basic condition, which are the 2 rows, where the truth value of the basic expression is true and false. And, the output also toggles true and false and the remaining conditions basic conditions have the same value just see here false true false true and for this also false true false true. So, the other conditions truth value of other conditions remaining constant, a change of the truth value of the condition under consideration as it toggles from true to false the outcome toggles from true to false.

Similarly, let us look at the second clause. We check all rows and see that when it is false the outcome is true, but when it is true here the outcome is still true? So, these two cannot be the once, we just check all possible combinations here. And then we find that the second one along with the thirteenth-one, as the change from true to false is the condition b less than 50 changes from true to false. And, the other once remaining constant true false true false true, false true and this is false true, false true.

As, b less than 50 toggles from true to false the outcome toggles from true to false. So, we need 113 to so, the independent evaluation of the first condition basic condition. So, this shows the independent this first condition, independently toggles the outcome, when the others are remaining constant. Similarly 2 and 13 toggles the outcome when the rest are held constant. Similarly the third-one, as it toggles from true or false the outcome toggles from true to false, the other things remaining constant.

Similarly, the fourth-one as it toggles from true to false, the outcome toggles from true to false, false rest are held constant. And, similarly the last one as it toggles from true to

false. The outcome toggles from true to false. Now, we need all this out of the 32 rows, we need 1 2 3 4 5 6. We need 6 test cases, for achieving MC DC coverage and we have 5 basic conditions and we need 6 test cases.

(Refer Slide Time: 07:27)

Page 47/42

**Systematically Creating MC/DC test cases**

- Create truth table for conditions.
- Extend the truth table to represent test case pair that lead to show the independence influence of each condition.


**Example : If ( A and B ) then . . .**

Test Case Number	A	B	Decision	Test case pair for A	Test case pair for 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

{1,2,3}

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES



Now, let us see, how we can systematically create the MC DC test cases? Let us take an example; we will this is the example A and B. A and B are Boolean variables and we need to create the truth table of course, but then will create a extended truth table. In the extended truth table for each basic condition, we will keep a note of which 2 rows independently influence the output for a specific basic condition. Let us first see how to have the extended truth table? We have the truth table here a numbered the rows 1 2 3 4 there are only 2 basic conditions. So, A 4 rows in the truth table and this is the outcome.

And, then we just check here, that when A is true and B is true, the outcome is true. Now, if A is false and the B is remaining true, the outcome is false. So, 1 along with 3 will show the independent influence of this condition, first condition on the outcome. And, as we go round as we go check all the possible rows of course, we will find 3 along with 1 also independently influences the outcome. And, the other combinations are not able to independently influence the outcome, but for larger sub expressions and other examples, we will see that there may be several pairs of rows which independently so, which so, the independent influence of the first condition on the outcome.

Similarly, we check the second condition. We see that when A is true and B is true the outcome is true and in the next one we say that A is held true, but B becomes F the outcome toggles. And therefore, 1 along with 2 will show the independent influence of B on the outcome.

Similarly, 2 with 1 will also so, the independent evaluation. Ah. So, we can take either 1 3 sorry we have to take 1 3 tests 1 3 for independent influence of A on the output. And, 1 2 for the independent influence of B on the output, and our MC DC test suite will be the first row, second row, and third row. So, we had so, this is the MC DC test suite. So, we had 2 basic conditions and we need 3 test cases ok. The test suit is 1 2 and 3.

(Refer Slide Time: 11:25)

Page 41/44




### Systematically Creating MC/DC test cases

- Create truth table for conditions.
- Extend the truth table to represent test case pair that lead to show the independence influence of each condition.

**Example : If ( A and B ) then . . .**

Test Case Number	A	B	Decision	Test case pair for A	Test case pair for 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



Now, let us look at another example.

(Refer Slide Time: 11:34)

**If  $(A \vee B) \wedge C$  ....**

	A	B	C	Result	A	B	C	MC/DC
1	1	1	1	1			*	*
2	1	1	0	0			*	*
3	1	0	1	1	*			*
4	0	1	1	1		*		*
5	1	0	0	0				
6	0	1	0	0				
7	0	0	1	0	*	*		*
8	0	0	0	0				

**Another Example**

This is slightly larger example, 3 Boolean variables A B and C. And therefore, we need 2 to the power 3 rows in the truth table I drawn the truth table here, extended truth table where not only drawn the truth table, but also we keeping note of which rows result in independent influence of A on the outcome, independent influence of B on the outcome, independent influence of C on the outcome. And, then we will identify the MC DC test suite. So, this is the extended table.

Now, let us take this the first Boolean variable A. Now, as we scan through here A is 1, when B C and held at 1 result is 1. Now, let us see where B C are held at 1 here, but then this does not help A has toggled, but the outcome has not toggled. So, that does not help us. Now, let us check the second one. When A is 1, B C is 1 0, outcome is 0 now let us check where 1 0. So, this also does not help us. Now, let us check the third-one. So, A is 1 B C are held at 0 1 outcome is 1, now see let us see where it is 0 1 ok. So, A has toggled from 1 to 0 and B C are remaining 0 and the outcome has toggled. And therefore, we select these 2 these 2 so, the independent influence of a on the outcome.

Similarly, if we scan through for b we will find that 4 and 7, that so, the independent influence of B on the outcome of the expression evaluation. Similarly, for C if we scan through the truth table, we will see that 1 and 2 will show independent influence of C on the outcome just C here, C has toggled from 1 to 0 outcome has toggled from 1 to 0 the rest are A B are held at 1 1. So, this two will do.

Now, let us create the MC DC test suite, we need 1 that will show 1 and 2 that will show the independent influence of C we need 3 4 and we need 7. So, that is 1 2 3 4 5. So, that is 5 test cases ok. So, it is linear in the number of test suite even though it is not N plus 1, because here that does not hold, but it will be around that.

(Refer Slide Time: 15:19)

If (A and (B or C)) then...

TC#	ABC	Result	A	B	C
1	TTT	T	5		
2	TTF	T	6	4	
3	TFT	T	7		4
4	TFF	F		2	3
5	FTT	F	1		
6	FTF	F	2		
7	FFT	F	3		
8	FFF	F			

**Minimal Set Example**



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}

Now, let us see when there are multiple rows can show the independent influence of a variable will have alternate possibilities for the MC DC test suite. And, we need to select the minimal test suite in that case. Let us look at this expression A B C there are 3 A B C are the Boolean variables, and there are 2 to the power 3 is 8 rows in the truth table, and we have written the truth table here. We see here that 1 along with 5 shows the independent influence of A on the outcome.

Similarly, 2 along with 6 also shows the independent influence of the basic condition A on the outcome. Similarly 3 along with 7 also shows independent influence of A on the outcome. And, naturally will have 5 and 1 6 and 2 and 7 and 3. Thus the same thing here, because this is 1 5 2 6 and 3 7 we have 5 1 6 2 7 3. So, this of course, can be easily find out based on this.

Now, let us see the independent influence B on the outcome, we have 2 4 and of course, 4 2. Similarly, for C it is 3 4 4 3 now when we choose the MC DC test suite we can find that 2 3 4 and 6. So, 2 6 4 3 and 4 2 this has to be included anyway, 4 3 and 4 2 will be there and we can either choose 2 and 6 or 5 and 1 etcetera.

So, if we see select 2 and 6 for the A and then we have these 2 possibilities, 2 3 4 6 and 2 3 4 7 and also there is a non-minimal set where we include 2 4 4 2 and 3 4 4 2 and 3 4, that has to be there 4 2 and 3 4 and also we include 1 5. So, this is a non-minimal set requiring 5 test cases, but we can have the minimal set, which is 4 test cases.

(Refer Slide Time: 18:47)

**Observation**

- MC/DC criterion is stronger than condition/decision coverage criterion,
  - but the number of test cases to achieve MC/DC still linear in the number of conditions  $n$  in the decisions.

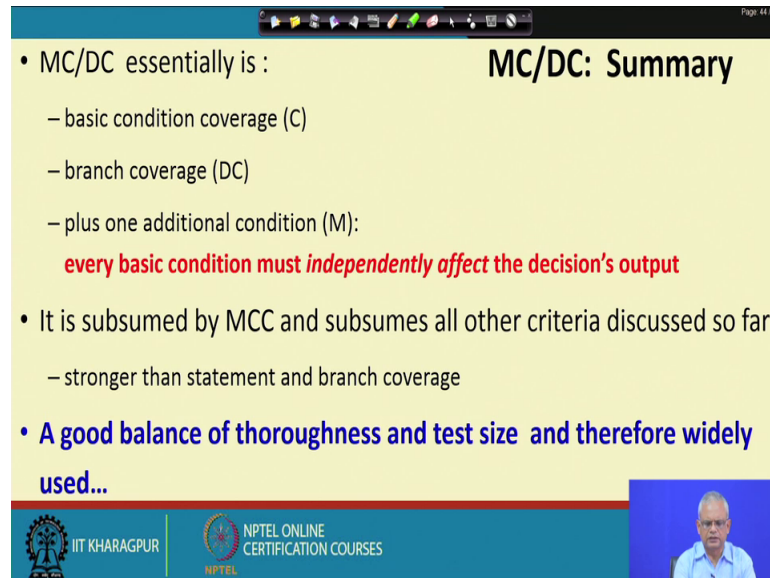
MCC  
 ↓  
 MC/DC  
 ↓  
 Condition/Decision  
 ↓  
 Decision  
 ↓  
 Statement

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Now, let us note down our observation, MC DC criterion MC ORS condition decision coverage and also it shows the independent influence of every basic condition and the outcome. The number of test cases is linear in the number of basic conditions. Typically for  $n$  basic condition we need  $n$  plus 1 test cases. Of course, sometimes we have non minimal test cases where we need 1 more, but this is a very strong testing.

It subsumes the condition decision coverage, which in turn subsumes decision coverage and which subsumes statement coverage. Of course, MC DC is subsumed by MCC, but MCC is in practical. So, MC DC is a strong enough test criterion and also requires small number of test cases and therefore, it is very popular.

(Refer Slide Time: 20:07)



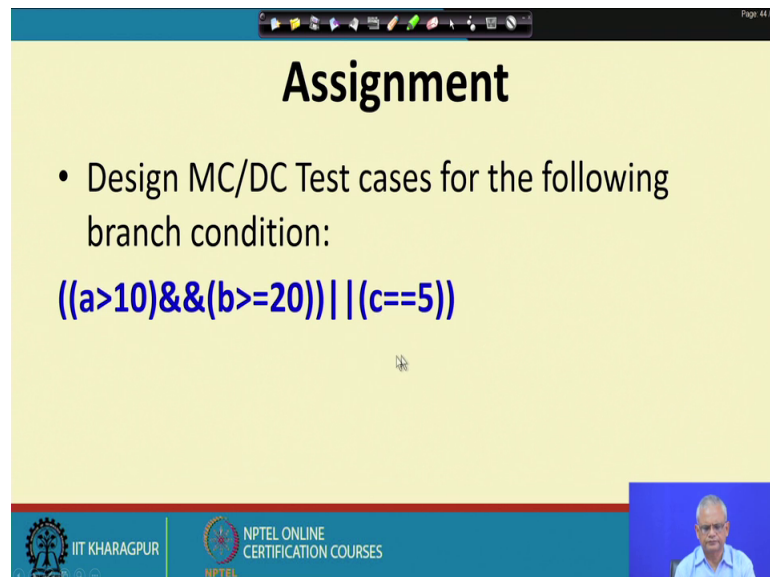
The slide is titled "MC/DC: Summary" and contains the following content:

- MC/DC essentially is :
  - basic condition coverage (C)
  - branch coverage (DC)
  - plus one additional condition (M):  
**every basic condition must independently affect the decision's output**
- It is subsumed by MCC 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...**

The slide footer includes the IIT KHARAGPUR logo, NPTEL ONLINE CERTIFICATION COURSES logo, and a small video inset of a speaker.

So, MC DC just to summarize, we require basic condition coverage, branch coverage, and also we show that every condition independently affects the decisions output. It is subsume by MCC and subsumes all other criteria, and is stronger than statement and branch coverage it is a good balance of thoroughness and test sizes widely used.

(Refer Slide Time: 20:40)



The slide is titled "Assignment" and contains the following content:

- Design MC/DC Test cases for the following branch condition:  
**`((a>10)&&(b>=20))||(c==5)`**

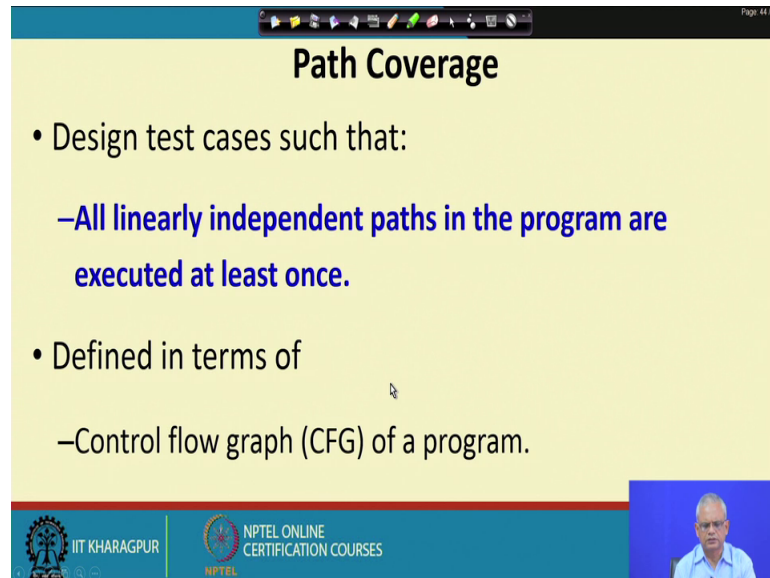
The slide footer includes the IIT KHARAGPUR logo, NPTEL ONLINE CERTIFICATION COURSES logo, and a small video inset of a speaker.

Here is just a small assignment for you just note it down a simple branch condition having 3 basic conditions. And, please develop the MC DC test suite, you can of course,



frame further branch conditions on your own involving 3 4 5 basic conditions, develop the truth table and decide on the MC DC test suite.

(Refer Slide Time: 21:24)



The slide is titled "Path Coverage" and contains the following content:

- Design test cases such that:
  - All linearly independent paths in the program are executed at least once.
- Defined in terms of
  - Control flow graph (CFG) of a program.

The slide also features logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES at the bottom, and a small video inset of a speaker in the bottom right corner.



Now, let us look at path testing. Path testing is based on the control flow, the main idea here is that we need to design test cases such that, all linearly independent paths, all linearly independent paths are executed at least once. So, we need to define this term, what we mean linearly independent paths. And, as mentioning the paths in a program are defined in terms of the control flow graph of a program.

So, we need to first discuss, how to draw the control flow graph a program. Or the CFG of the program, and once we are able to draw the control flow graph of a program, we will see what are the linearly independent paths and, how to estimate? The number of linearly independent paths in a program and that will tell us about the number of test cases required to achieve path coverage.

(Refer Slide Time: 22:50)

### Path Coverage-Based Testing

- To understand path coverage-based testing:
  - We need to learn how to draw control flow graph of a program.
- What is a control flow graph (CFG)?
  - **The sequence in which different instructions of a program get executed.**
  - The way control flows through the program.





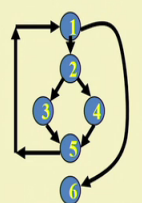
The most basic question here is at what exactly is the control flow graph? What does it indicate? The control flow graph indicates how the control flows through the program, which in turn is the sequence in which the different instructions of the program get executed, when a test case is executed?

(Refer Slide Time: 23:23)

### How to Draw Control Flow Graph?

- **Number all statements of a program.**
- Numbered statements:
  - Represent nodes of control flow graph.
- Draw an edge from one node to another node:
  - **If execution of the statement representing the first node can result in transfer of control to the other node.**

```
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; }
```



Now, the next question is how to draw the control flow graph? The first step given a unit or a function, the first step is to number all the statements. We can develop the control flow graph either for a segment of a unit or the full unit; we have not numbered the first

statement here, because anyway that will get executed. And, the rest once we have numbered here 1 2 3 4 5 6, and these numbered statements become nodes in the flow graph. Now, the we draw the edges between the nodes. So, this we draw the nodes here 1 2 3 4 5 6. And, the next work is to draw the edges between these nodes, we draw an edge from 1 node to another, when we see that execution of the statement representing the node, when transfer control to the other node.

Now, let us look at this case, we have one is the while statement from here it can enter the loop the control can come to the next statement. So, 1 2 2 it can come and also if while becomes false the control might come to 6. So, I drawn here 1 to 6, 2 is a decision statement and from 2 either the control can come to 3 or it can come to 4. So, that is what I have represented? So, that is the basic idea here. That given a program segment or a unit, we first number the statements here and then these become the nodes and the control flow graph representing each statement. And, then we draw an edge between the nodes, if in the program control can flow from 1 node to the other.

(Refer Slide Time: 26:16)

```
graph TD; 1((1)) --> 2((2)); 2 --> 3((3)); 2 --> 4((4)); 3 --> 5((5)); 4 --> 5; 5 --> 1; 2 --> 6((6));
```

So, the same example here that, we have numbered the statements and then we have drawn the corresponding control flow graph, but let us look at a systematic way, how we can develop the control flow graph for any given program?

(Refer Slide Time: 26:50)

The slide is titled "How to Draw Control flow Graph?". It contains the following text:

- Every program is composed of:
  - Sequence
  - Selection
  - Iteration
- If we know how to draw CFG corresponding these basic statements:
  - We can draw CFG for any program.

The slide also features logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES at the bottom, and a small video inset of a speaker in the bottom right corner.

To draw the control flow graph for any given program, we need to understand that every structured program consists of 3 types of statements; sequence, selection, and iteration and if we can understand, how the control transfers? For these 3 types of statements, then on the control flow diagram we can represent those edges. Whenever we find a sequence type of statement in a program and we know how to represent sequence, which is then we can easily draw that edge. For selection if we know how to draw the edges? We will do the same thing on the control flow graph and for the iteration. So, these are the basic 3 constructs.

Let us understand how to draw the edges and the CFG corresponding to these 3 types of statements. And, then given any program we should be able to easily develop the control flow graph, because they will after all be consisting of these types of statements. We are almost at the end of this lecture we will stop here and we will continue in the next lecture from this point.

Thank you.