

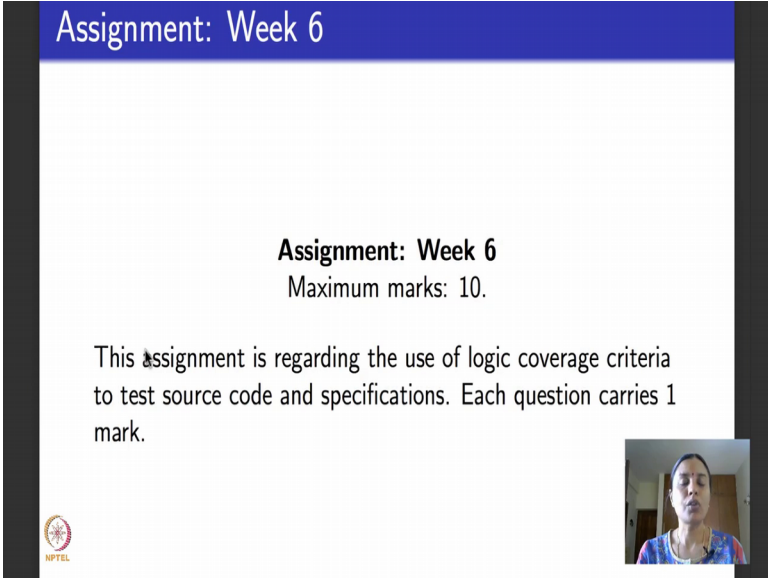
Software Testing
Prof. Meenakshi D'Souza
Department of Computer Science and Engineering
International Institute of Information Technology, Bangalore

Lecture - 30
Assignment Solving

Hello all welcome to week 7. The first thing that we will do this week I will help you to go through the assignment that was uploaded for week 6, and then help you to solve that assignment.

If you remember what that assignment was weeks 6 we finished logic coverage criteria specifically, we saw how logic coverage criteria applies to test source code to test design aspects things like preconditions or post conditions or invariants that come bed design. And we also saw an example of how logical predicate come as guards in finite state machines and so how to design test cases based on that.

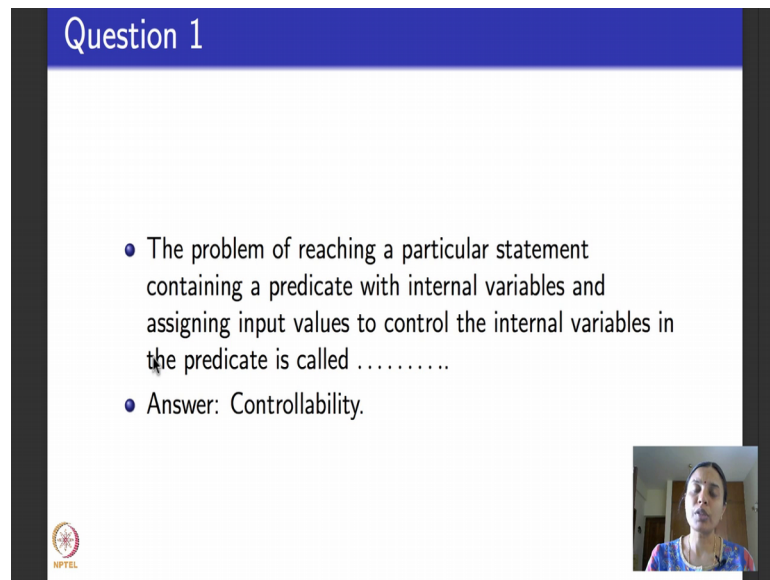
(Refer Slide Time: 00:53)



The slide has a blue header with the text "Assignment: Week 6". The main content area is white and contains the following text: "Assignment: Week 6", "Maximum marks: 10.", and "This assignment is regarding the use of logic coverage criteria to test source code and specifications. Each question carries 1 mark." In the bottom right corner, there is a small video inset showing a woman speaking. In the bottom left corner, there is a small NPTEL logo.

So, given you a simple assignment for 10 marks consisting of fill in the blanks and objective choice questions for that week, this video will walk you through the assignment. And we will discuss how we could solve it. I hope all of you have made an attempt to solve the assignment yourself before viewing this video.

(Refer Slide Time: 01:10)



Question 1

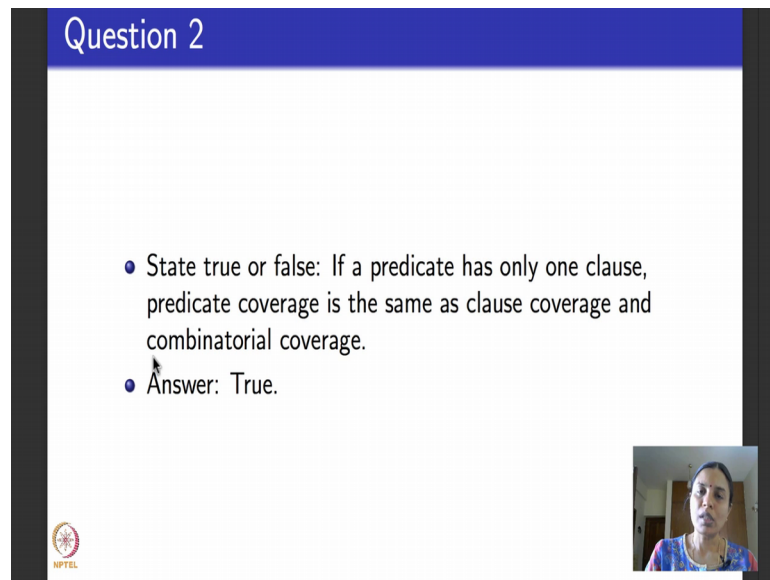
- The problem of reaching a particular statement containing a predicate with internal variables and assigning input values to control the internal variables in the predicate is called
- Answer: Controllability.

NPTEL

So, what was the first question in the assignment? It was a problem of reachability and controllability or R I P R criteria, as we called it in the first week. So, some predicates in the source code purely contain internal variables may not contain any inputs. So, you need to be able to reach those predicates and effect whatever coverage criteria that I want to effect on them by changing values of inputs.

So, this question is about that (Refer Time: 01:39) is it asks you the problem of reaching a particular statement containing a predicate with internal variables, and assigning values to input such that the predicate can be made true or false the particular clause in the predicate can be made true or false, what is this problem called. It is the problem of controllability it comes after the problem reachability, reachability will just reach the predicate controllability will help you to assign values to input variable so as to control the predicate.

(Refer Slide Time: 02:10)



The slide is titled "Question 2" in a blue header. It contains two bullet points: "State true or false: If a predicate has only one clause, predicate coverage is the same as clause coverage and combinatorial coverage." and "Answer: True." In the bottom right corner, there is a small video inset showing a woman speaking. In the bottom left corner, there is a small logo for NPTEL.


The second question was a simple state true or false question, that asks consider a predicate that has only one clause it ask you to say is clause coverage the same as predicate coverage is the same as combinatorial coverage the answer to this question is an obvious yes or true, why because the predicate has only one clause. So, making the predicate true or false will also make the clause true or false, and there are exactly two possible assignments for this single predicate that contains a single clause that is the entire truth table this single predicate is made true once single predicate is made false once.

So, the truth table also has exactly only two values. So, it is the same as all combinations coverage. So, whenever a predicate has a single clause, all these three coverage criteria predicate coverage, clause coverage and all combinations coverage turn out to be the same.

(Refer Slide Time: 03:03)

Question 3

- Which of the following represents the normal form in which predicates corresponding to pre-conditions occur?
 - 1 Prenex normal form
 - 2 Disjunctive normal form
 - 3 Conjunctive normal form
 - 4 None of the above
- Answer: Third option above.



Question number 3 is a multiple choice question, which reads as follows it asks you the following which of the following represents the normal form in which predicates corresponding to pre conditions are. There were four options given to you in the question, the option 1 was prenex normal form, option 2 was a disjunctive normal form, option 3 was conjunctive normal form, an option four says it occurs in a normal form that is none of the above 3.

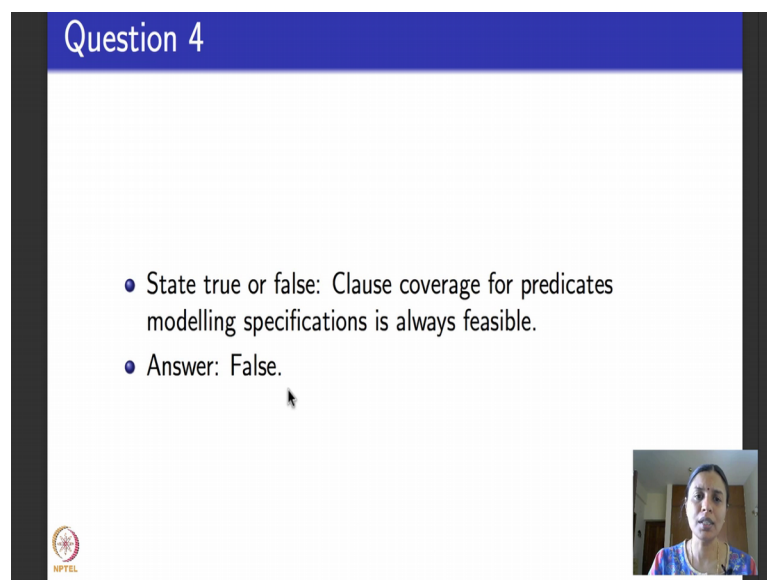
The correct answer to this is the third option which is conjunctive normal form; why is that so. If you remember when we had seen predicate conditions we have seen an example relating to these the probably example program that we had seen was calculate the number of days between two given dates in within the same year and then the program has several pre-conditions which specify what the values for months will be like, what the values for the various dates will be like. And the fact that the value for the first month should be before the value for the second month, so that I can specify the number of days that have passed in between the first date and second date.

Being taken all these pre conditions and in the program that is the pre conditions that were written in English originally, but transformed into predicates. Each condition was written as a simple clause and the entire precondition corresponding to the program was and of all these individual clauses. So, if it is an and of all the individual clauses or if it is an and of what is called disjunct where inside there are odds, than the normal form is

supposed to be conjunctive normal form that is condemned represents and this entire predicate which occurs is a precondition is an and of all the individual conditions because we really cannot write an or because you cannot say that one of the pre conditions should be true it is just each and every single precondition should be true.

So, the natural connective is an and when the connective is an and, the normal form that we are talking about is conjunctive normal form. Disjunctive normal form deals with odds prenex normal form. In fact, does not deal with this kind of thing it deals with formulas which have quantifiers I have just given it as a filter option. So, the correct option here is conjunctive normal form.

(Refer Slide Time: 05:13)



Question 4

- State true or false: Clause coverage for predicates modelling specifications is always feasible.
- Answer: False.

NPTEL

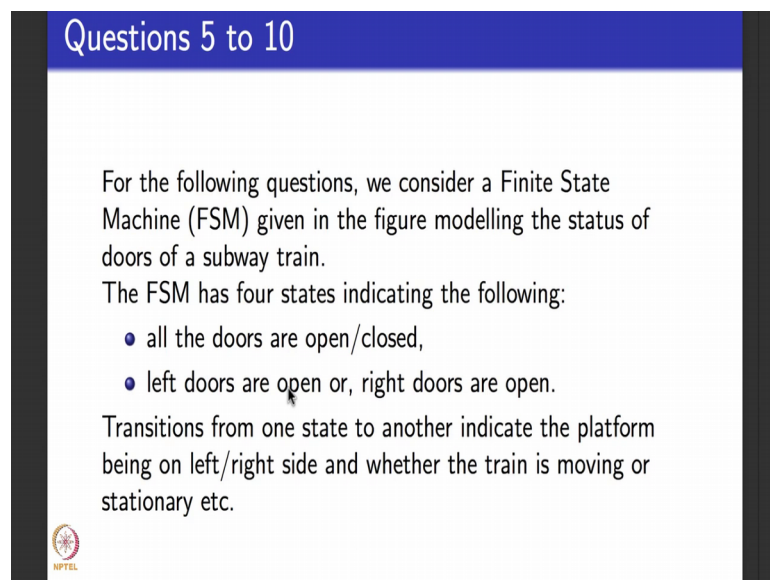
So, the next question is asked you to state at the whether of given statement is true or false the statement that was given is this clause coverage for predicates, that model specifications always visible. In other words when we consider specifications which are requirements given in English or as finite state machines, write out the logical formula corresponding to them and consider clause coverage for these predicates it has is it always possible to write test cases for the t r of gloss coverage or in other words is clause coverage feasible.

The answer is false why is it false we saw one example through the course of lectures last week if you remember we saw the example of a finite state machine corresponding

to a subway train, and there using illustrations I showed you how if one clause becomes true then the other one cannot be true.

So, two classes in a particular predicate cannot be true simultaneously. So, if I have to consider clause coverage I have to be able to make each clause intern true and false, but they occur in such a way that if one clause is true the other will never be true. So, if I try to achieve clause coverage for one clause, I will not be able to achieve clause coverage for the other clause; so clause coverage for predicates that occur as specifications are not always visible.

(Refer Slide Time: 06:38)




Questions 5 to 10

For the following questions, we consider a Finite State Machine (FSM) given in the figure modelling the status of doors of a subway train.

The FSM has four states indicating the following:

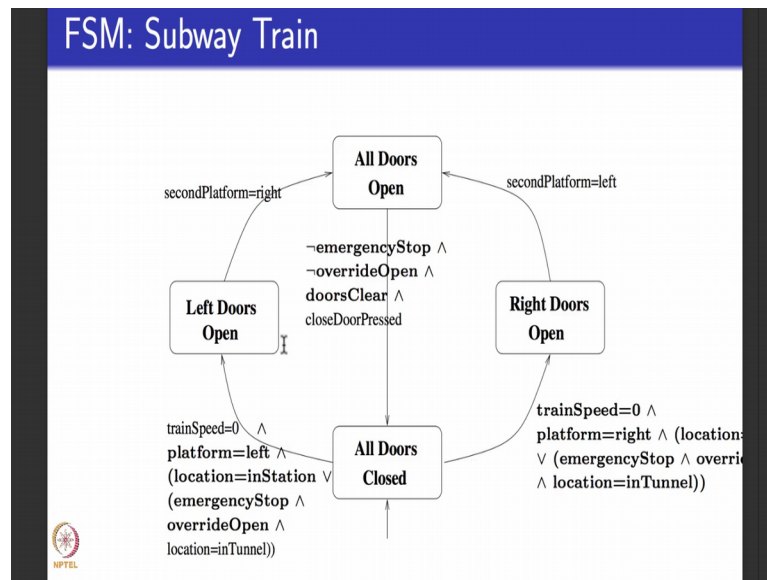
- all the doors are open/closed,
- left doors are open or, right doors are open.

Transitions from one state to another indicate the platform being on left/right side and whether the train is moving or stationary etc.



So, the answer to this question is false. For the remaining 6 questions that is question number 5 to question number 10, I had given the state machine that we saw during the lectures which was the state machine corresponding to a subway train and had asked you to work out a few things related to predicate coverage for that state machine.

(Refer Slide Time: 06:59)



So, the state machine has 4 states just to recap these are the 4 states all doors closed all doors are open, the doors on the left hand side are open and the doors on the right hand side are open. Initially in that train the initial state as marked like this is the state where all doors are closed. And the train starts moving out of the station when it reaches the next station if the platform is on the left hand side then the left doors becomes open, when the platform is on the right hand side the right to become open, in case there is platform that is accessible on both the sides then all the doors become open.

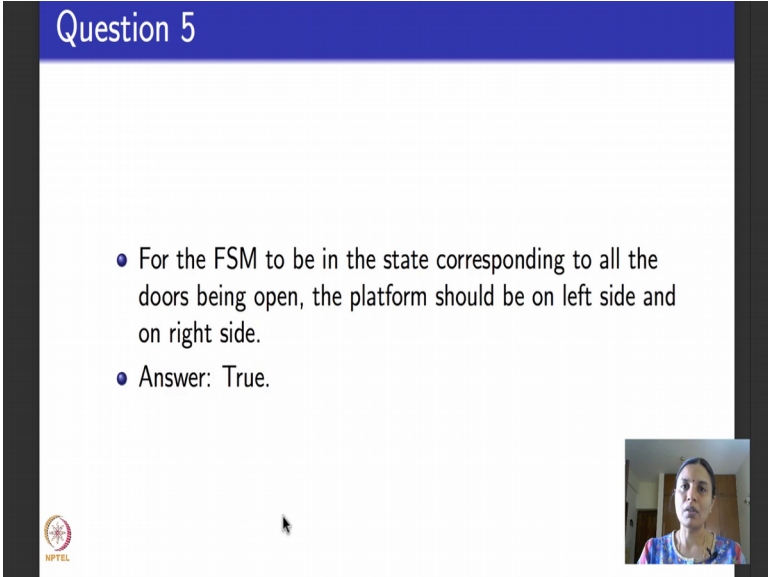
So, what are these predicates that model the transitions between the trains, between the states? So, it is say from the state all doors closed to the states left door open, I can do the transaction provided this fairly big predicate is true. So, let us read the predicate and understand what it says.

So, it says train speed is 0 which means the train is not moving it is come to a station, if train is moving you better not open the doors right and it says the platform is on the left hand side that is what I told you a little while ago, the location is either the state the train is in the station which says location is in station or somebody has pressed an emergency stop button that is why the train is not moving, and has also done an override open of the train, while the train is still in tunnel is. It clear please similarly on the other side I am sorry this thing guard is running over a bit out to the slide, but it is an exact flip of the guard on the left hand side.

So, you see from a state where all doors are closed, I can move to a state where the right door is open provided the train is not moving platform is on the right and the train is in the station or, somebody has pressed an emergency stop done over ride open for the door and the train is still in the tunnel. If any from any of these states the other door is open the platform is available on the other side like if the right told that open and the second platform is available on the left hand side then you can open the left door.

So, you got a state all doors similarly if the left doors are open, and the second platform is available on the right hand side then you can open the right doors also and then you go to the state all doors open. So, from the state all doors open I can go to all doors closed, provided there is no emergency stop nobody has pressed override open, all the doors have been cleared in the sense that there are no passengers moving in and out the doors and somebody has pressed the close door, but. So, is it clear please what the state machine looks like, this is a simple four state machine started with a lot of predicate. So, we will see some questions about the state machine and about predicate logic coverage criteria.

(Refer Slide Time: 09:52)



The slide has a blue header with the text "Question 5". Below the header, there are two bullet points:

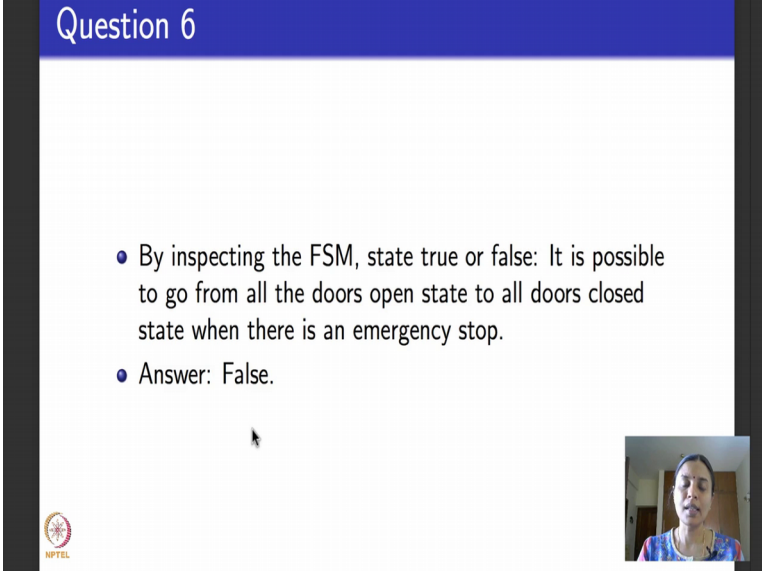
- For the FSM to be in the state corresponding to all the doors being open, the platform should be on left side and on right side.
- Answer: True.

In the bottom right corner of the slide, there is a small video inset showing a woman with dark hair, wearing a blue patterned top, looking towards the camera. In the bottom left corner, there is a small circular logo with the text "NPTEL" below it.

So, the fifth question which was the first in the list of questions about the state machine reads as follows. So, it is for the finite state machine to be in a state corresponding to all the doors being open, the platform should be on the left hand side and on the right hand side. Is your answer true or false the answer is obviously, true because if you inspect this

to finite state machine as we saw, it could be in left doors open or right doors and if the platform is available on the other side then it goes to all doors open. So, the answer to this is true.

(Refer Slide Time: 10:26)



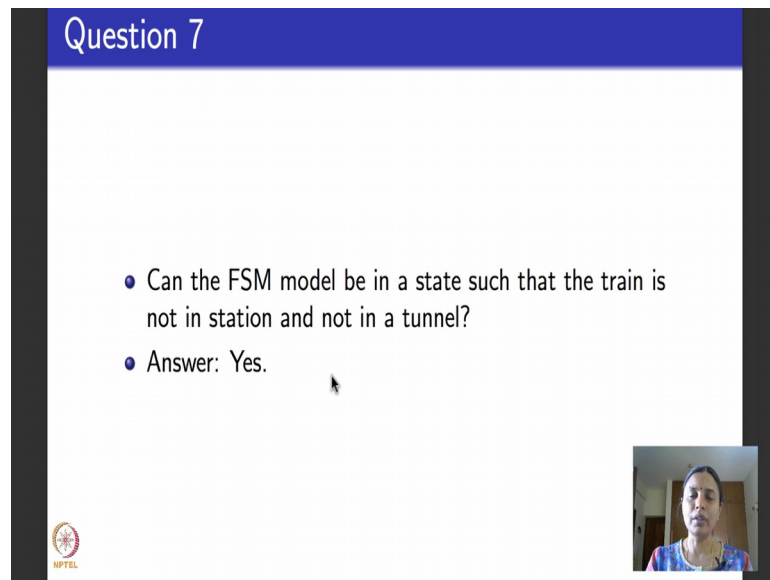
Question 6

- By inspecting the FSM, state true or false: It is possible to go from all the doors open state to all doors closed state when there is an emergency stop.
- Answer: False.

So, the next question says look manually inspect the FSM which means study the FSM and state true or false for the following question. It is possible to go from all doors open state to all doors closed state when there is an emergency stop. So, let us go back and inspect the turing machine, the state that we are considering the transition from out off is the state all doors open, the state that we are considering to translation into is the state all doors closed; and the questions says I can go from this state to the state when there is an emergency stop.

So, let us look at the guard in that transition there is an emergency stop clause in the guard in the transition, but what is the clause come with? It comes with a negation if you look at it here the first clause is the clause on emergency stop it is a negation of emergency stop not emergency stop. So, the answer to this question is false.

(Refer Slide Time: 11:23)



Question 7

- Can the FSM model be in a state such that the train is not in station and not in a tunnel?
- Answer: Yes.

NPTEL


So, moving on the next question question number 7 in the assignment was the following, can the finite state machine model be in a state such that the train is not in station and not in a tunnel. Is that possible if you remember we had looked at it even when we did the lectures it is possible. So, when you try to do the truth table for each of the clauses you will realize that there is a particular state, where the not in tunnel in tunnel need not be true and in station need not be true.

Whereas, in real life that is not possible and in fact when we did the lectures we had understood that there is something wrong with finite state machine. So, from the point view of testing it should be flagged as a potential error or vulnerability back to the designer.

(Refer Slide Time: 12:10)

Question 8

- Consider the guard:
 $\text{trainSpeed}=0 \wedge \text{platform}=\text{left} \wedge (\text{location} = \text{inStation} \vee (\text{emergencyStop} \wedge \text{overrideOpen} \wedge \text{location} = \text{inTunnel}))$.
This predicate has six clauses. Answer the following questions with respect to this predicate.
- Determine the conditions under which the clause $\text{trainSpeed}=0$ determines the guard.
- Answer: $\text{platform}=\text{left} \wedge (\text{location} = \text{inStation} \vee (\text{emergencyStop} \wedge \text{overrideOpen} \wedge \text{location} = \text{inTunnel}))$



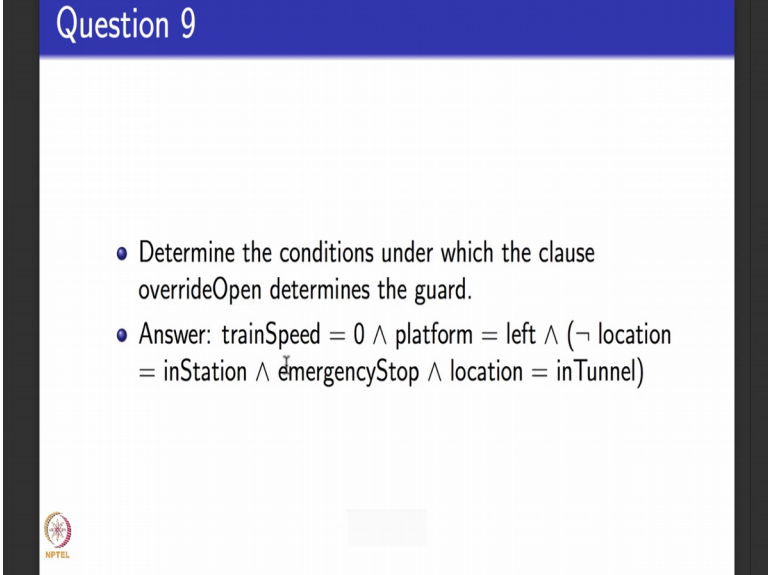
So, the next question was about guard clauses in a predicate determining the predicate. So, the we were asked to be started with one guard which is the guard that takes you from all doors closed to left doors open, which is this guard it is says train speed 0 platform is on the left hand side train is in station or there is an emergency stop an override button pressed and the train is still in tunnel. If you see there are 6 guards here 6 clauses sorry here 1, 2, 3, 4, 5, 6 and what it asks you to do it takes we have taken examples of 3 clauses and ask you to determine the condition about when the clause will determine the predicate.

So, in this question we have taken the first clause which is the clause of train speed being equal to 0, and you are asked to determine when this clause will determine the predicate. I have not given you the working out here, but the kind of thing that you have to do is substitute for the this clause to be true in this whole predicate, substitute for this clause to be false in the whole predicate and absorb it. Then use the logical operators that we know inference rule that we know to be able to simplify it. So, when I substitute it with true it will be true and something else.

So, the true will go away this one will get retained, when I substitute it false it will be false and something else. So, the false will stay back and then you go on simplifying this will be the final answer. It will be platform is equal to left and location in station or emergency stop and override open and location in tunnel; this is what will happen this is

the predicate that will get if you try to make the first clause determine the guard. So, you stop at this you do not have to write test requirements and test cases for c a c c and other active clause criteria, I just ask you to tell the conditions or the predicate under which this clause will determine the predicate that is all is the question about.

(Refer Slide Time: 14:08)



Question 9

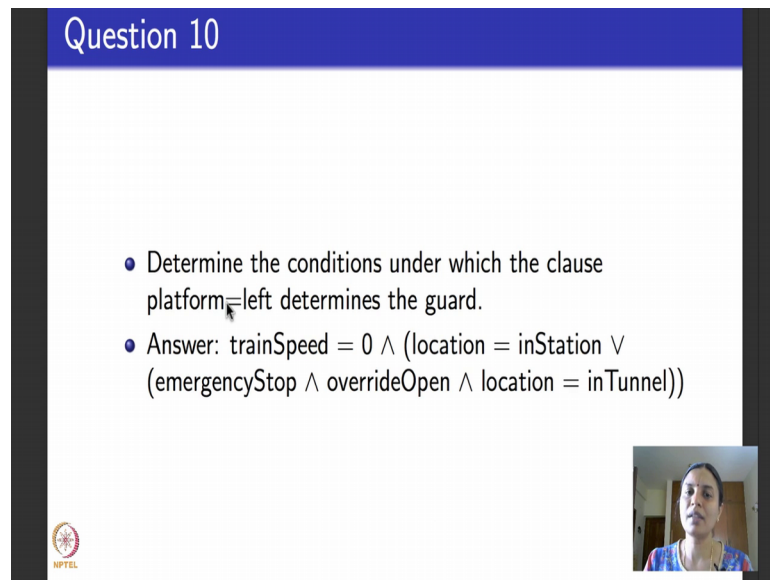
- Determine the conditions under which the clause `overrideOpen` determines the guard.
- Answer: $\text{trainSpeed} = 0 \wedge \text{platform} = \text{left} \wedge (\neg \text{location} = \text{inStation} \wedge \text{emergencyStop} \wedge \text{location} = \text{inTunnel})$

NPTEL

Next question is very similar, instead of taking it takes the same predicate and it takes this clause `override open` which comes inside this hand and ask you to determine the conditions under which this clause will determine the predicate. So, you have to do the same thing substitute `override open` with true ones, substitute `override open` with false ones and then simplify the resulting logical formulae using the inference rules of logic and write the final predicate which will tell you exactly when this clause will determine the predicate.

Here again you do not write the test cases because there is no `t r` given in terms of do this to achieve certain active clause coverage criteria or inactive clause coverage criteria. So, you just stop at writing the resulting predicate and leave it.


(Refer Slide Time: 14:55)



Question 10

- Determine the conditions under which the clause $platform = left$ determines the guard.
- Answer: $trainSpeed = 0 \wedge (location = inStation \vee (emergencyStop \wedge overrideOpen \wedge location = inTunnel))$

NPTEL



Question number 10 is again a same thing it picks up another clause this time the clause that is picked up is the clause which says platform is on the left, and it asks you to determine the conditions or the predicate under which the clause determines the guard (Refer Time: 15:10) do the same thing substitute this to be true ones, substitute this to be false one XOR, then use the inference rules of logic to be able to simplify it this is the result in answer that you will get. Try and do it on your own if you get stuck feel free to query me in the forum and I will try to work it for you through a reply.

I hope this small assignment solving exercise was useful for you and your answers did match a lot with the answers that we have worked out. What you will do today for rest of this week is I will start with a new chapter which is called input space partitioning. So, my next lecture we will not do logic coverage criteria anymore. We will begin with the new set of test case algorithms based on input space partitioning.

Thank you.