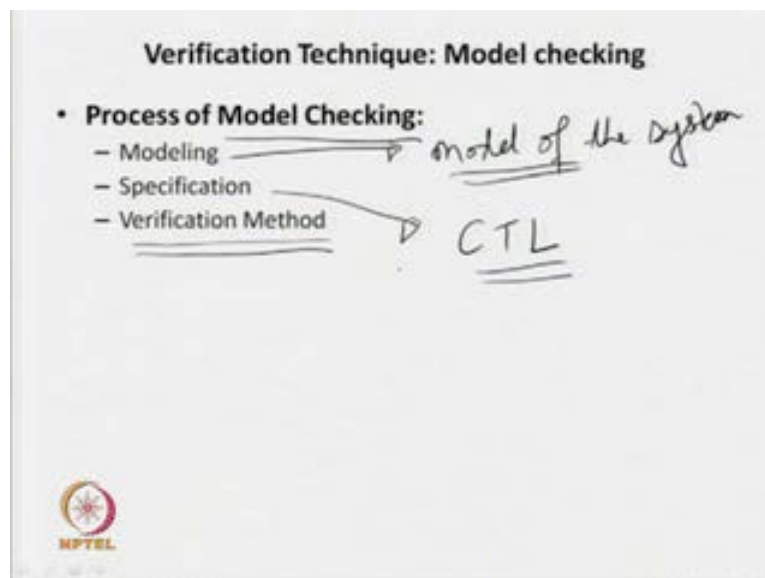


Design Verification and Test of Digital VLSI Designs
Prof. Dr. Santosh Biswas
Prof. Dr. Jatindra Kumar Deka
Indian Institute of Technology, Guwahati

Module - 5
Verification Techniques
Lecture - 1
Introduction to Model Checking

So in our last module we have introduced one logic called temporal logic and which will be used to specify the property of our system. Now in this module we are going to look for verification technique. And mainly we are going to look for particular verification technique which is known as your model checking.

(Refer Slide Time: 00:51)



So in this class I will just briefly introduce what is model checking and what we will we can do with the help of model checking. So I think you might be remembering that in our introductory class, we have slightly introduced about that process of property verification and we have slightly introduced about the model checking. Now if you recall back we will find that in the process of model checking we are having basically three component.

First one is your modeling. So in case of modeling what you have to do? You have to give a model of the system, somehow model of the system. With some formalism we have to define or we have to give the model about system, if we are going to design a new system then we are coming up with design and we are going to represent that design

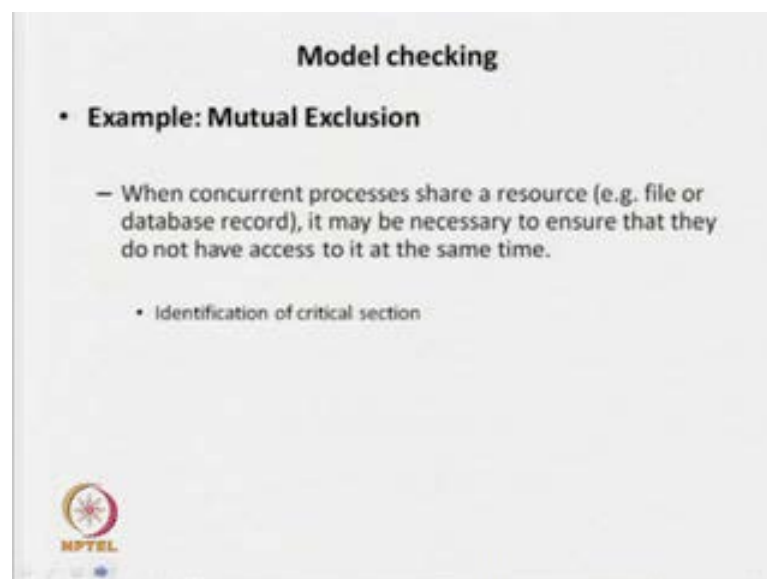
with the help of some model. Will see how we are going to do it.

Second component is specification, or we can say the property to be satisfied by this particular system, or you can say that this is the property that will be satisfied by the model. Because we are going to abstract the model of our system.

So in last class we have introduced temporal logic, temporal logic can be treated as a specification language and we can use temporal logic to specify all property. And we have discussed the special class of temporal logic which is known as your CTL-computational tree logic. So basically will concentrate on this particular logic CTL and you see how property can be specified over here. So one we have the model of the system that we have abstracted the model, we represent our property which we allow you can give the specification, then we need some methods or some mechanism by which we can check that this properties of indeed to in the models. So this is the third component which is your verification method.


So in this particular course we are basically going to look about the model checking technique, which is a verification technique and which is basically property verification technique. So here in this module we are going to talk about this particular verification method called model checking.

(Refer Slide Time: 02:56)



Model checking

- **Example: Mutual Exclusion**
 - When concurrent processes share a resource (e.g. file or database record), it may be necessary to ensure that they do not have access to it at the same time.
 - Identification of critical section



So this model checking we are going to introduce with the help of one example, first will

see an example and then will see how this example will be model or how we are going to give the system of the model, then we will see what are the properties that need to be satisfied for this particular example and how we are going to write those particular property or specification in CTL. Once we have the model, we once we have the specifications in your CTL computational tree logic, then we see that model checking approach or how this can be verified further particular model. So the example, that here we are going to discuss about mutual exclusion.

I think all of you know about this particular problem that it is basically when we are working with the shared resources. So one concurrent process shared the resource. You can have several resources it may be a shared memory, we may have a file, the file may be shared by the foreign users, on the other hand we may look into the databases also, where the database will be used by different persons or different activities.

So in record also we have to see that record of database will be excess from different locations or in all those cases where we are having the concurrent execution. Always we have to look for the consistency of our data if it is a file or if it is a database record or may be the shared memory. Whatever data we have over here it should be consistent. All users should get the similar view of this data, it is not like that I have updated something and that is not reflected for you, then you are not going to get the correct information. So, this is the problem that we have when we are going to use shared resources.

So, to have a proper consistency of data we can use this particular mutual exclusion principle. So, here in this particular case we have to see what are the criteria that we need to look for it when we are going to model such type of system. So in this particular case first we have to find out what are the criteria that has to be satisfied by this particular design. So we are talking about that we have to look for those criteria that what are the basically we have to look for that particular portion of fort or that particular concurrent processes, where they are using this particular shared resources. It may be your file.

(Refer Slide Time: 05:03)

Model checking

- **Example: Mutual Exclusion**
 - When concurrent processes share a resource (e.g. file or database record), it may be necessary to ensure that they do not have access to it at the same time.
 - Identification of critical section
 - How to model the system
 - What are the specifications

Diagram: A large curly bracket on the right groups the terms 'process 1', 'process 2', 'process 3', and 'process n'. An arrow labeled 'file' points from this group towards the left, indicating shared access.

MPTEL logo is visible in the bottom left corner of the slide.

So, you can have a file so, it may be used by several processes, sources of the user you can say that process 1, process 2. Like that and different process you are having and this n processes are going to use this particular file. So, all those particular process are in own course segment, so, in a particular course segment I think you can think that it is going to access this details of particular file. So we have to identify by that particular course sets.

So two such processes should not access the files simultaneously, that is the restriction we have to say and we say that we have to identify those particular critical section. We say that the file used by this particular course segment will be detected as critical section so all processes may have critical sections. So, they should not use this particular file at the same time.

Now by looking into this criteria, we are going to talk about the critical section, when we talk about the mutual exclusion. Then we have to say how to model this particular system. Now we are going to talk about the concurrent processes, going to use some shared resources. So, further we have to come up with the model and what will be the specification of what are the specifications, that need to be satisfied by this particular model. So once we are having the specification, we have to come up with the specification, we have to look for properties, will come up with the system model and what we are going to see? How this we are going to check? Whether the specification indeed holds in this particular system model or no.

(Refer Slide Time: 07:15)


Mutual Exclusion Example

- Two process mutual exclusion for shared resources
- Each process has three states
 - Non-critical (N)
 - Trying (T)
 - Critical (C)
- Initially both processes are in the Non-critical state $\rightarrow N_1 N_2$

P_1 P_2

| | | | | | | |
|-------|---------------|-------|--|-------|---------------|-------|
| N_1 | \rightarrow | T_1 | | N_2 | \rightarrow | T_2 |
| T_1 | \rightarrow | C_1 | | T_2 | \rightarrow | C_2 |
| C_1 | \rightarrow | N_1 | | C_2 | \rightarrow | N_2 |

N:
T:
C:



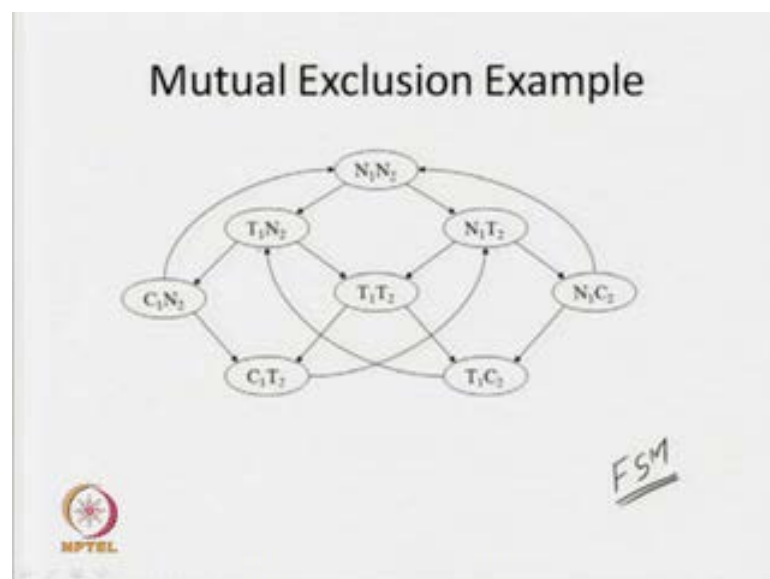
So our first step is to come up with a model. Now you just see that now what is this particular mutual exclusion, we are considered simple cases, where which we are having two processes, mutual exclusion or shared resources. We are considering only two processes, these two processes are going to access some shared resources. It may be your shared memory, it may be a file or it may be a database record. So for that particular cases, what happens if process can be divided into three different state. We can say that it is your non-critical N trying T and critical C. So non-critical and basically it says that the process is in your non-critical region; that means, it is not using any shared resources. So all process can walk in the non-critical region simultaneously.

The state T we are talking about this trying, basically say that no one particular process is trying to enter into the critical region. Basically it gives the request that it is trying to walk with this particular shared resources. And our requirement is like that only one process can use this particular shared resources. One should get access to the shared resources, then we say that the process is in your critical section; that means, now it is accessing this particular shared resources. So this is critical C we are seeing. So we can say that one particular process can be in any one of this particular three states, N, T or C. N is non-critical region, T is it is trying to enter into the critical section; that means, it is requesting what is particular shared resources and critical means, now the process is executed in the critical region. That means, it is working with this particular shared resources.

Now we are talking about two process mutual exclusion. So we are having two processes. Now say that I am having processes P 1 and I am having process P 2. Now process P 1 can have in this particular state that it was N 1, T 1 and C 1, similarly, process two can be state it is either it is N 2, T 2 and C 2. Now what are the transition users see that, if it is in the N 1; that means, it is non-critical section. Now at some point of time it may go to the trying state; that means, now it is trying to enter into the critical section, once it is entering into the trying state; that means, it is going to access the shared resources now it can the shared resources then what happens? It will go into the critical section; that means, it is going to use this particular shared resources especially it is going to use this.

Once the job is over then what will happen now, it will come out from critical section to non-critical section; that means, it will going to do the other normal execution. So this is the transition of our process P 1 similarly, for process P 2 also we have this particular similar transition. So, N 2 to T 2 and T 2 to C 2 to N 2. So these are the possible transition that we have one process P 1 and process P 2. Now we are going to see these things as an one system and further what we are going to do? The behaviour of these two process P 1 and P 2 will be composed together. Generally, we say this is the parallel composition. Now one we are going to get this composition of this process P 1 and P 2 then we are going to get the total behaviour of our system. Now how now we will see what this total behaviour will look like?

(Refer Slide Time: 10:41)



Now eventually it can come up with such type of finite step machine, why I am talking about out, already say that this is your some sort of your finite step machine, you are having finite number of states and we are having transition among them. Now what are the states you just see that first I am talking about this is even as your N 1, N 2 what does it means, it says that process P 1 is your non-critical section, process N 2 is your non-critical section. So both are executing doing their own job but, none of them are using the critical section. Now when the state is N 1 and N 2 then what will happen? At some point of time process P 1 may look for shared resources.

So it will go from N 1 to T 1. So it is having a transition from N 1 to T 1 so we are having next other P 1 and N 2 it is process P 2 is a non-critical section, process P 1 is in your trying section, at the same time it may happen that now process P 2 may look further particular shared resources. So we may have another transition so it is from N 1 and N 2 to N 1 to T. So now you just see that initially both are in the non-critical regions now either P 1 try to enter into the critical section.

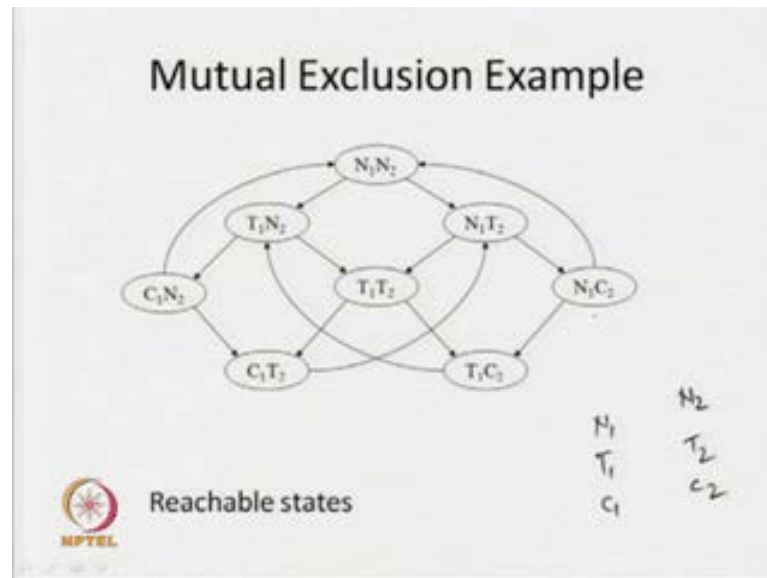
So it will then the states you are get is your N 1, T 1 and N 2, N 1 process P 2 once enter into the critical section then we are going to get the state N 1 T 2. Now when it is in your T 1 N 2 you just see that either now process P 1 is going to get the shared resources. So P 1 will enter into the critical section and N 2 remain in the process P 2 remains in the N 2 state it is in the non-critical section but, it may also happen that now once is trying that process P 1 is trying similarly, process P 2 may also try to trying to enter into the critical section so the state you are going to have is your T 1 and T 2.

Now after some point of time that process your this things what you call P 1 will come out form the critical section. So we are going to have this particular N 1 and N 2 state and now when both similarly, this part is symmetric to the other part that is from T 2 it is going to C 2 then it will come out then it will go from N 1 and N 2. Now when both are entering into trying to enter into critical section so, one will go to process P 1 will go into critical section so it will follow this particular part and process P 2 will go into the critical section and then it will follow this particular part and formed that it will come to particular P 1 state.

So you just see that with this particular state transition machine we have basically given the behaviour about system so, what we have to said that mutual exclusion we say that

we are having two process P 1 and P 2 these are the possible states and these are the possible transition. Now when we are looking into them as a whole then we are just getting this particular transition behaviour.

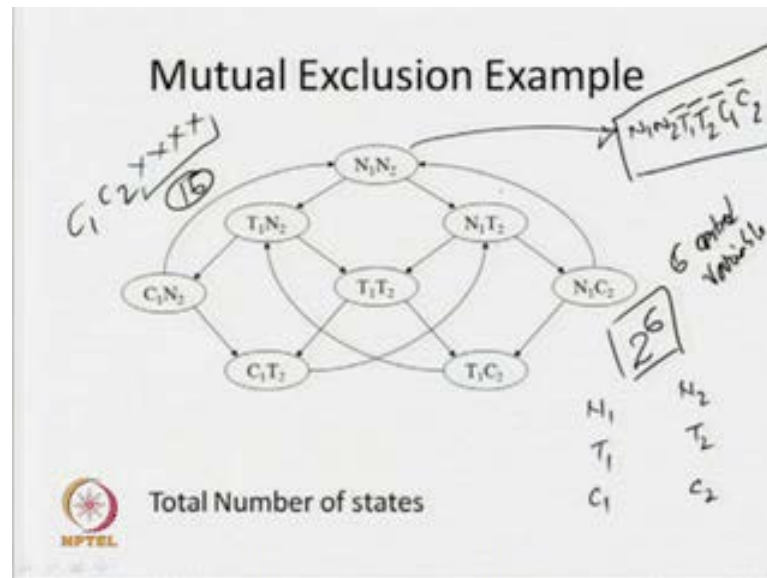
(Refer Slide Time: 13:55)



Now you just see that we are getting a transition behaviour and this transition system is going to depicts the behaviour about system. Now here we are talking about say some state variables we are saying that we can have process P 1, we are having N 1 T 1 and C 1, for process P 2 we are having N 2 T 2 and C 2, out of that if you look into this particular behaviour will say that we are getting 1, 2, 3, 4, 5, 6, 7, 8 total eight different states and we are having some transition.

So these are the states that we are going to talk about these are the reachable states of my system. So if my system is working going to work correctly or work perfectly these are the different possibilities that we are having, so these are the different states that my system can go and we are going to talk about that these are the reachable states.

(Refer Slide Time: 14:53)



If you are talking about the reachable states then we may have some other states which may be non-reachable state. So for this particular example you can think that what are the total number of states that we may have? Now you just see that already we have said that we had been $N_1 T_1$ and C_1 of process P_1 and $N_2 T_2$ and C_2 . Now when I talk that in this particular state I am talking about that this is your $N_1 N_2$; that means, both are in your non-critical section.

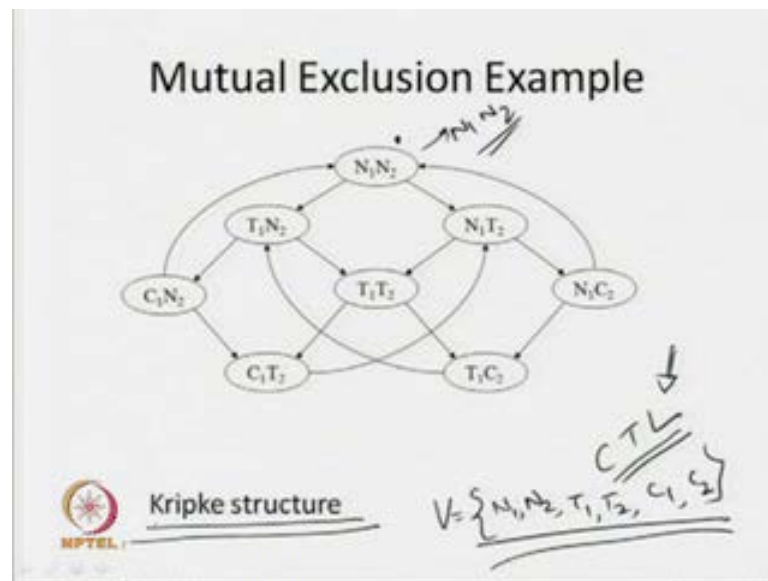
So these are some states variables we can say that these are two basically N_1 is true and N_2 is true and what are the status of other four variables? We can say that this is not trying into enter into critical section. So it is \bar{T}_1 that process T_2 is not trying to enter into a critical section. So it is \bar{P}_2 similarly, process P_1 not in the critical section so, it is \bar{C}_1 and similar process P_2 is also not in the critical section. So it is your \bar{C}_2 bar. So you just see that with this particular combination basically we are representing this particular states. Now in this particular case so it is basically N_1 and N_2 are true; that means, both the process are in your non-critical section. So similarly, \bar{P}_1 is false and \bar{P}_2 is false they are not trying to enter into critical section and \bar{C}_1 and \bar{C}_2 also false they are not in the critical section.

So like that for all those particular states we can have the behaviour of this other variables. Now in this particular case we are having total six control variables. So if we are having six control variables what will be the different possible combinations of this

particular control variables? We will see that will get altogether 2 to the power 6 different states because if you are having n control variables, then we are going to get 2 to the power 6 different states. So; that means, my total state is your 2 to the power 6 which is at a 64; that means, I am going to get 64 different states, this is the total number of states of this particular system but, out of that what will happen? We have seen that only 8 are reachable, others are not reachable because, for a proper system both cannot be in the critical region; that means, C 1 and C 2 may not should not be true simultaneously.

So if this is your C 1 and C 2 then we are having other four variables T 1 T 2 and N 1 N 2. So see this particular four variables now have either true or false so; that means, we can get four variables, we are going to get 16 different states; that means, this 16 different states are not reachable this is not the proper behaviour of our system and we are going to say that these are the not reachable states, like that some other states will also go out from this particular model and eventually, we are getting only this 8 possible states. So these are the reachable states. So you just see that trying to model our system depending on the input variables or the control variables you may have bigger states to express but, all states may not be relevant for us and we are going to basically concentrate on the reachable state. So these are the reachable states.

(Refer Slide Time: 18:18)



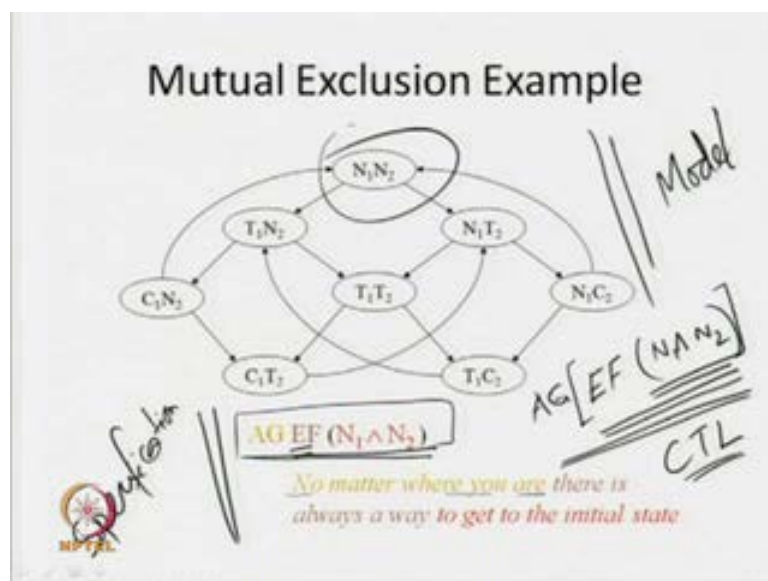
You just see that when we discussed about your temporal logic or CTL computational

tree logic, we have mentioned that the meaning of this particular CTL is define over a model. Now what is the model? If you recall back you will find that this is nothing but, some sort of financial system we are having finite number of states, we have the transition. So this is the special behaviour of our financial system but, along with that we are having one additional function which is known as your labelling function, it states will be label with the atomic proposition if the atomic proposition is two in a particular step we are going to label this particular step with the help of this particular atomic propositions. Now when we are coming up this particular model for mutual exclusion here will find that our set of atomic propositions V will be your $N_1, N_2, T_1, T_2, C_1, C_2$.

So this is the set of atomic proposition that we have in this particular model. And you just see that in this particular state we have marking with N_1 and N_2 . So what does it mean? That means, this particular atomic propositions N_1 is true over here that the atomic propositions, N_2 is true over here and other part of atomic propositions T_1, T_2, C_1, C_2 are false at that particular states.

Now you just see that when I am coming up this particular model, this particular model now very much similar to our kripke structure that is used for our in CTL to define a meaning of our CTL computational tree logic formulas.

(Refer Slide Time: 20:54)



So now eventually we have come with this particular kripke structure so for model

checking we need a model, we need a specification, specifications are nothing but, the properties that they are satisfied by this particular system and we need some techniques by which we are going to check whether this particular properties are true in our model or not.

So now the first part we need some model we have come up with this model of this particular examples of mutual exclusion and eventually we have found that the way we are representing it, it is going to give us the kripke structure that is leaded to define a meaning of my CTL formula. So in last module we have talked about this particular CTL what is the syntax of CTL and what are the symmetric of CTL? Now we have come up with the model.

Now you just see now when you are coming to this particular model so we have come up with this particular model and this is very much similar to our kripke structure. Now I think now we are writing over here is see that $AG\ EF\ N\ 1\ and\ N\ 2$ $AG\ EF\ N\ 1\ and\ N\ 2$. Now if you look into this particular expression I think you can recall that this is very much similar to our CTL formula and in that this is the CTL formula, because you just see that $N\ 1$ and $N\ 2$ these are two atomic proposition and they are connected by this particular conjunction.

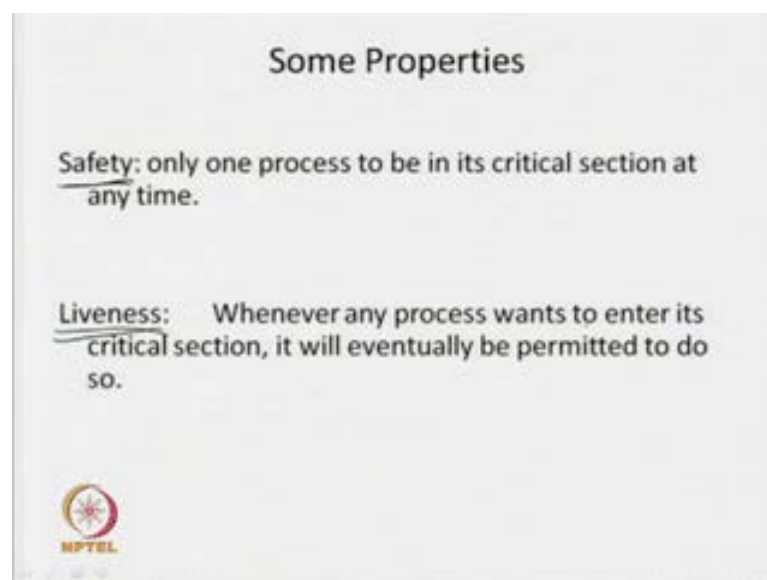
So $N\ 1$ and $N\ 2$ is a CTL formula. Now this is EF what is EF? There exist a part in future so, there exists a part in future $N\ 1$ and $N\ 2$. So I am going to get this is also CTL formula since this is CTL formula $EF\ N\ 1$ and $N\ 2$ will be CTL formula, and before that I am writing AC; that means, in all part globally there exist a part in future $N\ 1$ and $N\ 2$; that means, I can give this particular packet so you see that since this is the CTL formula so eventually this is the CTL formula. So now I am writing of CTL formula, now what this CTL formula means you just see that what we can in sentence no matter where you are there is always a way to get to the initial step.

So no matter where you are there; that means, in the system wherever you are it hardly makes any difference no matters there is always a way, we are going to get a part to get the initial state, here we have to just think that this is my initial state. So this is what the initial state I am having $N\ 1$ and $N\ 2$ must be true. So no matter where you are there is always a way to get to the initial state. Now this is the property that need to be satisfied this particular model. Now you just see that I am having model so this is the model of my

system mutual exclusion, this is the specification or the property, one simple specification that I have mentioning over here so this is not property. Now I need some mechanism to check whether this property or this specification is true in this model or not. And this particular mechanism or technique is known as your model checking, this model checking is a kind of verification technique, it is a kind of verification technique and in this class we are going to or in this course we are going to talk about this particular model checking.

So for model checking what is our requirement? We need a model of the system somehow we have come up with the model which is very much similar to the kripke structure, we have to represent our property or specification in CTL then we are going to use model check up which is basically known as your CTL in this particular case since we are I think the specification in your CTL formula, so we are going to look for CTL model checker and CTL model checker we are going to check whether this particular specification is true in this particular model or no. So this is the way that we are going to look in verification technique and in this course we are going to talk about model checking and particularly model checking of our CTL formula because we are going to write our formula in CTL.

(Refer Slide Time: 24:40)

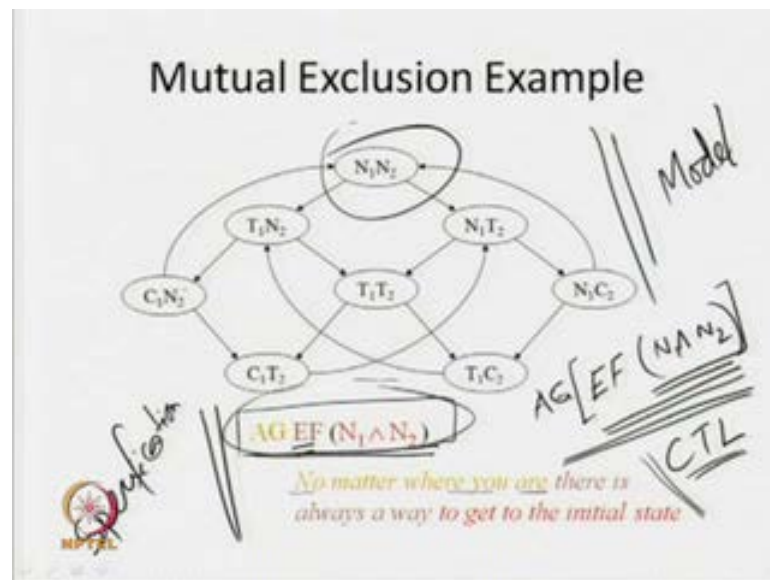


Now when we are talking about this particular mutual exclusion problem of shared resources. Now it satisfies some of our properties we have been some requirements

should fulfil those particular requirements. First we have to identify what are those particular requirement. So one requirement we are talking about is safety requirement. What is this safety requirement? It says that only one process to be in its critical section at any time, this is your basic requirement I am saying that this is the safety requirement. So at any point of time only one process will be in its critical section. If two process are going to remain in the critical section at the same times both of them are going to manipulate the data and manipulation of one process may not be reflected in the other because it will be over written by the other processes. So this is basically safety property only one process to be in its critical section at any time.

Now second property talking about the liveness. What we are talking about the liveness? Whenever any process wants to enter its critical section it will eventually permitted to do so. So we are talking about the liveness; that means, all process are having equal right to enter into the critical section. So whenever a process wants to enter into a critical section; that means, it wants to use the shared resources eventually it should get the permission to use that particular shared resources so this is about liveness.

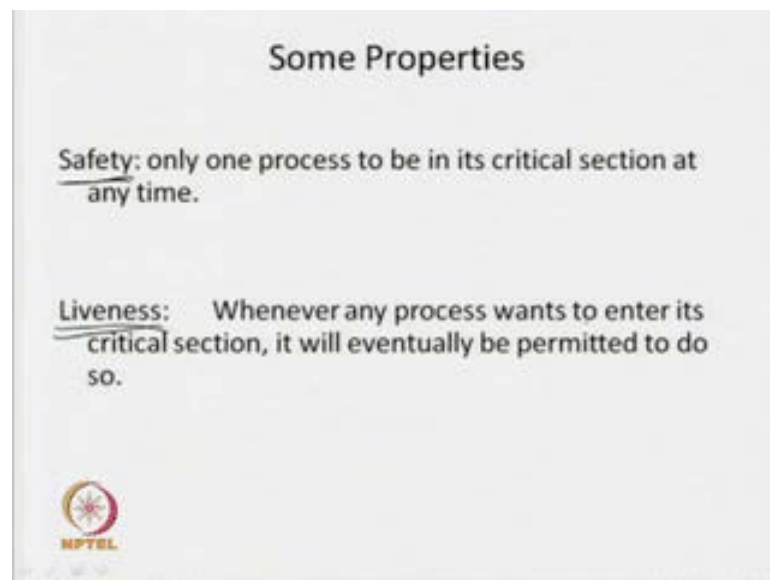
(Refer Slide Time: 26:45)



If someone is waiting and waiting for long enough of time then you can say that process is going to start; so it is not desirable. So the second property is liveness now say, we know that these are the two requirements that we are having. Now whether if I am having this if I say that these two properties are satisfied this particular model or not I

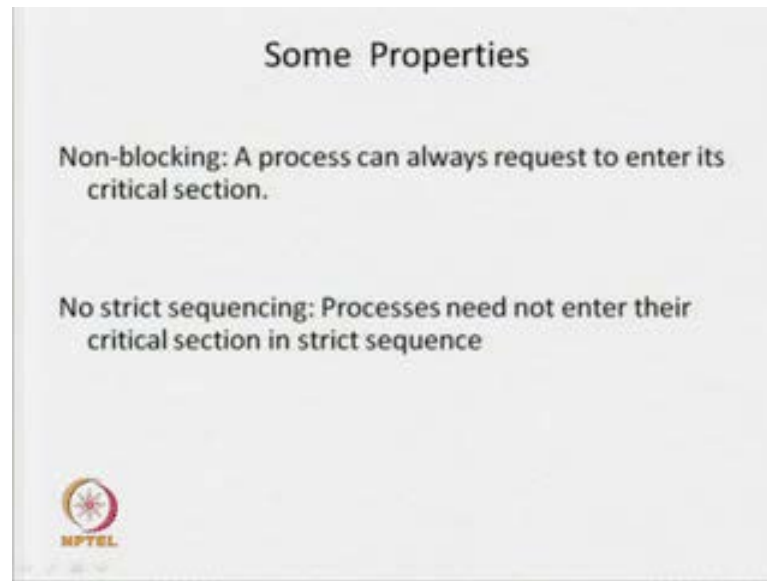
think then you will go or remove with your domain when we have to look for the lesser language processing because we are representing our requirement in lesser language. So we have to use some formalism or use some formal methods to represent such type of properties say like that in my previous example I am saying that no matter where you are there is always a way to get to the initial state. So this is some sort of requirement I am representing this requirement with the help of this particular CTL formula. So some because this CTL formula have been free defined meaning and we has to use some free define syntax to write such type of formula since meaning is obvious so now we are going to look for the foot of this particular formula in this particular model.

(Refer Slide Time: 27:16)



So like that these are the two basic property that must satisfied by mutual exclusion protocol. So somehow we have to represent these things you know in some formal way and since we know about CTL computational tree logic will see how these two will be represented in CTL computational tree logic.

(Refer Slide Time: 27:43)



Now another properties that we are having that non-blocking: A process can always request to enter its critical section. So this is the non-blocking property we should not have some criteria for sometimes we are going to block one particular process, that process will not be allow to enter into critical section, then we say that this is some blocking measure. So we say that on mutual exclusion protocol must satisfy this particular non-blocking property also, A process can always request to enter its critical section at any point of time at any moment, any process can give a request to enter its critical section you should not quote any blocking criteria.

And another criteria or another property is that must satisfied by the mutual exclusion protocol is no strict sequencing, which should not give any sequence any strict sequence to that particular process it is the sequence will we can say that process P 1 will go into the critical section then I will allow the process P 2 then I will allow process P 3 and again I will allow process P 1 to enter into the critical section, then P 2 like that we should not put any restriction in this particular case without any restriction without any sequence will be allowed to enter into the critical section it may happen that we have when P 1 is entered into the critical section after sometimes again P 1 may allow to enter into the critical section, we will not wait for that first P 2 must go then only again P 1 will be allowed. So such type of restrictions we are not going to put this is talk about no strict sequencing.

So basically we said that our mutual exclusion your protocol must satisfy this particular four properties. Now already I have mentioned that now to be understand it properly how many you have to specify this thing and here in this particular course, we are going to look for the CTL representation of those particular properties because we are going to look for CTL model checking.

(Refer Slide Time: 29:25)

Some CTL Properties

Safety: only one process to be in its critical section at any time.

$$\text{AG } \neg(c_1 \wedge c_2)$$

Liveness: Whenever any process wants to enter its critical section, it will eventually be permitted to do so.

$$\text{AG}(t_1 \rightarrow \text{AF}c_1) \wedge \text{AG}(t_2 \rightarrow \text{AF}c_2)$$

$\text{AG } \neg(c_1 \wedge c_2)$

$\text{AG}(t_1 \rightarrow \text{AF}c_1)$

$\text{AG}(t_2 \rightarrow \text{AF}c_2)$

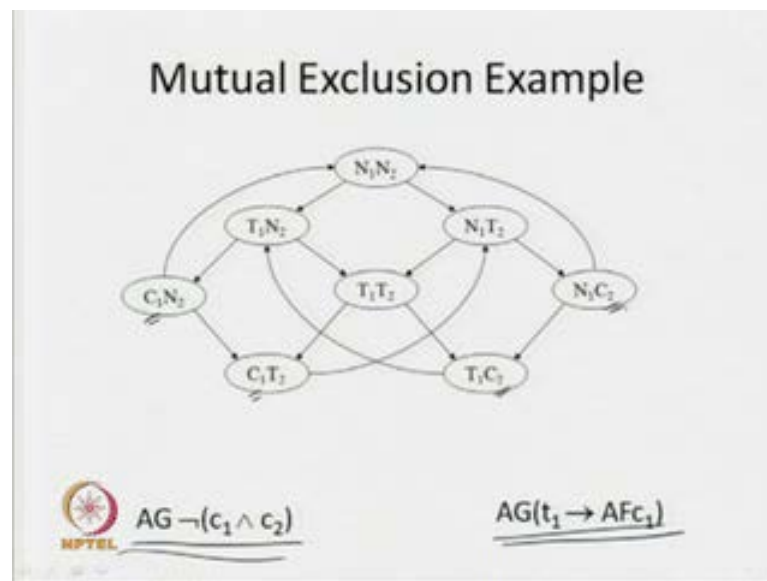
Now the first property that safety property you can say that only one process is to be allowed in its critical section. So we can write this particular CTL formula like that in all part globally not of C 1 and C 2. So if C 1 and C 2 if it is true, then what will happen? Both the process in the critical section simultaneously, so not of this much be true and where it much be true? In all part globally; that means, in the entire system in all states in along all parts this much be true not of C 1 and C 2 that side I think this property is in all part globally not of C 1 and C 2. Now you just see that this is a CTL formula because C 1 and C 2 is a CTL formula because C 1, C 2 is our atomic propositions so negation of CTL formula is also a CTL formula and AG is a temporal CTL globally in all parts so this is a CTL property. So we are I think this is safety property in CTL formula.

Similarly, liveness we are saying that whenever any process wants to enter into critical section it must be allow to enter into the critical section. So we can say that T 1 basically, it says that the process P 1 is trying to enter into the critical section, then if it is true then what will happen? In all part if you saw C 1. So if process P 1 is trying to enter into the

critical section then found that particular point wherever you go in all part in future that process must go into the critical section. So that is why I am saying that once to enter its critical section, it will eventually be permitted to enter so to do so, that is why I am saying that if P 1 is true, then it implies that in all part in pusa we should get tested where C 1 is true and this must be true in all states so we are saying that in all part globally this is true. So this is the way that we are going to write now.

You just see that this property I am writing $AG T_1$ implies $AF C_1$ this property is basically related to the process P 1 because we are having now two processes one process P 1 and process P 2. Now similarly, we can write a formula for process P 2 so it will look like or will come up like that AG if process P 2 is trying to enter into the critical section then in all part in future C 2 must be true so this is the property related to your process P 2. So process P 1 having this particular property, process P 2 is having this particular property.

(Refer Slide Time: 32:33)



But in case of safety that AG not of C 1 and C 2 this is for entire system so this is for both process P 1 and P 2. So now so this active property and we are having liveness property these two properties we have represented with the help of CTL formula. So these are the two formulas for safety and this is your liveness. Now we have to check these two property to be must true in this particular system. Initially just I am going to look for this one say AG not of C 1 and C 2. So if you see that we are having 8 possible

states in 8 possible states both C 1 and C 2 is not true; that means, in all state this is true because here in this particular portion C 1 and C 2 is not true but, here C 1 is true and C 1 is true but, C 2 is false and in this particular case C 2 is true, C 2 is true but, C 1 is false. So this AG not of C 1 and C 2 is true in this particular case.

(Refer Slide Time: 33:20)

Some CTL Properties

Safety: only one process to be in its critical section at any time.

$AG \neg (c_1 \wedge c_2)$


$AG \neg (c_1 \wedge c_2)$

Liveness: Whenever any process wants to enter its critical section, it will eventually be permitted to do so.

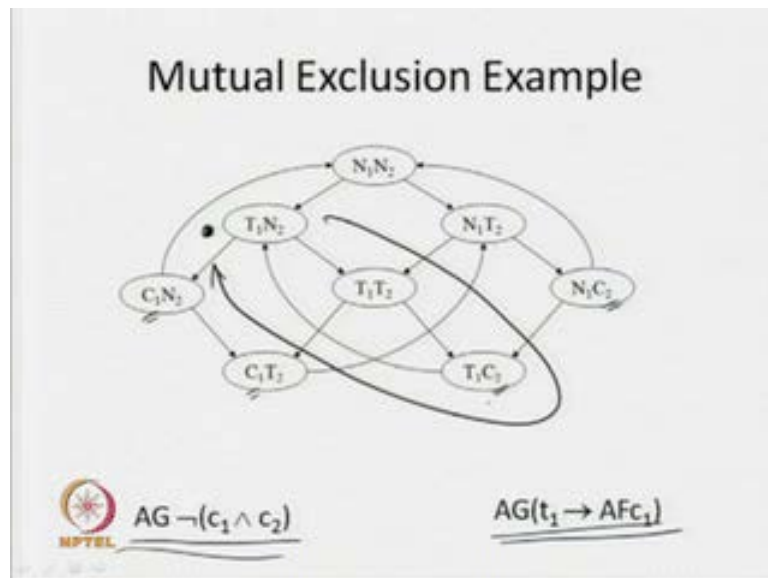
$AG(t_1 \rightarrow AFC_1)$

$AG(t_2 \rightarrow AFC_2)$

$AG(t_1 \rightarrow AFC_1)$



(Refer Slide Time: 33:32)

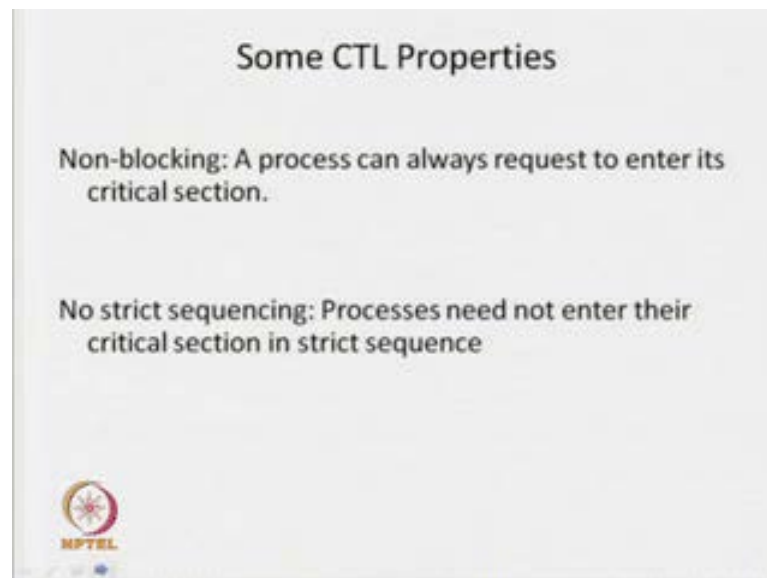


Now second property what we are talking about whenever any process wants to enter its critical section it will eventually be permitted to do so, if it is trying to enter into its critical section in all part in future we should get C 1. So in all part globally P 1 implies

AF C 1 so, in all part globally C 1 but, if you look into it here I will have been slight problem just I will just mention it say if you consider a I mean in this particular step, the process P 1 is trying to enter into its critical section. If I follow this particular part then it enter into its critical section but, it is having another part you consider this particular say from T 1 N 2 it is going to T 1 and T 2 then it is going for T 1 and C 2.

Now process P 2 is enter into the critical section then it will go back, now you consider this particular part now if you consider this particular part then what will happen? Will find that in this particular part we are not going to get C 1; that means, process P 1 is not entering into the critical section. So whatever as a this particular formula is formed; that means, model that I have come up over here is not setting at satisfying the liveness property because the first one is safety property we have safety, safety properties has been satisfied by this one but, when I am coming to this particular liveness property, we have seen that it is not satisfying this particular liveness property because we are having one particular part it will remain in this particular part and always C 2 is enter into the critical section.

(Refer Slide Time: 35:03)



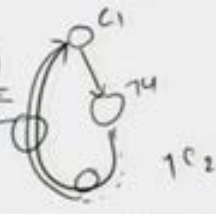
(Refer Slide Time: 35:30)


Some CTL Properties

Non-blocking: A process can always request to enter its critical section.

$$\text{AG}(n_1 \rightarrow \text{EXT}_1)$$

No strict sequencing: Processes need not enter their critical section in strict sequence

$$\text{EE}(c_1 \wedge \text{E}[c_1 \text{ U } (\neg c_1 \wedge \text{E}[\neg c_2 \text{ U } c_1])])$$




Now similarly, we have talk about the non-blocking and we have talking about the non strict sequencing. So we should not block any process to enter into their critical section and we should not put any strict sequence that it process must enter into the critical section in some predefine or strict sequence. So we should not put such type of criteria so these two properties also we can represent with the help of CTL formula. So it says that it is always request to enter into its critical section.

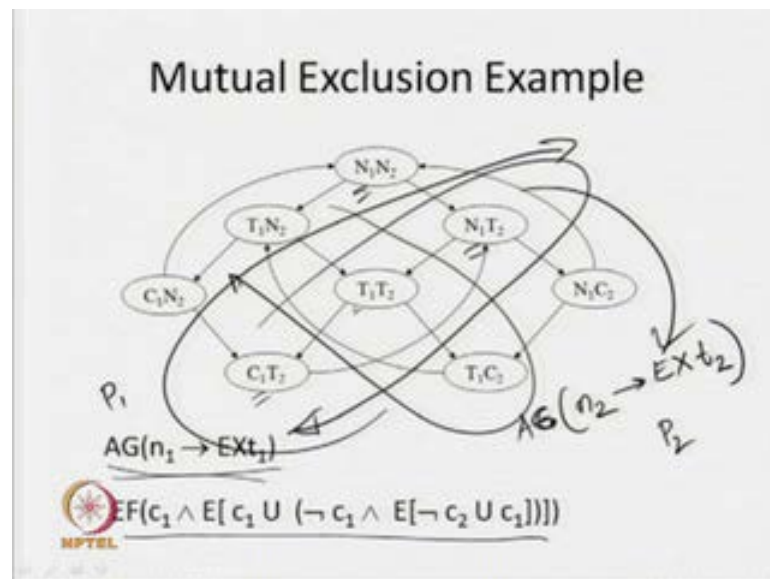
So if the process P 1 is in your non-critical section. So there exist a part in next state it is t 1; that means, if the process P 1 is in your non-critical section from the particular state we should get at least one part where it is trying to enter into the critical section; that means, we are not blocking wherever you are if you are in the non-critical section, we are having you are having a provision to try to enter into the critical section and that strict non-sequencing no strict sequencing we can represent this particular property with the help of this particular CTL formula, what it says that since it is not we do not put any strict sequence; that means, it is basically says that if it is entering into the critical section say it is in a critical section.

It may happen that after coming out from critical section it will again allow to enter into the particular critical section that is way we are saying if c 1 is true, and there exist a part c 1 will remain true until this particular property satisfies not of c 1; that means, c 1 it will go to become negation along with that c 2 will remain negation. So c 2 will remains

false until again c 1 become false you just see that here we are saying that we are in the c 1, we have entered say you can say that in this particular c 1 I can come out from this particular point will go to that particular side not of c 1; that means, now c 1 is false I am coming out from this particular critical section.

Now what will happen? You should get a part as such like that I can again my system will go to this particular c 1 but, in none of the state c 2 must be true; that means, c 2 will remain false; that means, c 2 is not entering into the critical section. So this is the way that I am representing this particular behaviour. So we are saying that not of c 2 remains true until again c 1 becomes true. So you just see that again this is a CTL formula because it is satisfies or this satisfying the other requirements or syntax about CTL expression you can check it because we have said that you are having some temporal operator until only we are reaching over here and this particular pusa also we are using and we are using this particular part quantifying e n. So this is the way that we are represent it is now we have to check whether this property is the true or not.

(Refer Slide Time: 38:12)



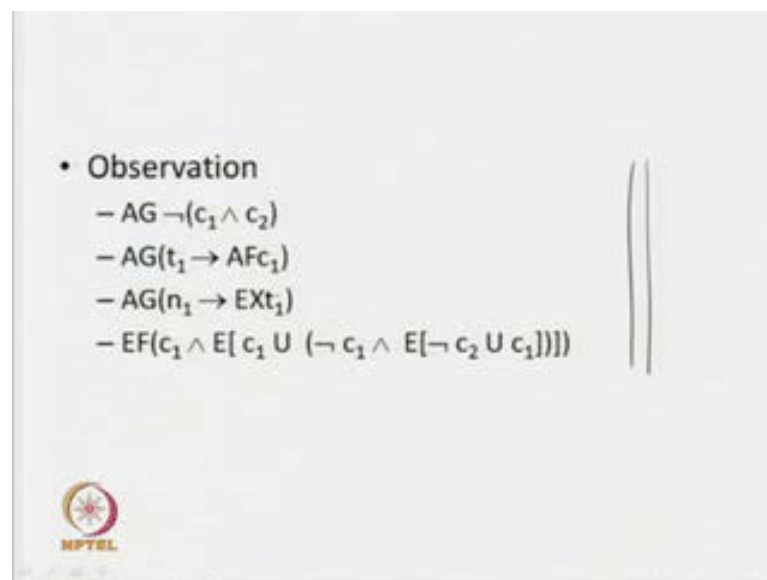
Now above this particular thing say $AG n_1 EX t_1$ you will find that wherever n_1 is there in next state I am having that t_1 I am having n_1 then you can select I am getting t_1 . So if it is in n_1 ; that means, it is in your non-critical section there exist a part in next step it is your t_1 . So it is getting a state where from this trying to enter into the critical section. So this behaviour is related to your process P_1 similarly, for process P_2 also I

can write the behaviour or property in all part globally if n_2 is true then it says the their exist a part in next state t_2 . So this is the property corresponds to the process P_2 and this is the property correspond to the process P_1 .

Similarly, here whatever I have seen wherever n_1 is true I am basically concern about this particular state and we are going to say that in next step is t_1 is true, because if n_1 is false basically this implication will always give may true will see this and this is again the property that we are having which is related to process P_1 similarly, for process P_2 what happens? We can said that this c_1 will be replaced by c_2 and c_2 will be replaced by c_1 then it will the property will be relevant for your process P_2 .

Again just I am giving you introduce idea about this you can say that this property is again true over here because you are going to get one particular part where this particular property is true, because this is you can correlate with the liveness property because we have said that in this particular part c_1 is not entering. So every time c_2 is entering so similarly, I am can look into this particular part now.

(Refer Slide Time: 40:58)



In this particular part what will happen? Always process P_1 is going into this critical section at no point of time process P_2 is going into the critical section; that means, in this particular part I am allowing to enter process P_1 to in its critical section. So in between process P_2 is not getting any chance; that means, we are not following any sequence of order c_1 can enter into the critical section for more number of times and

eventually we can say that after sometimes from here it will come out and it will go to this particular process P 2 will go into this process critical section.

(Refer Slide Time: 41:17)

• Observation

- $AG \neg(c_1 \wedge c_2)$
- $AG(t_1 \rightarrow AFc_1)$
- $AG(n_1 \rightarrow EXt_1)$
- $EF(c_1 \wedge E[c_1 U (\neg c_1 \wedge E[\neg c_2 U c_1])])$

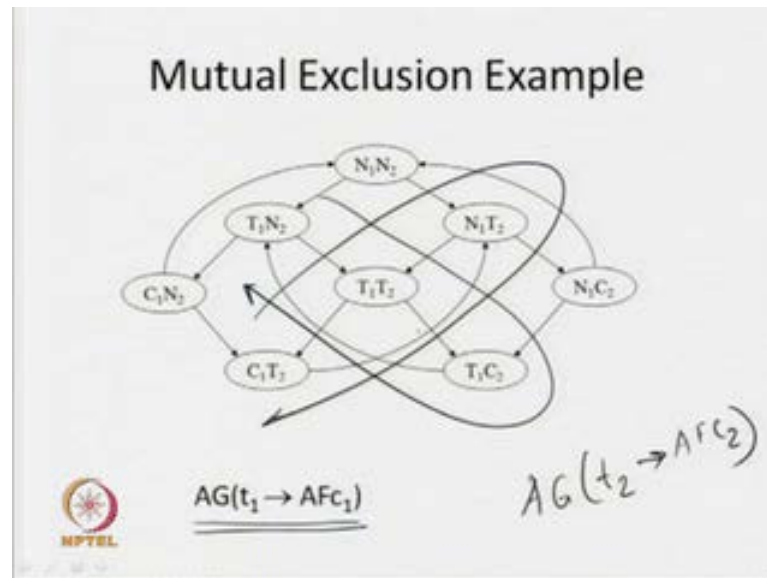
One property is not true: $AG(t_1 \rightarrow AFc_1)$

NPTEL

Now you just see that here we are talking about this particular mutual exclusion problem and we have come up with a model and along with that we have seen this particular poll property as specification this is safety, liveness, then no strict sequencing and non-blocking and out of this particular four formula what happens we have found that this property is not true in my model. Now say I am trying to design my this particular critical section excess of critical section or basically by mutual exclusion I have design it.

Now I have seen that it satisfy four requirements, four properties come up with four properties out of that four properties say one is not true it is not satisfied already we have seen. Now in that particular case now what I have to do? As a designer I should relook my design and try to modify my design in such a way that it is going to satisfy this property this is the way that we are going to design our system. So since this property is not true then I will revisit my design and I will try to check where is the problem and I will try to fix of that particular part.

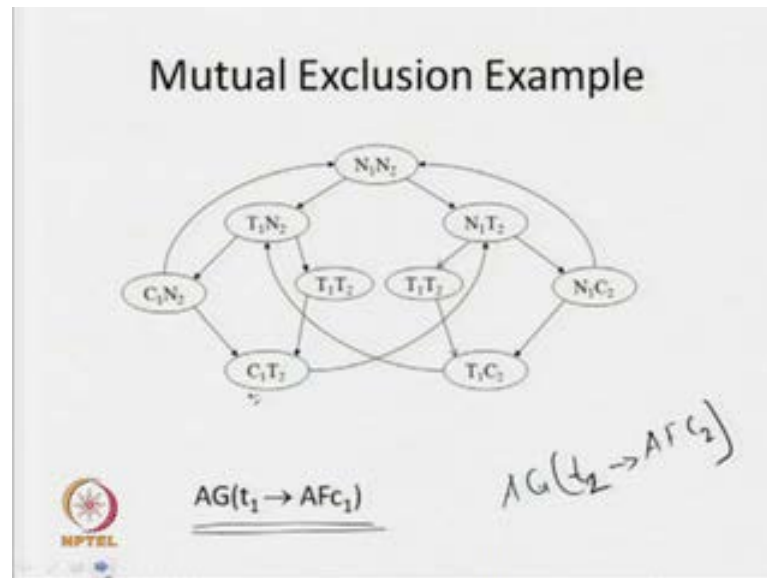
(Refer Slide Time: 42:11)



Now after looking into this what will happen? Now I can think something like that so, this is the model that I have come up and I have seen that this particular property is not true over here because already I have seen that this is the part that we have the problem; that means, in this particular part what will happen? Process P 1 is not getting any chance to enter into this critical section c 1 similarly, for process P 2 and having $AG t_2$ implies $AF c_2$.

Since this property is not true similarly, this property is also not true because you are going to get this particular part over here, where process P 2 is not getting any chance to enter into critical section; that means, you just see that I am having problem in this particular portion in my design I am having some problem over here, it may not be problem so I will concentrate in this particular part and I will try to modify my design. So in that particular case what will happen? Somehow we have to break this particular tool loops.

(Refer Slide Time: 43:42)



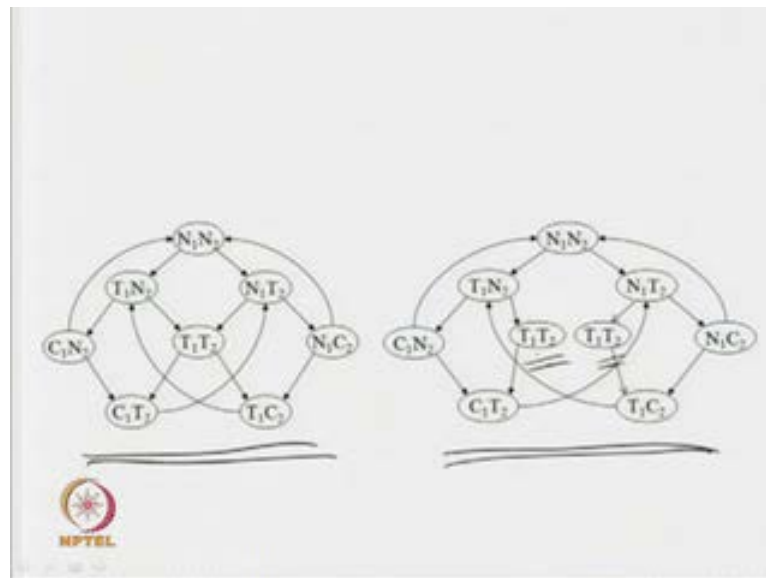
Now what will happen in this particular case. Now I can break this particular tool loops and I can come up with this particular design. So this is the way that we are going to design our system and we are trying to remove some errors of faults that we may have in our design. Now in this particular case now you just see that what will happen now earlier we have 8 states now we are getting 9 states, this particular states has been broken into two particular states t_1 and t_2 has been broken into two different states. So now in this particular case what will happen you just see that now when I am having that t_1 it is trying to enter into the t_1 then it is entering into your c_1 and after that I am coming back to that particular state where n_1 and t_2 .

Now from here that either we can go to t_1 and t_2 or it can get n_1 and c_2 ; that means, it is already entered into this particular critical section. So this is way that we can form here we can say that t_1 and c_2 from here we are going to t_1 and n_2 and eventually I am coming to this particular c_1 either from this direction or from this direction. You just see now that after breaking this particular tool loops I am getting this particular design. In this particular design we find that these two this formula is true over here similarly, that other formula is $AG(t_1 \rightarrow t_2 \rightarrow AFC_2)$ will also make it; that means, that liveness property will be satisfied over here.

So this is the way that we are going to sets after applying this particular model verification technique I have to check this particular property I have to found that there is

some problem. So now we keep it our design and we modify it and we have seen that now this particular design is going to satisfy my liveness property. Now once it is satisfying my liveness property then we have to see whether it is not my other three properties in this now we can say that other three properties are also remaining take over here, you can recheck it over here that other three properties are true over here.

(Refer Slide Time: 45:44)

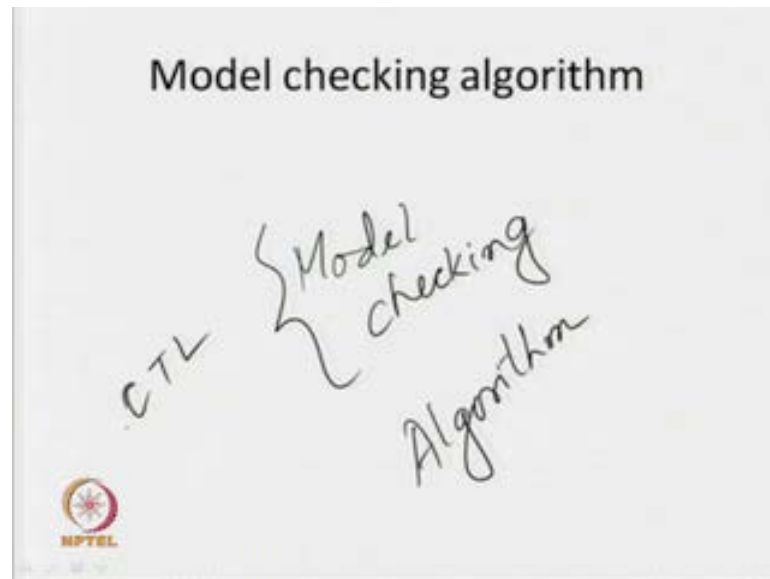


Now basically you just see that we have starting our design we have come up with a model M and after looking for the properties we try to check whether these properties are true over here or not and we have found that it is not satisfying some of the property, we have revisit our model and we have come up with these particular model, second model and in this particular second model we have seen that at least it satisfy all those particular force specification of four properties that we have. Now say as for the some of CTL formula we are having slight problem over here that two states cannot be marked with the same level, somehow they have to be different.

Now here I am not talking into this particular internal details but we can say that instead of this particular 6 states that I am having out of the process is that $N_1 T_1 C_1$ and $N_2 T_2 C_2$ we may have some of the internal signals also in the controller so, these two states will be distinguished by some of this particular control signals in that point of time we are look into that we can think that they are different because some of the control signals which are not relevant with respect to our properties so we are not showing them

marking of this particular properties.

(Refer Slide Time: 47:22)



Now what we need next time we are having that model, we are having the properties CTL properties, we are representing our specification with the help of CTL property. Now we need a mechanism by which we are going to check that those particular properties true in this particular model or not. So for depth we are going to look for a particular model which is known as your model checking. So we are going for this particular model checking and basically we are going to talk about CTL model checking because our special case will be represented by CTL formula and we are going to look for an algorithm by which we are going to check whether particular CTL formula is true in our model or not. So we are going to look for model checking algorithms. Now what is this model checking algorithm how we can visualize this.

(Refer Slide Time: 48:00)

Model Checking Algorithm

Given the model M , the CTL formula Φ and a state s_0 of S as input

Model checking algorithm generates answer 'yes' ($M, s_0 \models \Phi$ holds), or 'no' ($M, s_0 \not\models \Phi$ does not hold).

Handwritten notes on the left: $M = (S, \delta, L)$, M, Φ, s_0

NPTEL logo at the bottom left.

(Refer Slide Time: 49:25)

Model Checking Algorithm

Given the model M and a CTL formula Φ as input.

Model checking algorithm provides all the states of model M which satisfy Φ

Handwritten notes on the left: $M = (S, \delta, L)$, Φ

Handwritten notes on the right: M, Φ, s_0

Handwritten notes at the bottom: $(s_i, s_j, s_k \dots)$ with a circle around it and Φ next to it.

NPTEL logo at the bottom left.

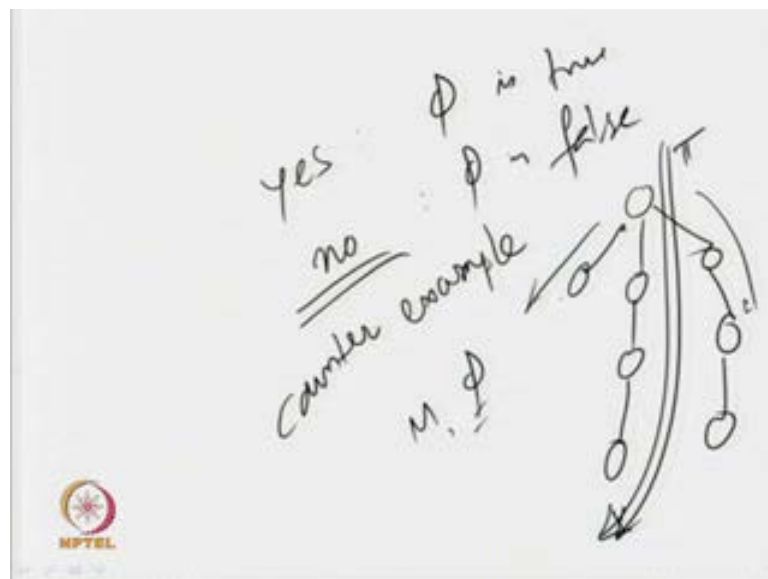
You said that given a model M so we are having a model M the CTL formula ϕ . So we are having a model M , we are having a CTL formula ϕ and along with that we are having a state S of that S ; that means, we are going to look for any state that S_0 which belongs to set of states S of this particular model because model M is basically defined by your state of set transitions function and labelling function. So this is the critical. So what is the model checking algorithm? So model checking algorithm generates answer yes, it will say yes, if $M S_0 \models \phi$ or; that means, you can say that ϕ holds in your $M S_0$ or ϕ is true in S_0 so that model checking algorithm generate answer here is true,

if ϕ is true in this particular state S_0 or it generate answer no if ϕ does not holds over here so M, S_0 models ϕ does not holds means ϕ is not true in S_0 .

So this is the way we can look into the model checking algorithm that we are giving a model M , we are giving a CTL formula ϕ and one particular state and we are going to check whether this particular CTL formula is true in this particular state S_0 or not so it is going to say either yes or no. Now model checking algorithm may be pure in the another way also it says that we are giving model M and a CTL formula ϕ so we are giving a model M in a CTL formula ϕ . So model M again we are saying that it is having state of state transition will be lesser and labelling functions.

Now model checking algorithm we can now view like that, model checking algorithm provides all the states of model M which satisfies ϕ . So we are going to giving the entire model M now, the model checking algorithm will be run in such a way that it is going to give me the set of states I can say that s, S_i, S_j, S_k like that some states we are having in which states this particular formula ϕ is true. So you can think the model checking algorithm in this particular model so, in one case in first case what we are thinking, you are giving a particular state and we are going to check whether ϕ holds over there or not.

(Refer Slide Time: 51:41)



In the second case what we are thinking that I am giving the model and formula it will give me all the states where this particular formula is true. So if you know the method for

one, the second one can be always derived like that say if I am giving this particular method say i giving the model M and formula ϕ and it is going to give me all the states where ϕ is true. Now if I say that I am giving a model M formula ϕ and particular state say S_0 whether this formula ϕ holds S_0 or not.

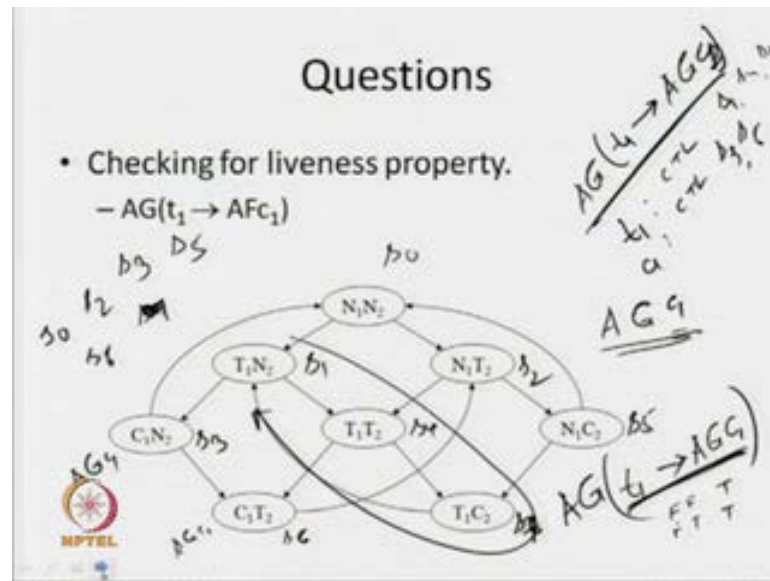
So this method is returning me all the particular states where this particular ϕ is true. So I am going to check whether this S_0 is a member of this particular return side or not, if it is a member of this particular return we can say that yes ϕ is true in; on the other hand if I know this particular method then very well I can find out the particular all the states in this particular formula is true because I am going to run this particular method on each and every states of the model and where it is true it is going to give me this particular state.

So thus see that these are specifically complemented to each other if I know one, the other can be derived and this model checking algorithm is having another advantage also or another feature also or that it is I just say either it is going to see yes if ϕ is true and it is going to give me answer no if it is ϕ is false. This formula is false. Now when it is no, it is false, generally that model checker generate a counter example also what is this counter example? It counter example gives me the execution stress test it says that where this particular formula is false, basically I am having a giving a model.

So in this particular model say I am giving a formula say ϕ if ϕ is not true then it will said that it is not true in this particular part then it will give me this particular execution this particular part says that the particular formula is not true in this particular part. So; that means, it is give me a feedback to the designers when the formula is not true; that means, the problem with my design I have to revisit my design and I have to redesign it but model checker is giving me some clue that my error is somewhere in this particular execution part.

So I can concentrate over here and I can now try to redesign it in such a way that this formula becomes true in my model but, when I am trying to basically concentrate on this particular part some error may occur in some apart from that so I must be careful about those particular issues also but, model checker is giving me a some hints so that I can quickly try to fix my parts. So this is the issues of our model checking so, in our next class we are going to discuss about this particular model checking algorithm

(Refer Slide Time: 53:25)



Now just look for some question that we are talking about this particular mutual exclusion problem. Now I am going to look for the liveness property, checking of this particular liveness property how I am going to check it? Or how we are going to check it? Now you just see that I am having $AG t_1$ implies $AG c_1$. So this is the CTL formula liveness property that we have given so here t_1 is a atomic propositions, so it is a CTL formula c_1 is a atomic propositions so it is also a CTL formula since there atomic proposition so we know the labelling function and we know the states where this particular formula or CTL formula is true.

So from labelling itself will say that this is say I can numbering it so this is S_0 , this is $S_1, S_2, S_3, S_4, S_5, S_6, S_7$ so t_1 is true in your S_1, S_4 and $S_6, S_1 S_4$ and $S_6 S_1, S_4, S_6$ so t_1 is true over here where c_1 is true will find that c_1 is true in your S_3 and S_6 . So these are the CTL formula and we know that the true values on those particular state. Now what is the next formula that we have to look into it so these are atomic proposition so this as these are the CTL formula.

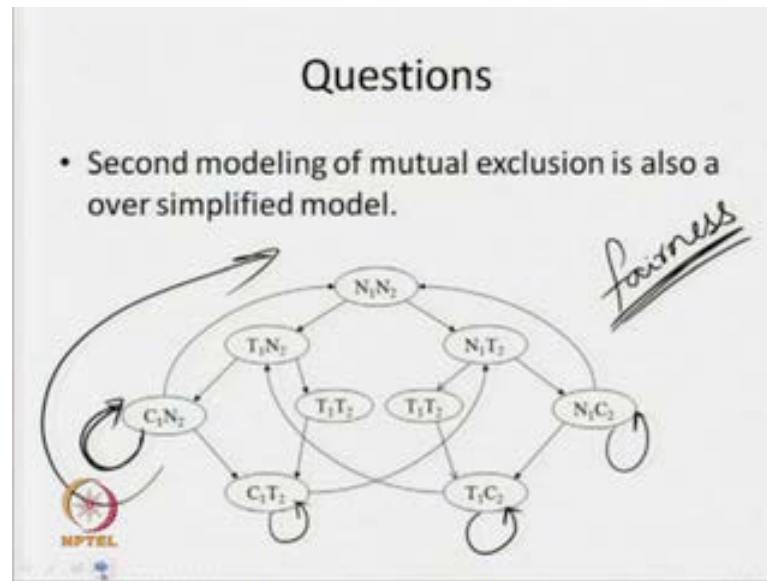
Now the next formula will come as your AG all part globally c_1 , since c_1 is a your distance what you call c_1 is a CTL formula so AG is also a atomic propositions. So in all part globally whether it is true or not. So if you look into the then you will find that $AG c_1$ is not true over here. So basically, if you look into it all part globally, you will find that as per our semantics since c_1 is true over here I can say that all part globally c_1

is true over here all part globally c_1 is true over here. So these are the issues that we are having, now in this particular case the next CTL formula will come as your t_1 implies $AG\ c_1$.

Now in this particular case you just see that if t_1 is true then $AG\ c_1$ must be true. So if t_1 is false then obviously, this formula will false because false we know that in case of implication if it is your say false and false it will going to give me true and false and true also give me is true. So if t_1 is false then this particular formula is true. So in this particular case if I look into the t_1 implies as a c_1 then it will be a true in your S_0 because t_1 is false, it will be true in your S_2 , it will be true in your S_3 because your t_1 is false it is not your.

So it will be true in your S_5 , it will be true in your S_6 and it will be true in your S_7 . So this is your S_7 because sorry it is not true in your S_7 because if t_1 is false then only it is true, so these are the states that we are having 1, 2, 3, 4, 5 in this five states this is true but, now we have to concentrated about this particular three states what will happen to t_1 implies this as c_1 we just see that t_1 is true, then in all part globally c_1 was true. So already we have seen that in this particular part nowhere I am going to get c_1 so in this particular three states this particular formula is false now we know the levelling of this particular formula then I can look for the complete formula $AG\ t_1$ implies $AG\ c_1$. Now you just see that in this particular case what happens? When we are going to look for a true values of these particular formula, then we must know true values of each and every sub formula. So in our model checking algorithm now we are going to do in this particular where that we are going to look can go for each and every sub formulas one it is done then we look for a main formula.

(Refer Slide Time: 58:20)



So another one is the CTL formula, now we are having a second modeling of mutual exclusion where I have break this particular step T 1 and T 2 to different states T 1 and T 2 and I am saying that the second modeling of mutual exclusion is also a over simplified model. Why I am saying this is a over simplified model because you just see that if someone is entering into the critical section I am saying that after that it is going for non-critical section but, now how many plans it is going to be in this particular critical section. This particular model is not going to critical section for how long it will be if it is trying state. If it is trying state it can go the c 2. Now how long this will be in this particular critical state c 2 so, if I am going to give this particular information basically I can give some sort of this particular self loop over here like that; that means, it says that it will be remain over here then it will come out.

Now once I will give such type of self loop then what will happen? Again it is going to highlight my liveness property because now I will be remain in this particular case. Now that is why I am saying the second modeling is an over simplified model but, if I will come up with this particular self loop I will say that it may remain in their for more number of plans and it will come out then what will happens still we can try to look for checking those particular property at such liveness will be false over here but, what we can do? We can use some fairness of model checking to these are the issues.

(Refer Slide Time: 60:58)



Either I can come up with the simplified model or I can try to depict the entire information where it may not highlight the come out of properties but, what will happen? We are going to use some fairness execution in this particular model or we are going to use some fairness constant. What is this fairness basically liveness immense with this particular step forever but, we know that in any system it will enter into the critical section, it will do this job for required job, it will take some amount of time and eventually it will come out; that means, we are going to look for a fair execution only that means it will not remain over here infinitely, eventually it will come out after completion of the job it will come out; that means, what we are going to say that, we are going to look for a fair execution only; that means, what happens I can think some model.

Now you just see that in this particular loop the model will remain infinitely; that means, if my system remain in this particular loop then I am going to have some beg behaviours but, in fairness what I am going to say that if my system is correct I know then what will happen eventually it is going to come out from this particular loop and eventually it will follow this particular part; that means, we are going to look for only fairness execution we can somehow omit this particular infinite loops; try to omit this particular part and you will see that eventually it will come out from this particular loop and it will going to execute this thing.

So in this particular test with this particular fairness from strength we can check for those particular property because while I am designing my system it is coming actually I cannot break this particular loop because it may be there but, my model checker is giving me some problem or it is reporting that it is having some error. So I will say that eventually my system will come out from this thing so basically, loop to check my property in the fairness path. So fairness constant we are going to look for fairness path; that means, my system will go to the fair paths of the model only. So like that we can incorporate planners and we can go for a model checking on the same system only without breaking this particular loop, with this I will stop my lecture today. So in next class we are going to look for a model checking algorithm.