INDIAN INSTITUTE OF TECHNOLOGY GUWAHATHI

NPTEL

NPTEL ONLINE CERTIFICATION COURSE
An Initiative of MHRD

VLSI Design, Verification & Test

Prof. Jatindra Kr. Deka
Department of CSE
IIT Guwahati

Module V: Verification Techniques

Lecture IV: Model Checking with Fairness

Okay we are disusing about model checking algorithm.

(Refer Slide Time: 00:28)

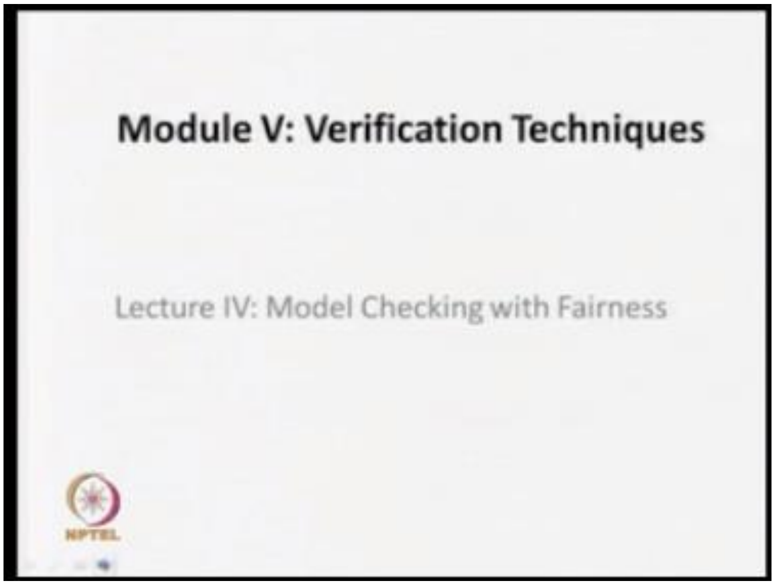
NPTEL Phase-II
Video course on

Design Verification and Test of Digital VLSI Designs

Dr. Santosh Biswas
Dr. Jatindra Kumar Deka
IIT Guwahati

And in this model checking algorithm what happens.

(Refer Slide Time: 00:30)



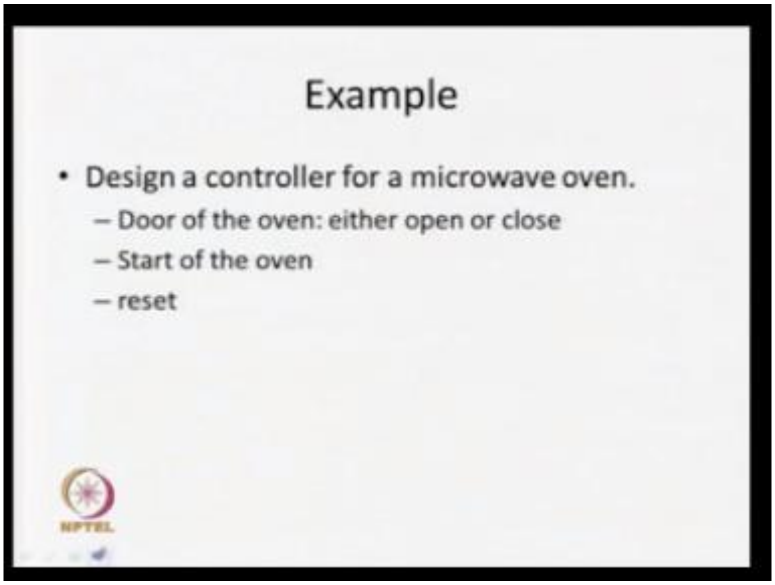
Module V: Verification Techniques

Lecture IV: Model Checking with Fairness

We have seen that we need one model as input and one CTL formula as another impute and our model checking algorithm will look into the states where this particular formula is true okay now today we are going to see this particular model checking algorithm with some other constant which is known as fairness constant why we have to look into the fairness constant why it is required for that we will just see our general model checking algorithm with an example.

We will try to device an example we will go through the example and later on I will say what this fairness is our coming into picture and how it is going to help us during model checking.

(Refer Slide Time: 01:44)



Example

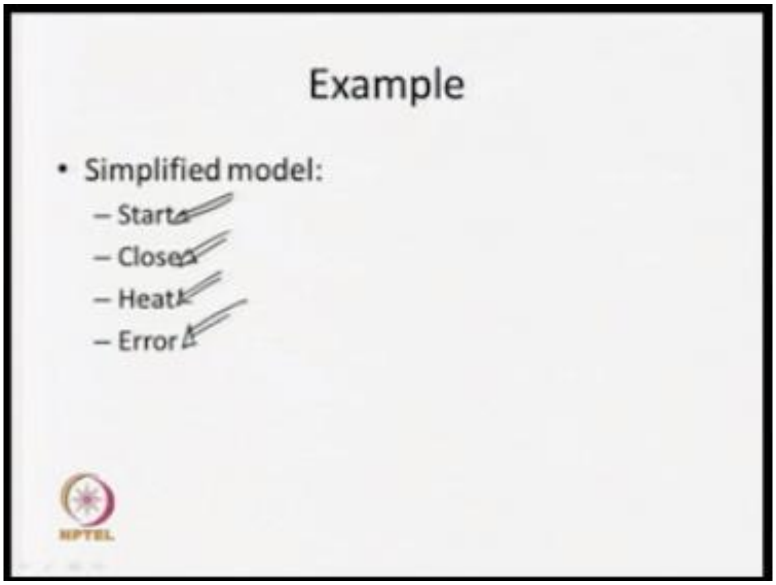
- Design a controller for a microwave oven.
  - Door of the oven: either open or close
  - Start of the oven
  - reset

Okay do that first we are going look a design issues that one example design a control of what a microwave oven all of you know that micro oven is used for cooking food so it is atomic so we should have controller to control the operation of this particular network work now when we are going to design a controller initially we are going through look for an abstract model of that particular controller and depending on that particular abstract model will design it later on we can refine it and we will go more details okay.

Now when you look into then what is the requirement for this particular controller we know that there us oven for your microwave oven we can either open the door to put food item inside the oven and we can close the door that means we need some mechanism and generally it is a sensor which will sense the signals and say that right the door is open and door is close later on after putting the food items and closing the door we can start the oven.

When we start the oven then what will happen the heating coil will heated up and eventually the food will be cooked so by looking into all these issues we can come with a very simplified model initially simplified abstraction and we will find that.

(Refer Slide Time: 02:36)



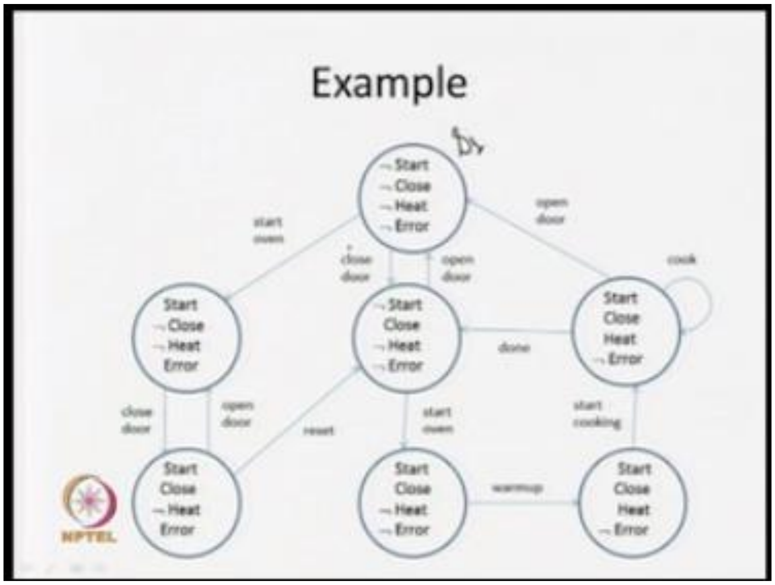
Example

- Simplified model:
  - Start
  - Close
  - Heat
  - Error

Some of the things are some of the signals required to model this particular controller so those signals are basically were occur abstracting out and we are saying that first one is start it is going to say that the oven has been started now it is going through out cooking our food item closed basically indicate that whether door close or door is open when we start that heating coil will be heated up and food will be cooked so we can say that another important issues that we need is it and sometimes there oven may go into the error satiations so you can take another signals called errors.

So with the help of this force signal we are trying to design our controller so this is some sort of abstraction later on we may have to refine it and we can go for the compaction but first we will see this particular abstract model and we will see how we are going to look for the property that as to be satisfied this particular model.

(Refer Slide Time: 03:34)



Example

Now with this particular things now what you see that now we have to look for the states so basically we can see the operation of our microwave oven just see that first we are saying that this is a state I can make it as S1 I am saying that naught of start naught of close naught of heat naught of error.

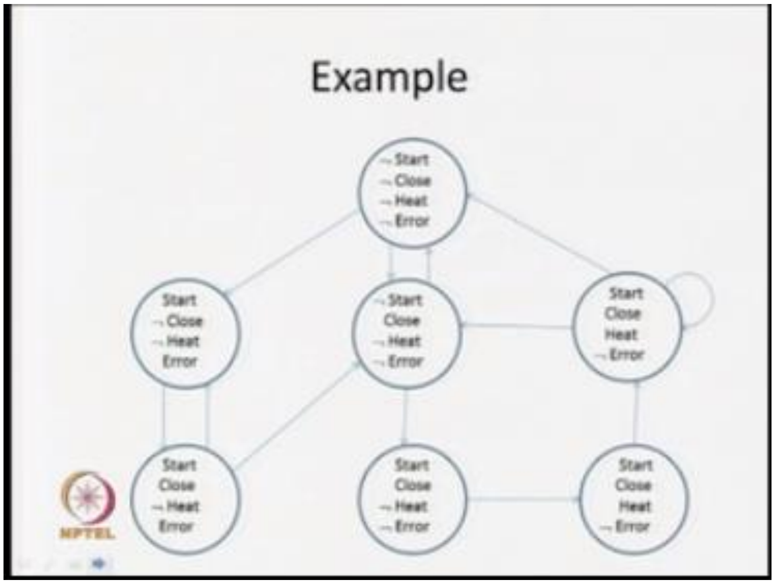
That means the microwave oven is in ideal satiation it has not started door is not close means door is open you can say it is not heated up and there is no error combination now these are the action that I can say that I can close the door then it is coming to this particular state weather naught start close heat and error because it was door was open now door is closed now one close the door is am I again open it because you may have to put some more items so he can be in this particular lower he can close the door he can open the door.

But sometimes it may happen that they are in the ideal situation ideal condition now some body as start the open now it is not close that means door is open so in this particular here started the oven then now what will happen this start becoming to true and it as gone to error condition now when it as gone to the error condition now user might have realized then he is going to close the door but again in this error condition he will open the door so he may be in this particular loop trying to open.

So we have given an option reset bottom through reset you can come down this particular case now say my door is close it is not in error condition once we are coming to this thing we can start the oven so start is true close and note it down then we can go for the worm up that means you are starting the heat so heating coiling is ring up then we can start cooking so it will remain over here and it will when it is complete so I am coming to this particular thing that door is closed and you can open the door.

Abstracter we can open the door and come to this value so this is the simplified operation of a microwave oven so now in this particular case now you should when we go for model checking we need a structure what is a kipper structure it is having set of states we are having transition and we are having leveling function so these particular evens are not required for a model see that start open close door open door reset warm up all those things are not required but we will leave the other in function door states these leveling function and this transition.

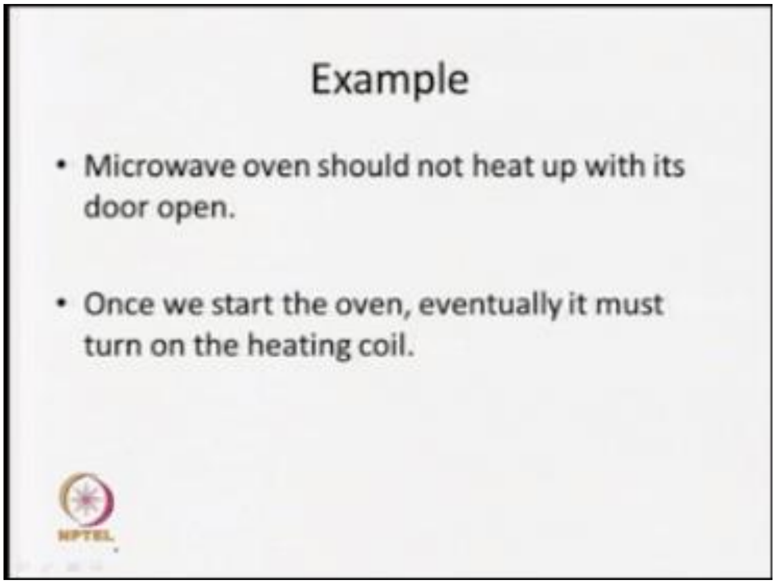
(Refer Slide Time: 05:58)



Example

So basically whatever you can say that eventually we have come up with this particular model and this is a simplified model and this in this particular model or in this kipper structure we are going to check some of our properties okay.

(Refer Slide Time: 06:10)



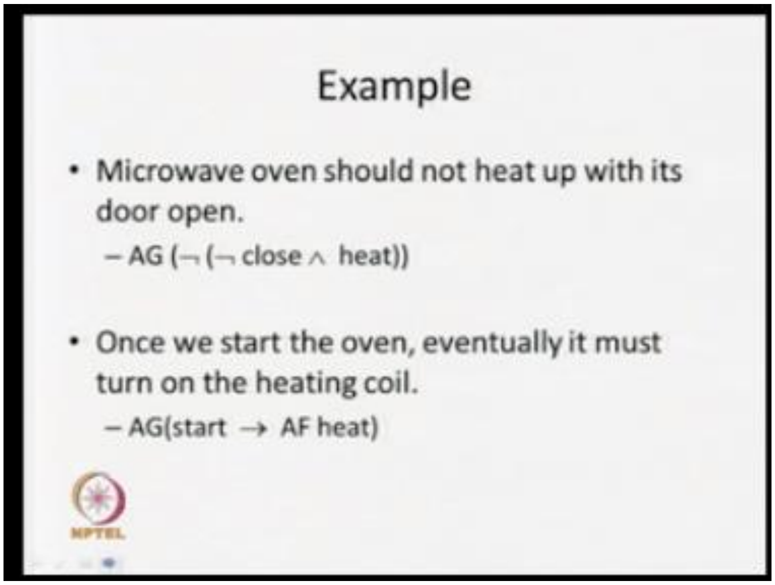
Example

- Microwave oven should not heat up with its door open.
- Once we start the oven, eventually it must turn on the heating coil.

Now what are the properties that it should satisfy so I am just coming up with two example first one I am saying the microwave oven should not heat up with heat door opens so on is door opens so coil should not get heated up state eventually put some of water second property we are saying that once you start the oven eventually heat most trun on the heating coil so once you start the oven it should go for the cooking mode so heating called mode way heated up.

So these are the two properties now how we are going to check this so we now that we are looking for CTL model checking so somehow we have to represent these two properties in CTL formulas.

(Refer Slide Time: 06:48)



Example

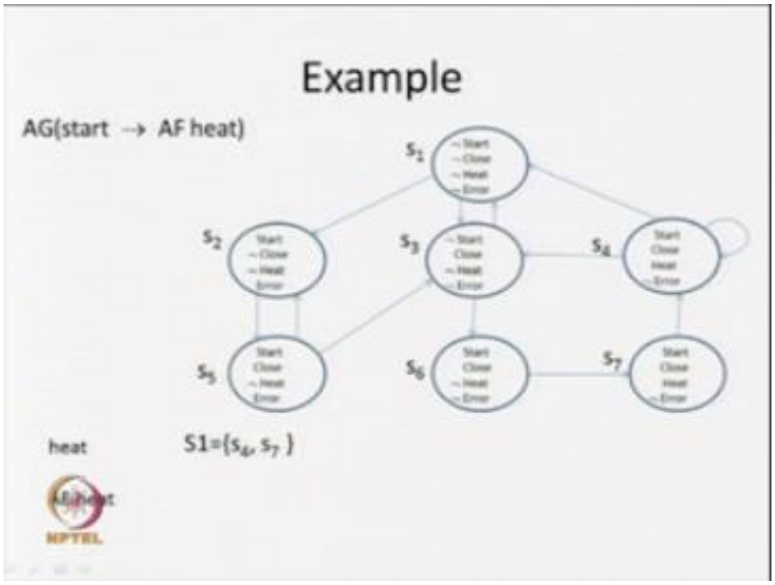
- Microwave oven should not heat up with its door open.
  - $AG\ (\neg\ (\neg\ close \wedge\ heat))$
- Once we start the oven, eventually it must turn on the heating coil.
  - $AG(start \rightarrow AF\ heat)$

So the first is very simple so basically¬of close and¬of heat should not be two together so that is why I am saying¬of close and heat so these two combination¬ of close that means door open and heat it should not be true together so negation in front of that we are putting that negation and it should hold globally so in all part globally these property must hold second one I am saying that once we start oven eventually it must turn on the heating coil that means when I truing on the heating coil that am so means oven will go into the heating conditions so I am saying that in all part globally start implies AF in all part in further it is true.

So I am capturing this particular property with the help of this particular CTL property now I have this two CTL property according to my requirements and we are going to check whether these two properties true one if you look into the first property it is simply based on the atomic proposition I am having atomic proposition close and one atomic proposition heat so we have to check weather in which situation in which state they are true or they are false and depending on that I can look for AG.

And second one I am having one start and heat again AF and AG is involved in both are so I am going to check the property for a second first one will be simpler one and you can check it yourself we will see how I am check this particular second property okay now we are having this particular kipper structure of model of our system.

(Refer Slide Time: 08:18)



Example

AG(start → AF heat)

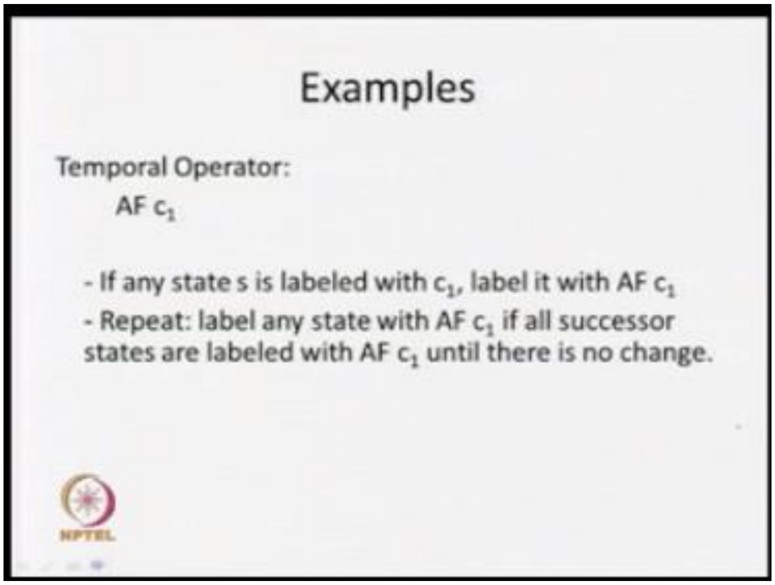
heat $S1=\{s_4, s_7\}$

AF heat

Microwave controller and I told you that we are going to check for the correctness of this formula in all part globally start implies AF heat so first we are going to see that we have to look for each and every sub formula first we are look for this particular sun formula heat now by looking in to this kipper structure you will find that heat is true instead S4 and S7 so this is a S4 and S7 this is heat is true in this diagram.

Now by looking into this is since it is true now we have to go for we know where which are states where state heat is true so we are getting S4 and S7 the we will go for the sub formula AF heat so AF heat now we have today how we are going to look for the correctness of this particular AF heat.

(Refer Slide Time: 09:07)



Examples

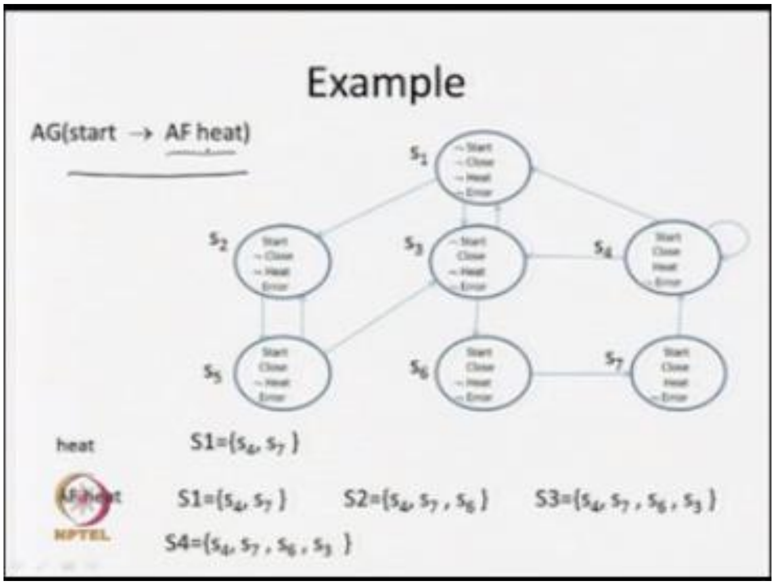
Temporal Operator:

AF c_1

- If any state s is labeled with c_1, label it with AF c_1
- Repeat: label any state with AF c_1 if all successor states are labeled with AF c_1 until there is no change.

So how we are going to do we know this temporal operation AF c1 now what will happen if any state that is S is level with c1 level it with AF c1 this is the first states then we are going to repeat this level any state with AF c1 if all successor states are leveled with AFc1 until there is no sense so we know this particular state now we are going to apply this algorithm to loop for the states where AF it is true.

(Refer Slide Time: 09:38)



Example

AG(start → AF heat)

heat $S1=\{s_4, s_7\}$

AF heat $S1=\{s_4, s_7\}$ $S2=\{s_4, s_7, s_6\}$ $S3=\{s_4, s_7, s_6, s_3\}$

$S4=\{s_4, s_7, s_6, s_3\}$

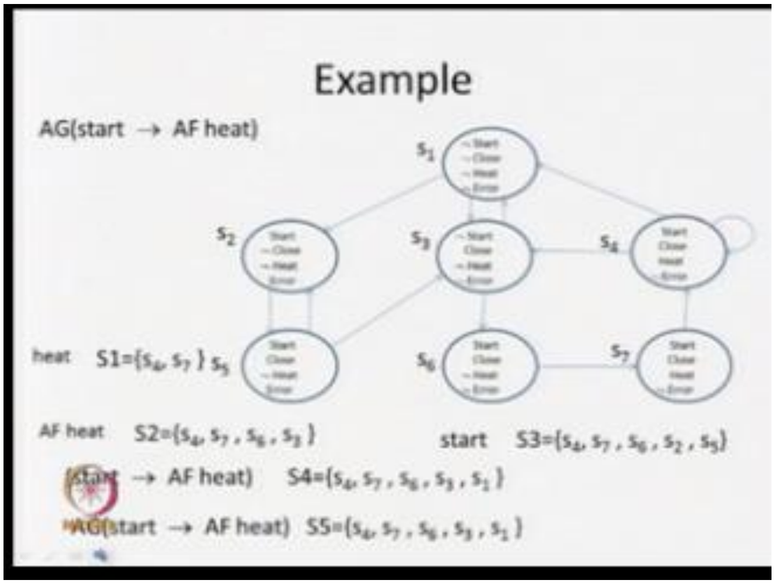
So now in this particular case so we know that AF heat is true in S4 and S7 because heat is true in this these two states so this is our first state now after that we will go into that particular repeat step now what will happen next we are going to say where it will goes so S4 and S7 we are getting it now we are going to see what are the perdition state of this particular true state so S4 it is having it is own perdition and S7 is having S6 as a perdition now only perdition.

Now what will happen we are going to level a state if all are with successor r leveled with AF heat since it is having one successor and it is leveled with your AF heat so S6 will also be leveled with AF heat so in AF heat will be true now S4 and S7 and S6 now since there is a sense out of states so we are going to repeat this particular process now if you look into then S6 we are going to get this S3 and perdition.

And this particular case it is having this S3 having this particular successor and this is the scenario that we having now what will happen it is having this particular perdition as string now we are see all with successor now what are these successor one is your s6b okay and second one is your S1 so in case of S3 we will find that it is having this particular successor so S3 will be leveled with your this particular AF heat next we will go and will find that it will remain is your F3beacsue when I am coming to this particular s3.

So it is having this perdition your S1 and it is having this perdition your S5 but they are leveled with your AF heat so it cannot include S1 and S3 S5 so our remain state remain same S4 S7 S6 and S3 so it is rested where AF it is true now once we know the state where AF heat is true now we can go for this particular sub formula start implies AF heat that is if start is true then AF must be true.

(Refer Slide Time: 11:54)



Example

AG(start → AF heat)

heat S1={s_4, s_7}

AF heat S2={s_4, s_7, s_6, s_3}

start S3={s_4, s_7, s_6, s_2, s_5}

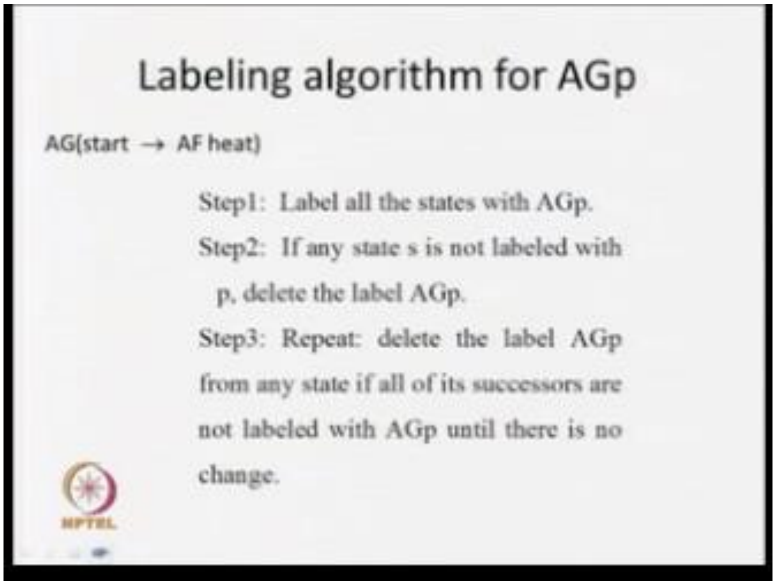
(start → AF heat) S4={s_4, s_7, s_6, s_3, s_1}

AG(start → AF heat) S5={s_4, s_7, s_6, s_3, s_1}

So in particular case how I am going to say that these are state S4 S7 S6 S2 and S5 these are the state I am getting because it is not of Start of AF heat so we know that AF heat is true for S4, S7, S6, S3 so these are the states and along here not of state is true over here so not of start because we know that P implies q is your not of P or Q so in this particular four states your f it is true and not of states is true in S1.

So S5 will also come into your this particular scenario so ultimately we are going to get these are the states where state is true and eventually you are going to get AF start implies AF is true in your S4, S7, S6, S3 and S1 so I am going to get this particular state, now after that once you kow the leveling of this particular sub formula then you can go for means of formula, so in this particular means of formula.

(Refer Slide Time: 12:56)



Labeling algorithm for AGp

AG(start → AF heat)

Step1: Label all the states with AGp.

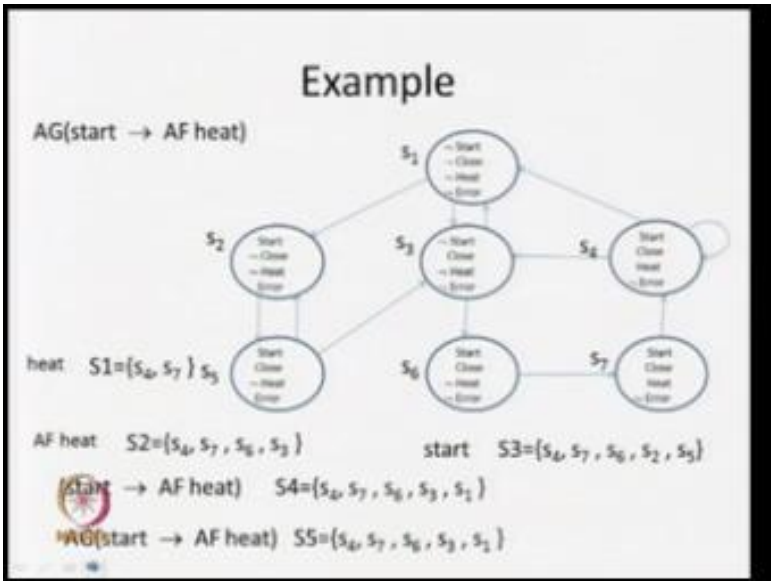
Step2: If any state s is not labeled with p, delete the label AGp.

Step3: Repeat: delete the label AGp from any state if all of its successors are not labeled with AGp until there is no change.

What will happen now you are having this particular AC state implies AFC heat we know the procedure now what is the procedure for Ac so how you are going to get, so in step one level all the stage with AFP any state s is not leveled with P delete the level ACP okay basically initially we are going to check where ACP is true then we will repast this particular steps during the level ACP from any step if all of its successor are not leveled with ACP until there is no sense so we are going to repeat this step 3 so if you look into this procedure then what will happen.

For AGP start initially we are going to level the all state then we will remove where it is not leveled with your start implies AF heat so these are the states where start into there is true so AC start AFP will be true in your S4, S7, S6, S3 and S1 in this particular five state it will be true this true state will not come, now we are going to repeat this particular step 3 delete the level AGP from any state if all of its successor are not leveled with AGP until there is no sense.

(Refer Slide Time: 14:12)



Example

AG(start → AF heat)

heat $S1=\{s_4, s_7\}$

AF heat $S2=\{s_4, s_7, s_6, s_3\}$

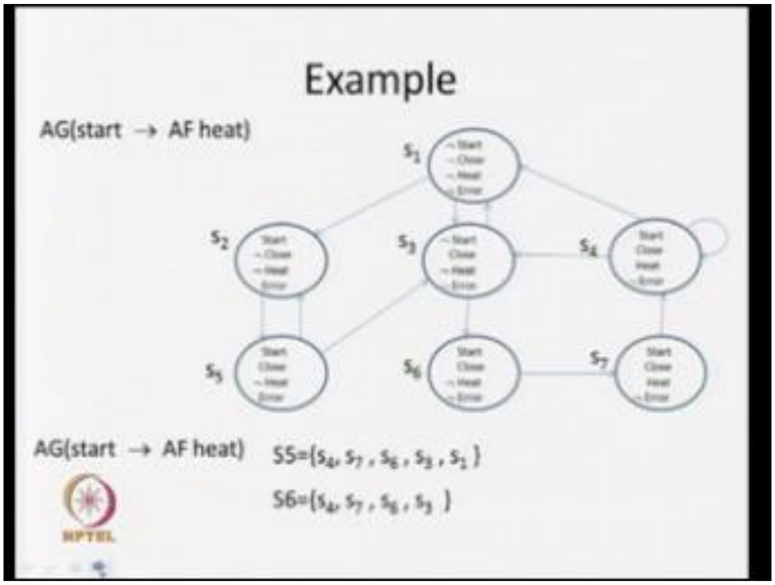
start $S3=\{s_4, s_7, s_6, s_2, s_5\}$

(start → AF heat) $S4=\{s_4, s_7, s_6, s_3, s_1\}$

AG(start → AF heat) $S5=\{s_4, s_7, s_6, s_3, s_1\}$

So I think with this particular state S4, S7, S6, S3 and S1 now we are going to repeat this step and see what are the states will eventually remain over here.

(Refer Slide Time: 14:25)



Example

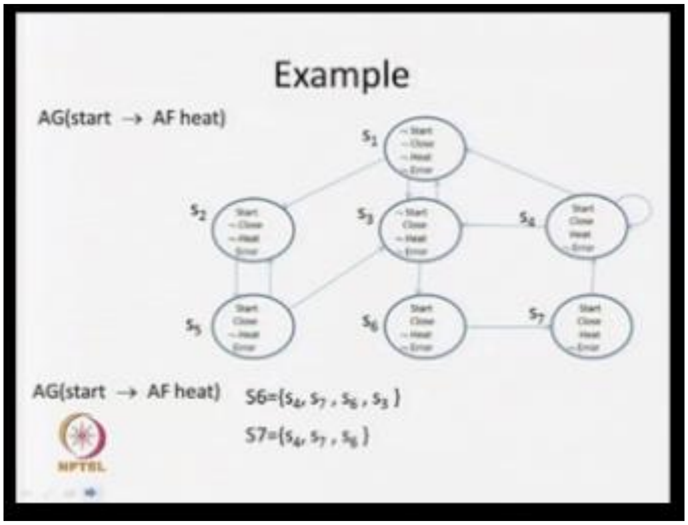
AG(start → AF heat)

AG(start → AF heat) $S5=\{s_4, s_7, s_6, s_3, s_1\}$

$S6=\{s_4, s_7, s_6, s_3\}$

Now in this particular can you see that S1 it is having successor S3 and it is having successor S2 since all the successor are not leveled with start implies AFA so we have to remove S1 from it state so eventually I am getting this set as S4, S7, S6, S3.

(Refer Slide Time: 14:53)



Example

AG(start → AF heat)

AG(start → AF heat) S6={s_4, s_7 , s_6 , s_3 }

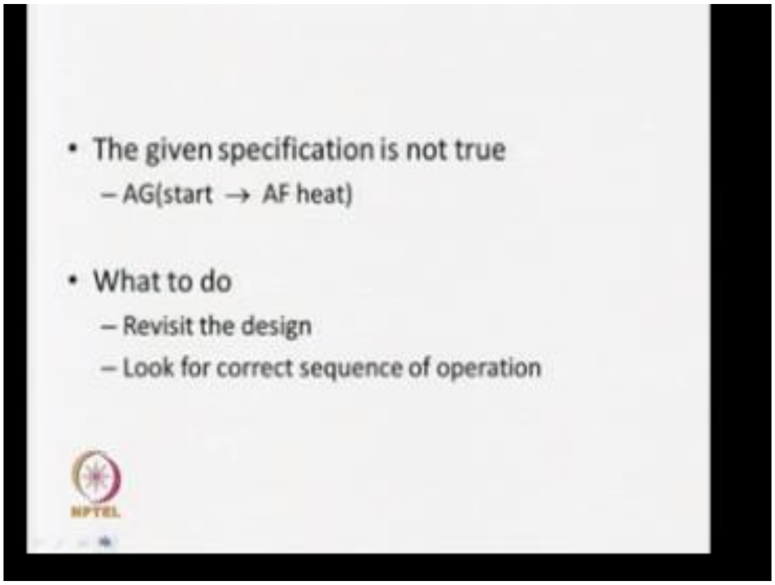
S7={s_4, s_7 , s_6 }

so I am going in once inside the loop, now in C again since there is a sense you have to go again inside this particular loop and we will find that now when I am coming with this particular state S3. Now how many successor it has, it is having successor as S1 and it is having successor in S6 now S1 is not leveled with AC start implies AF heat.

So we are going to remove S3 and so we are going to get S4, S7, S6 as that remaining set now from this particular things now we look again this particular condition and what will happen in S4 now you just see that it is having S4 as is one successor it is having one successor in S3 and another successor in S1 but these AG start implies AF heat is not prove in S3 and S1 so we are going to remove S4 also.

So ultimately we are remaining with S7 and S6 now when we come to this particular state now when you come to S7 it is having one successor S6 and it is not leveled with this particular formula so we will remain S7 from here now when I am coming to S6 then you will find that it is having one successor S7 it is not leveled with AG start implies AF heat so we remove that one also and eventually we are getting an M teaser. That means in none of the state this particular formula is true so that means AG start implies AF heat it is not true in this particular model so now what we are going to do.

(Refer Slide Time: 16:15)

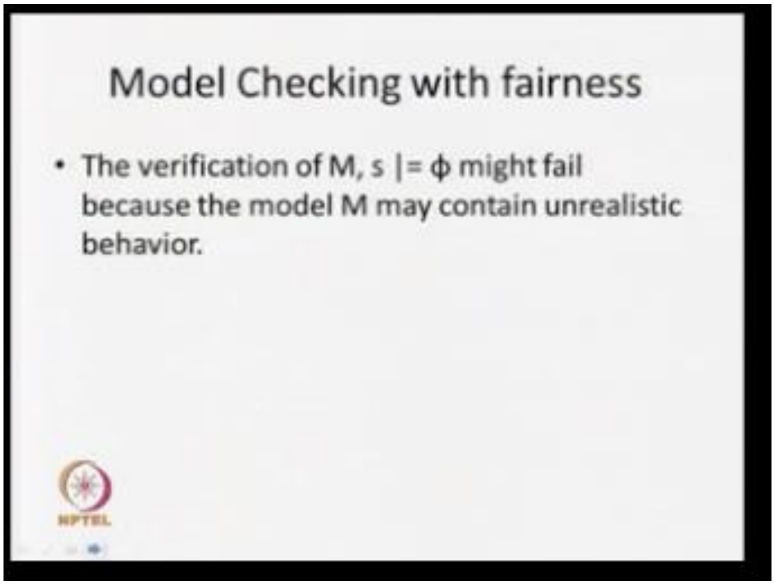


- The given specification is not true
  - AG(start → AF heat)
- What to do
  - Revisit the design
  - Look for correct sequence of operation

Now you say that we have come up with a model and we are looking to check for a particular formula and eventually you will find that after going through our model checking algorithm we found that this particular formula is not true now in this scenario what we have to do, we have to revisit our design and try to modify the design in such a way that the given formula or given property will be true in our model.

But if you look into this particular model after some extent we will find that this model or this design is somewhat correct because the designer will compensate but my model checking algorithm is saying that it is not correct, now either I have to revisit it or whether can I do with something else now we are going to look into that particular issue okay, now either we can revisit our design or you can look for a correct sequence of our operation.

(Refer Slide Time: 17:17)

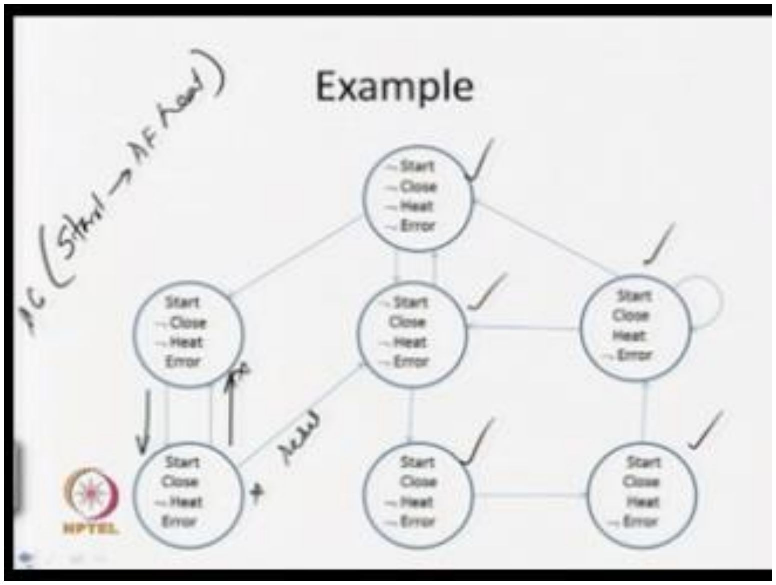


Model Checking with fairness

- The verification of M, s |= ϕ might fail because the model M may contain unrealistic behavior.

It may happen that when we are going to look these thing that when we are going to look for verification of properties in our model it may happen that we are having some unrealistic behaviors due to that unrealistic behavior it may happen that the given properties is not true in our model so somehow what we are going to do, we are going to filter out this particular unrealistic behavior, okay. Now just see this particular example.

(Refer Slide Time: 17:42)



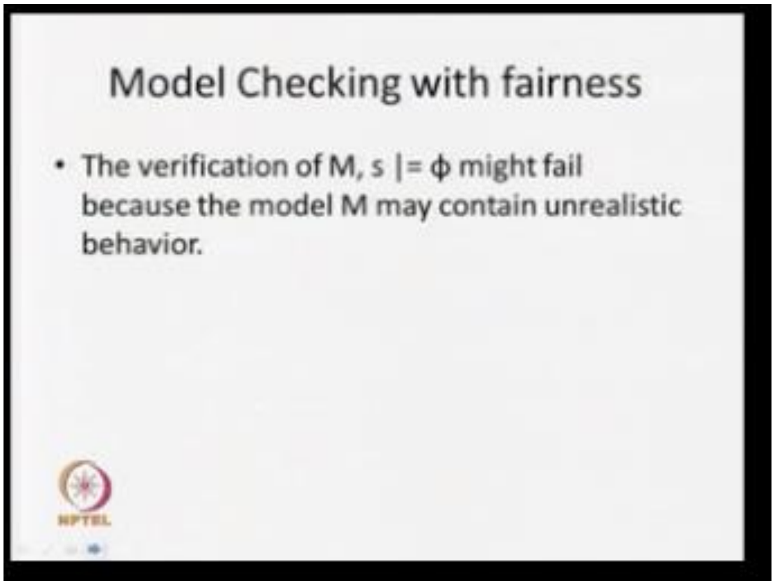
Example

If you come up with this particular model now if you find that if you look for this things we are saying that start implies AF heat okay if we have started a hitter then eventually on part in all part in future it should be hitter up now we have seen that these particular formula is true in this particular first step, okay. So these two in this particular and when but it is not true in this true stage.

So that is why when I am going for loop for AG of this whether is due to this particular states it is not true in our model but what is happening over here you just see that you know that this is your starting then we are closing the doors since it is not going to hitter out again we are opening the door so closing and opening we are doing over here and we will be lifting over here so this is some sort of your unrealistic behavior.

We know that eventually either realizing the mistake and you will reset it brought down and eventually it will come to this particular state one it is coming to this particular state you see that eventually it will go to heat of that call, so this is I am talking about that it may have some content.

(Refer Slide Time: 18:54)

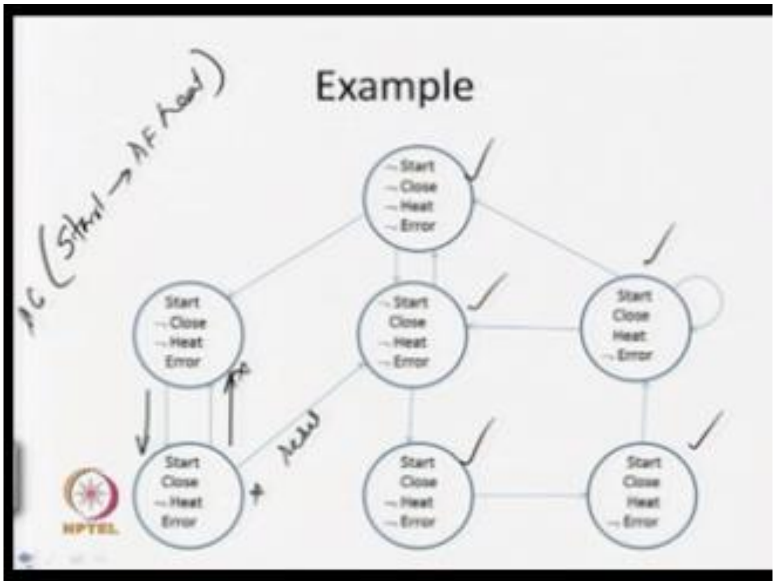


Model Checking with fairness

- The verification of M, s |= ϕ might fail because the model M may contain unrealistic behavior.

And realistic behavior okay.

(Refer Slide Time: 18:59)



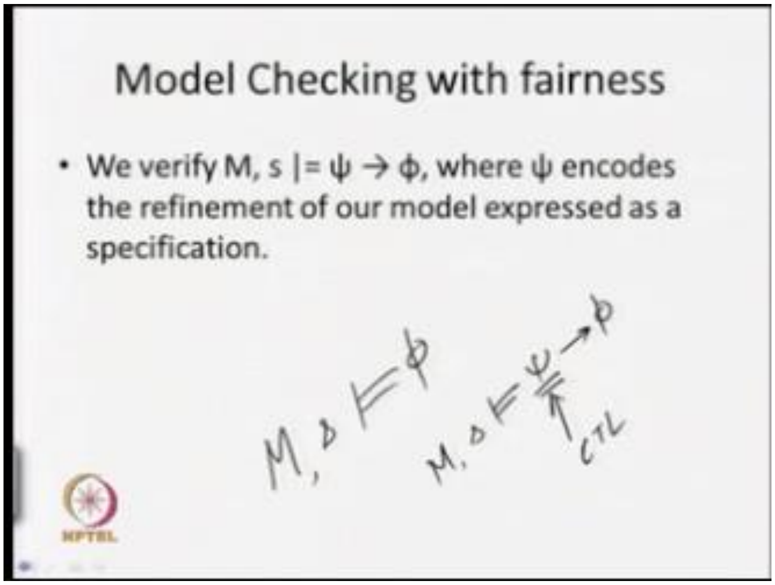
Example

SS this is unrealistic behavior but we cannot avoid it while modeling our system so now what we will see that where a can you go a with this particular unrealistic behavior, so that is why we are going to say that we will put some constant while doing the model checking so that it is going to remove those particular unrealistic behavior, so the constant that we are going to put is basically known as our fairness constants.

That means we are going to look for a fare system we are going to put some farness constraint in this particular with the help of this farness constant what we are going to do, we are going to eliminate such type of behavior that means we are going to loop the model only in the fair part so because eventually we know that you just will realize the mistake and it will phase the reset button and it will come to this particular state.

Once it is coming to this particular state eventually coil will be hidden out so this is the motion that we are talking about the fairness constant so we will put some constant which will filter out the unrealistic behavior of the system and it is going to do the model checking on the fair parts only that is why we are saying it is a model checking with fairness constant, okay.

(Refer Slide Time: 20:13)



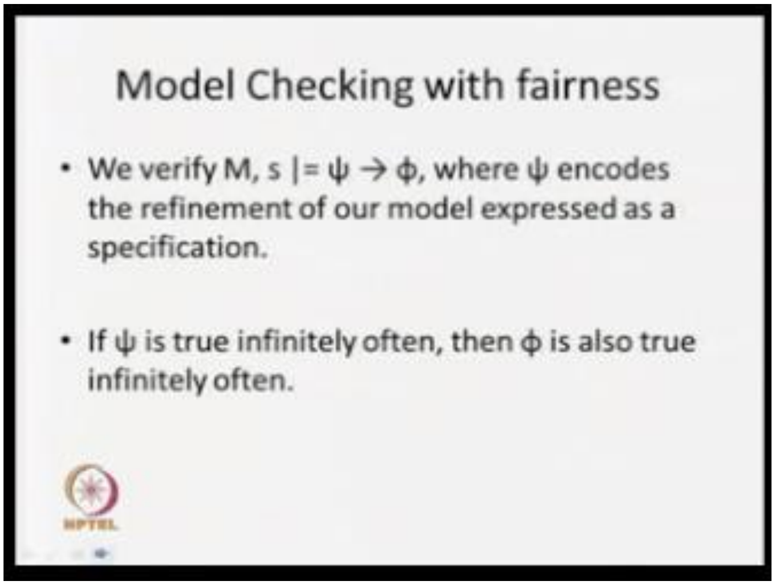
Model Checking with fairness

- We verify M, s |= ψ → ϕ, where ψ encodes the refinement of our model expressed as a specification.

Now how we are going to do this things so we are going to filter out those particular unrealistic behavior so in this particular case how to filter out this particular unrealistic behavior we are going to again do it with the help of our CTL Formula only now what will happen when I am going to model check a particular formula in a particular state S say M s | Ø| in this particular case in set of modeling M s |Ø| who will try to verify M S |Ø| implies Ø so what we are going to do.

We are going to model 6 M S |Ø| → Ø what does it means, where Ø encodes the refinement of our model express as a specification so what is this particular Ø it is nothing but a refinement of our model and we are expressing this particular refinement with the help of another specification so which is another CTL Formula, so we trying to capture these things that refinement of our model with the help of its CTL Formula. Now we will see how we are going to do it.

(Refer Slide Time: 21:26)



Model Checking with fairness

- We verify M, s |= ψ → ϕ, where ψ encodes the refinement of our model expressed as a specification.
- If ψ is true infinitely often, then ϕ is also true infinitely often.

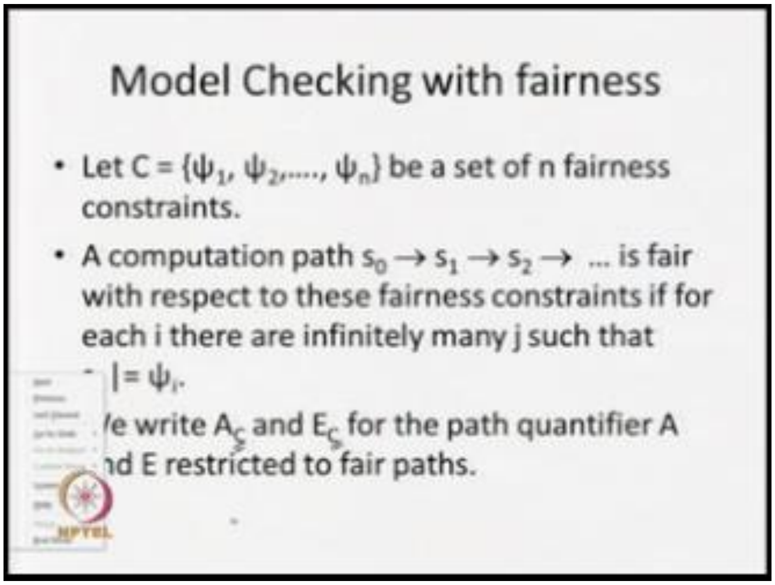
So in this particular case what will happen somehow we have to puts some constants so that it is going to eliminate this particular unrealistic behavior so we get time to capture this particular unrealistic behavior with another CTL Formula sign that means we will say that when ψ is true then Ø is true so that means ψ → Ø we are going to check for the correctness of ψ → Ø that means whenever ψ is true the loop of the correctness of part.

So in this particular case what we can say that if ψ is true infinitely open then Ø is also true infinitely open so that means if ψ is true infinitely open if it is infinity open false then ψ cannot be true bit if ψ is true infinity open then we are going to say that Ø is also true infinitely open so we are capturing this behavior with the help of this particular formula model M ins step S ψ → Ø where ψ is the refinement of our model representing as a CTL Formula.

And Ø is the formula or the property that we are going to check in our model, so we are saying this is the model checking with planner and this ψ is going to basically give us the fairness constraint and if ψ is true infinitely open we say that Ø is also true infinitely open if ψ is not infinitely open to that means it is the going through some unrealistic behavior it will be in some loops.

Where this particular ψ is not true but we know that in our system eventually it is bgoing to come out from this particular loop so that is why we are trying to filter our those particular part and we are going to filter out with the help of this particular constants ψ.

(Refer Slide Time: 23:07)



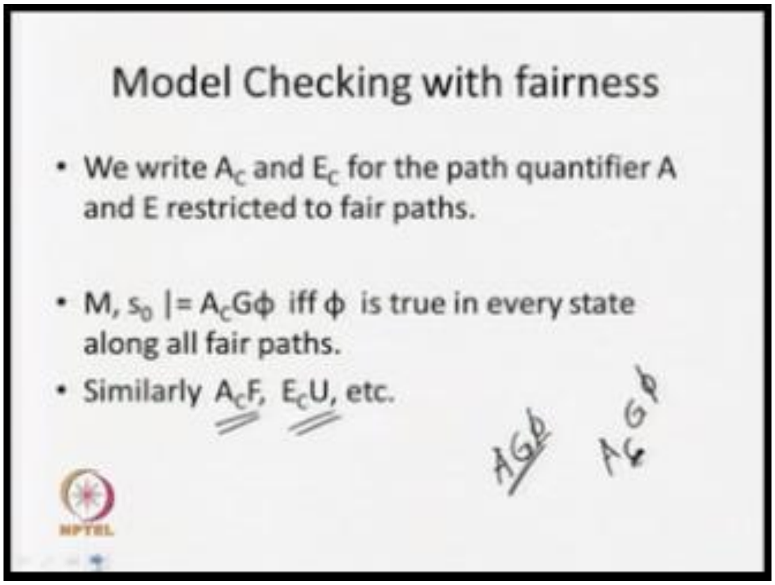
Model Checking with fairness

- Let C = {$\psi_1, \psi_2, \ldots, \psi_n$} be a set of n fairness constraints.
- A computation path $s_0 \rightarrow s_1 \rightarrow s_2 \rightarrow \ldots$ is fair with respect to these fairness constraints if for each i there are infinitely many j such that $|= \psi_i$.
- /e write A_C and E_C for the path quantifier A nd E restricted to fair paths.

Now in general now what we are going to say that instead of open particular farness constant we may have several farness constant so we are going to say that let C = ψ1, ψ2, ψ and be a set of n fairness constraints, so in maps several fairness constants and you say that C is a set of all farness constant so we said that a competition part S0, S1, S2 like that you say fair with respect to this fairness constant.

If for each I they are infinitely many J such that ψj model ψi so that means we are going to look for the part in that particular part those particular ψi will be true infinitely open, so in general situation what will happen you can say that and we writer these things as Ac and Ec for the path quantifier A and E restricted to fair paths so A_C basically says that this is all path through this particular fairness constraints that means all fair paths E_C there exist a path with respect to given fairness constraints C that means we are looking for a particular fair paths, so if we having a set of fairness constraints now we are going to have the path quantifier A_C and E_C instead of ZL A and E.

(Refer Slide Time: 24:32)



Model Checking with fairness

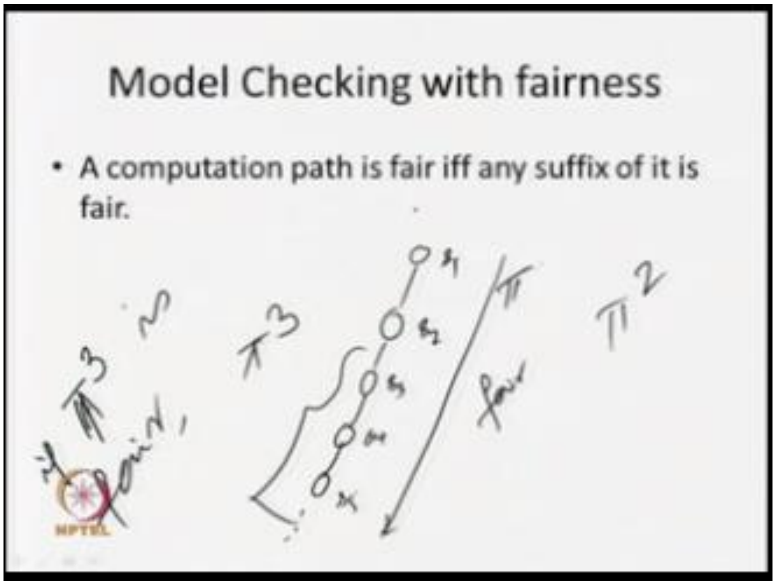
- We write A_C and E_C for the path quantifier A and E restricted to fair paths.
- $M, s_0 \models A_CG\phi$ iff ϕ is true in every state along all fair paths.
- Similarly A_CF, E_CU, etc.

$AG\phi$
$A_CG\phi$

Okay, now we write A_C and E_C for the path quantifier A and E rest that to the fair paths now in this particular case now what will happen we can say that we are having the formula $AG\phi$ so that means where the in all path globally ϕ holds or not so these particular formula if we write $A_CG\phi$ what does it means what are these ϕ holds globally in all fair paths that means in all paths where this particular fairness constraints are satisfied, okay.

So that means we are going to look for all fair paths that means we fairness constraints we may have such type of scenario M s0 model $A_CG\phi$ if ϕ is true in every state along all fair paths, so similarly we can define for A_CF and E_CU extra that means AGp is all path globally ϕ holds without any fairness constraints but $A_CG\phi$ says that ϕ holds globally in all fair paths that means in all the paths whether the fairness constraints 2C is true, okay. So this is the way that we are going to look into it.

(Refer Slide Time: 25:49)



Model Checking with fairness

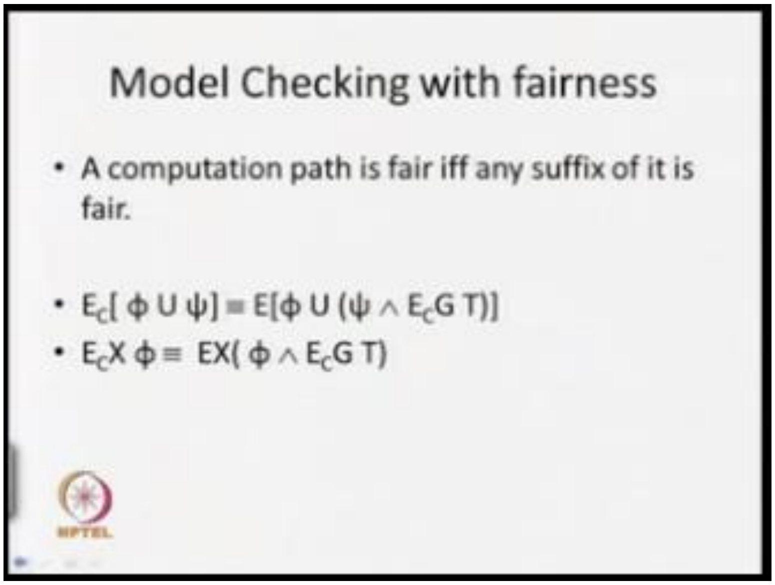
- A computation path is fair iff any suffix of it is fair.

s1
s2
s3
s4
s5
π
π2
π3
fair

Now we can say that a computation path is fair if any suffix of it is fair, so we can say that now you say what happen if I am going to say that this is a path say I am coming from s1,s2,s3,s4,s5 something like that I can say that these path is fair enough if any one of the suffix is fair that means if I can establish that this is the suffix of the given path so this is your path π then π3 is the suffix of this particular path starting at this particular state s3.

Now if π3 is fair then we can say that π2 will also be fair because since it is suffix is pair so we are going to say that the computation path is fair if any suffix of it is pair, so if we can establish any suffix fair then we can look into that particular path will also fair because I tell eventually it will go to the particular suffix. So in this particular case now what we can say that we can look for.

(Refer Slide Time: 27:05)



Model Checking with fairness

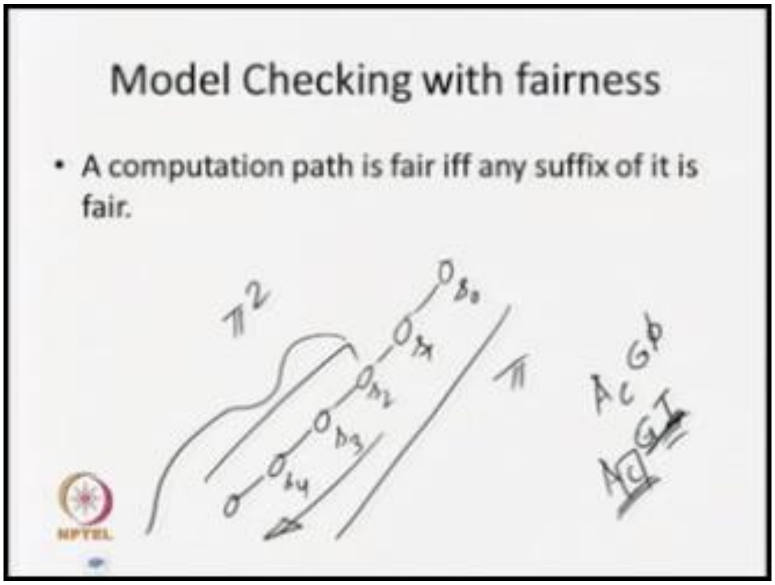
- A computation path is fair iff any suffix of it is fair.
- $E_C[\phi \, U \, \psi] \equiv E[\phi \, U \, (\psi \wedge E_C G \, T)]$
- $E_C X \, \phi \equiv EX(\phi \wedge E_C G \, T)$

Your model checking formulas similar to something like that so if I am going to say that.

(Refer Slide Time: 27:13)

Any suffix of a path if it is your fair suffix that means we can say that what I can say in this particular case so what we are saying that a computation path is fair if any suffix of it is fair.

(Refer Slide Time: 27:31)



Model Checking with fairness

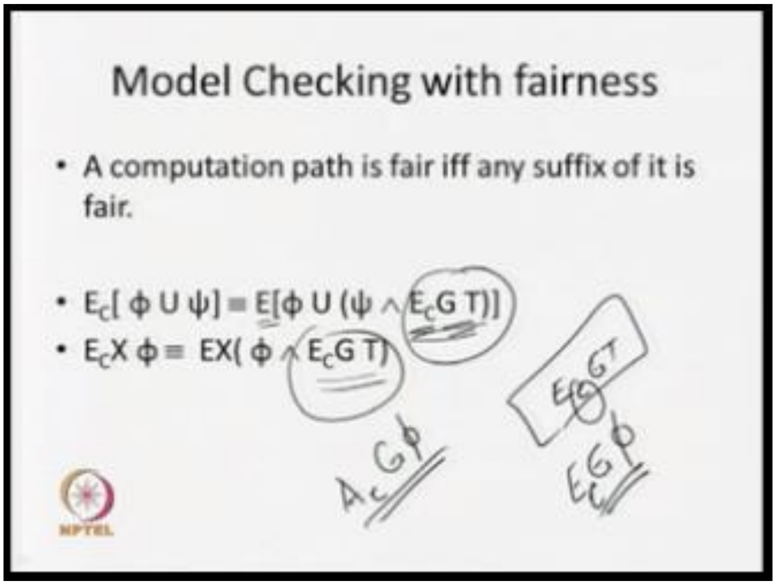
- A computation path is fair iff any suffix of it is fair.

So basically if you consider any path something like that s0,s1,s2,s3,s4 like that so if this is path 5 and if you consider any of its suffix say this is your π2 if we say that π2 is fair then we can eventually say that π1 will be fair and we can say that π0 or π with it because eventually it is going to this particular fair path, so that is why you can say that if you know formula what we are looking into it.

$A_C G\phi$ okay, so we are saying that $A_C G\phi$, now if I say that $A_C G\phi$ now if I say that $A_C GT$ so what does it means, that means in all state we noted in all state it is labeled with your truth value true so that means if we say that GT basically talk about a path where true is always true, but along with that instead of A we are saying that A_C that means it is the path where it is particular fairness constraints is true that means eventually we can write that $A_C GT$ so this is basically talk about this particular suffix where this suffix everywhere true is true.

But it is your A_C means that means the fairness constrainers C is also true over here, that means for those things we can say that if a suffix is fair then this path will also be here, so in that particular case what.

(Refer Slide Time: 29:14)



Model Checking with fairness

- A computation path is fair iff any suffix of it is fair.
- $E_C[\phi \text{ U } \psi] \equiv E[\phi \text{ U } (\psi \wedge E_C G\, T)]$
- $E_C X \phi \equiv EX(\phi \wedge E_C G\, T)$

$E_C G T$
$E_C G \phi$
$A_C G \phi$

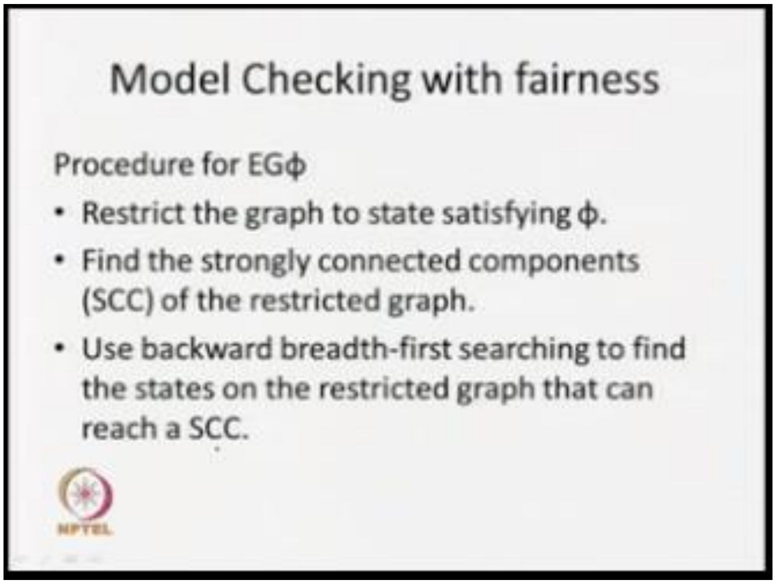
We can say now so we are going to look for say $E_C\phi$ until ψ so this is basically there are exist a path where this particular fairness constraints is true ϕ until ψ that means we are going look for this ϕ until ψ in fair path. So we can say that this is equivalent to their exist a path ϕ until ψ and E_CGT so this is your fair path where true is always true, that means we can say that eventually when I am talking about that E_C that means their exist a ϕ with fairness constraints ϕ until ψ we are talking using the general E until operator without any constraints so we are going to look for ϕ until ψ.

But along with this particular fair path, E_CGT so similarly for $E_CX\phi$ what we can write at $EX\phi$ and E_CGT that means we are going to look for the next step where it is a fair enough so E_CGT basically we will talk about the fair path that means the path where this particular fairness constraints C are true. So you just see that if it is your E_C until operator and E_CX that means until their exist the path with fairness constraint until and their exist a path with fairness constraints next step these two are can be represent that with the help of E_CGT.

That means eventually we will find out we need a procedure for either EGT, E_CGT or A_CGT these are very similar. So basically what happens we need the procedure for either $E_CG\phi$ or

$A_CG\phi$ that means once we have a procedure for $E_CG\phi$ then we can talk about E_CGT or $A_CG\phi$ we can talk about A_CGT so we need procedure for this one and even after that what will happen E until and EX can be expressed with the help of E_CGT so basically we need procedure for either $A_CG\phi$ or $ECG\phi$, if we have the procedure then we can have the model checking procedure for through with fairness constraints.

(Refer Slide Time: 31:59)



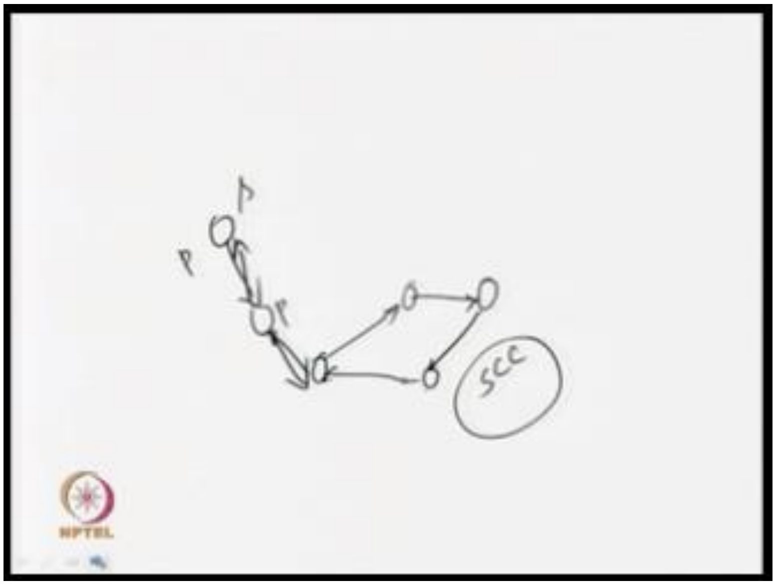
Model Checking with fairness

Procedure for $EG\phi$

- Restrict the graph to state satisfying ϕ.
- Find the strongly connected components (SCC) of the restricted graph.
- Use backward breadth-first searching to find the states on the restricted graph that can reach a SCC.

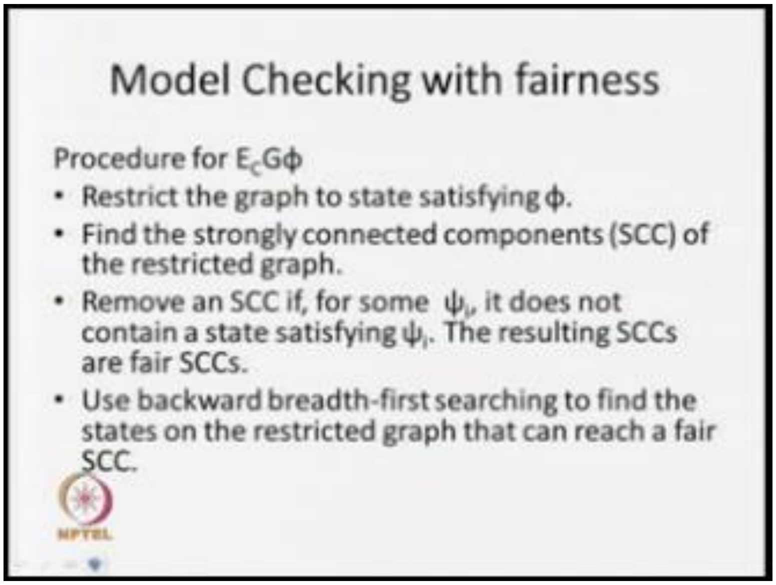
Okay, now just you what is the procedure for $EG\phi$ already we have said it we have another procedure we are saying that restricted graph to start state satisfying ϕ, so first state is we are restrict a graph to a state where it satisfies ϕ the find the strongly connected components SCC of the restricted graph and as I just start step we are saying that use ba$_c$kward breadth first searching to find the states on the restricted graph that can rea$_c$h a SCC, so with the help of this procedure we can talk about $EG\phi$, okay.

(Refer Slide Time: 32:34)



So basically what happens we can say that if I am having some state so we can say that this is your SCC because all state that can be reach from any other state. Now from this particular state what will happen we will follow the backward sorry, my role being this direction backward breadth first search and if I get any state then we can say that this is basically we can reach over here because if we are looking for p, p is true over here because it will be p is true will also be over here so in this particular way I am going to collect all the states.

(Refer Slide Time: 33:19)



Model Checking with fairness

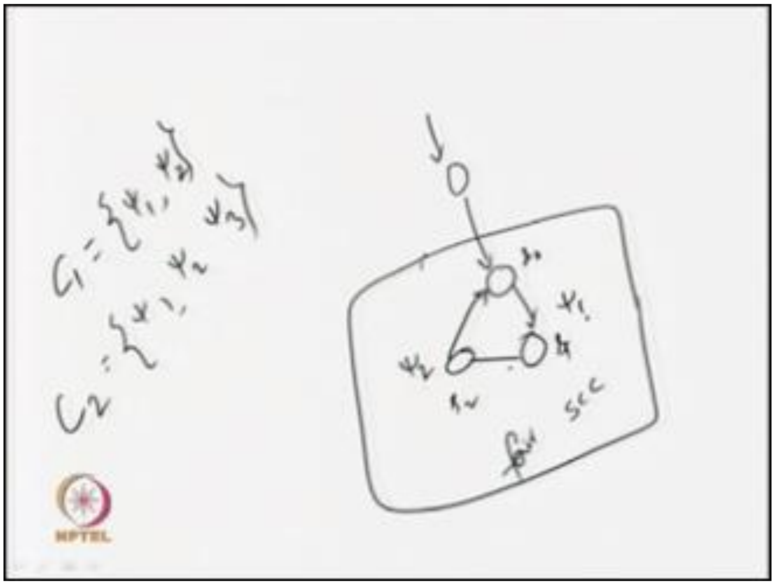
Procedure for $E_C G\phi$

- Restrict the graph to state satisfying ϕ.
- Find the strongly connected components (SCC) of the restricted graph.
- Remove an SCC if, for some ψ_i, it does not contain a state satisfying ψ_i. The resulting SCCs are fair SCCs.
- Use backward breadth-first searching to find the states on the restricted graph that can reach a fair SCC.

Now in fair what will happen now we are going to look for $E_C G\phi$ here exist a path with fairness constraints $G\phi$ is true, now this is also similar to that procedure only what we are going to say that first state restrict the graph to state satisfying ϕ, find the strongly connected component SCC of the restricted graph already we have discuss. Now we are having one particular condition over here remove an SCC if for some ψi it does not contain a state satisfying ψi. The resulting SCCs are fair SCCs.

Now we will just give you an example so we are saying that infinitely of all ψ is must be true, so if it is not true then we are going to remove those particular SCCs. Again the fourth step is similar use backward breadth first searching to find the states on the restricted graph that can rea$_c$h a fair SCC, okay.

(Refer Slide Time: 34:20)

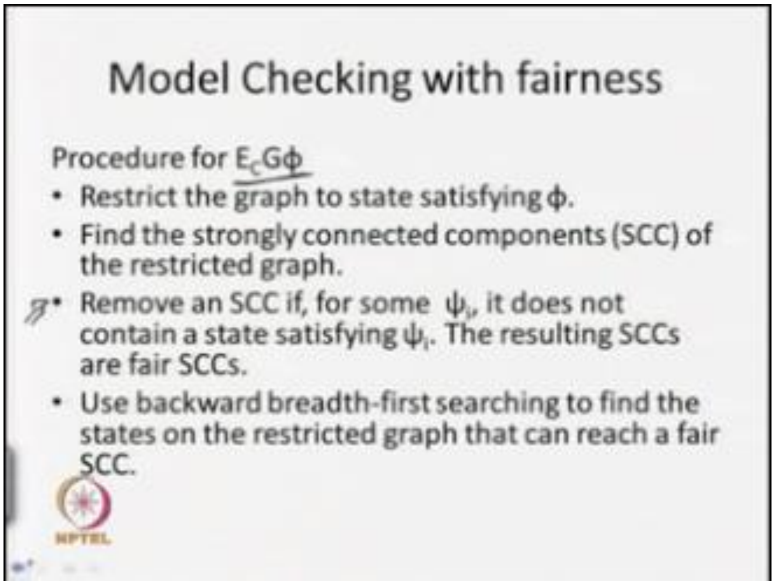


Now you just see that, now I am giving an example say so I know that these are the state s0, s1 and s2 so this is an SCC we do not have any problem now you just see that I am having a fairness constraint C=ψ1 and ψ2 okay, now what will happen say in this particular state s1 says ψ1 is true and in this particular state says ψ2 is true. Now in this particular okay whenever I am coming to say from some other state if I am coming to this particular state then what will happen you just see that it will be in this particular loop, it can go over.

So infinitely it can go over here but wherever it go some point of times ψ1 will be true or ψ2 will be true, okay so I can say that this is a fair SCC, okay. But if I talk about a fairness constraints say this is C1 and C2 and I am talking say that ψ1, ψ2 and ψ3 are fairness constraint, now when I come to this particular SCC okay, in this SCC wherever I go I will find that ψ1 will be true whenever I go I will find that ψ2 will be true so in none of the state in this particular SCC strongly connected components ψ3 is true.

So that means if I remain in this particular loop then ψ3 is not going to be true, so that means this plan is constant is not satisfied over here, so I am not going to say that this is the pair as is so as for the third step of my algorithm we are going to remove this particular SCC formula graph so this is the thing that we are talking about.

(Refer Slide Time: 36:14)



Model Checking with fairness

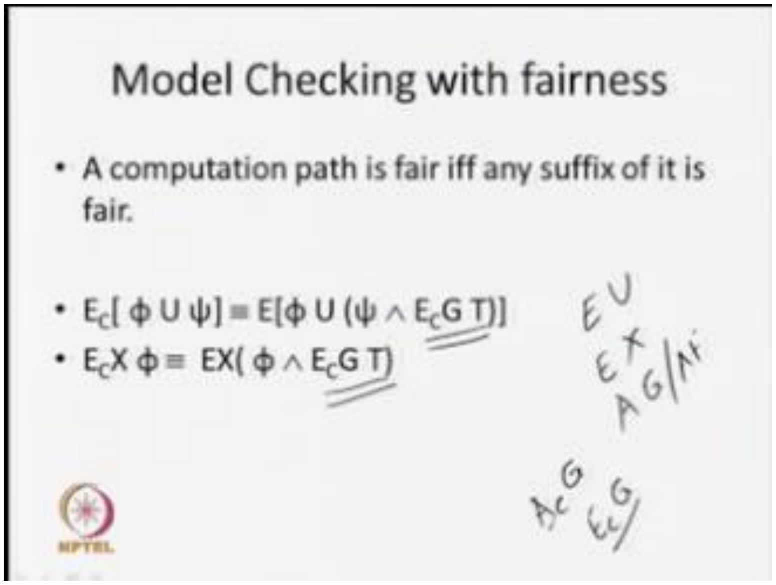
Procedure for $E_C G\phi$

- Restrict the graph to state satisfying ϕ.
- Find the strongly connected components (SCC) of the restricted graph.
- Remove an SCC if, for some ψ_i, it does not contain a state satisfying ψ_i. The resulting SCCs are fair SCCs.
- Use backward breadth-first searching to find the states on the restricted graph that can reach a fair SCC.

That remove an SCC if some φ I it does not continues state satisfying φ I the resulting SCC are the fair SCCs. So that means we just keeping on with the fair SCCs, so if this fair SCCs that means whatever fairness constant we are giving those will be true over here so this is model; checking algorithm which fairness constant. You just see that if once we have this procedure EG or EG then other operators can be defined with the help of the AG or EG.

So essentially we have seen that we have got a procedure for model checking with fairness and we have discuss here with EG G φ so this is similar to E φ accept that we are having one particular step over her where we are going to remove the unfair SCCs we are going to keep the fair SSCs now you just see that what are the procedure that we will needing.

(Refer Slide Time: 37:15)



Model Checking with fairness

- A computation path is fair iff any suffix of it is fair.

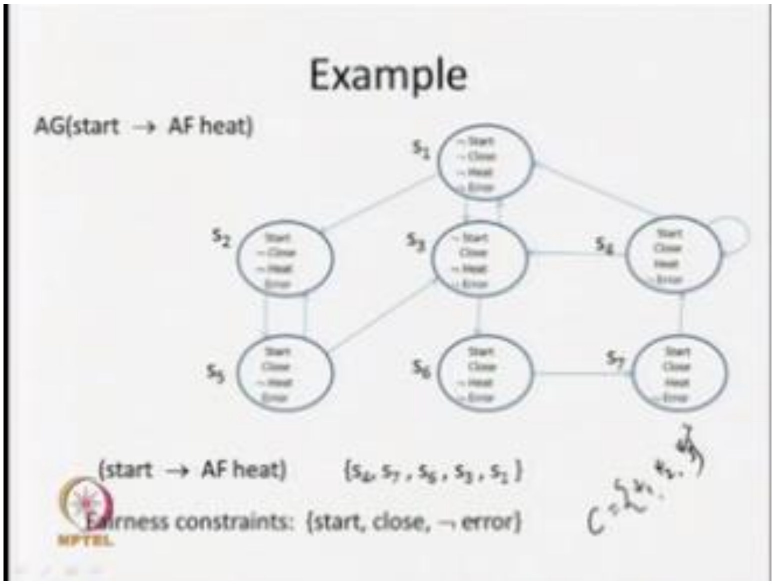
- $E_C[\phi\ U\ \psi] \equiv E[\phi\ U\ (\psi \wedge \underline{\underline{E_C G\ T}})]$
- $E_C X\ \phi \equiv EX(\phi \wedge \underline{\underline{E_C G\ T}})$

E U
E X
A G / A F
$A_c G$
$E_c G$ /

For your model checking algorithm with fairness set we need either one E ᴗ EX or say Ag or say AF now we have seen that with only ACG or ECG if we know the procedure for ACG or ECG then we can look for the procedure for E1 Ex because EC is again express with the help of the normal lee operator and ECX again express with the normal Ex operator but along with that we are having this particular fair stuff.

So Ec G T and EcG that means whether fair is constant of this basically we are going to keep all the fair SCCs okay this is the notion so we are having now we are seen the procedure for EcG or similarly we can contract for EcG also.

(Refer Slide Time: 38:17)



Example

AG(start → AF heat)

(start → AF heat) $\{s_4, s_7, s_6, s_3, s_1\}$

Fairness constraints: {start, close, ¬ error}

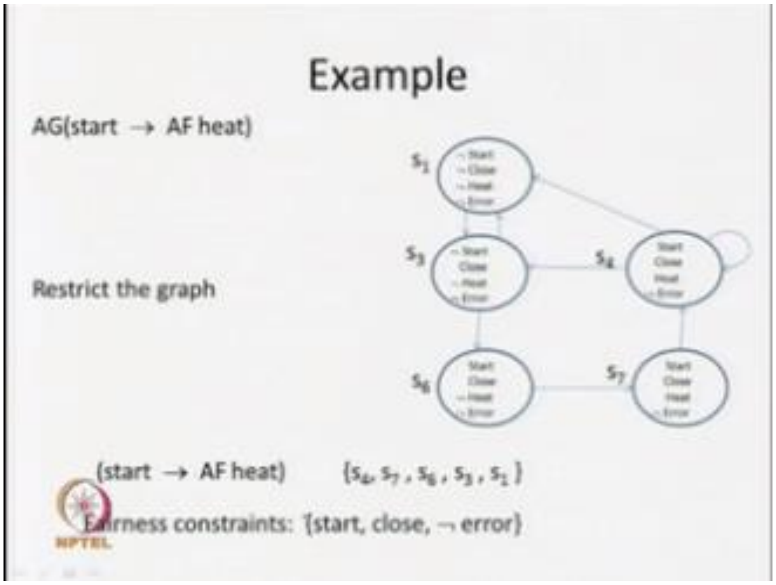
Now after knowing this particular fairness concern let us come back to our example that we are talking aborted micro woven examples and we come up with this particular model which is having seven states and we try to check for this particular formula AG start in plus AFE, now once we apply a normal modeling algorithm what you found that these formula is not satisfying this particular model.

Now one option is we should go for defining of this particular model come up with your answer the solution or secondly we have seen that can go here or remove or even those particular unrealistic behavior and try to check this particular formula in realistic behavior on it and for that we have to apply fairness constant, now we have seen that this is some unrealistic behavior that closing and opening the door repeatedly after starting it so we are try to remove this particular un realistic behavior.

So how we are going to do this thing that we going to the hit with the help of some fairness constant now what will be the fairness consent over here so what is the fairness constant that we are going to look for it start close and not of error that means when it go to start and close situation that it should go in to the error conditions, so not of error so this is basically we are talking about the c is the set of fairness constant where e I am having φ 1 φ2 φ3 as my fairness constant.

So here my fairness constant has start close and not of error and we are going to look for AG start In plus AF already we have seen the marking of start in plus AF it and we know that these are the steps where start in AF if it is true S4 S7 S6 S3 and S1 now along with that we are giving this particular fairness constants start close and not of error. Now will see hoped at model checking with fairness constant will be use to check for this particular formula.

(Refer Slide Time: 40:28)



Example

AG(start → AF heat)

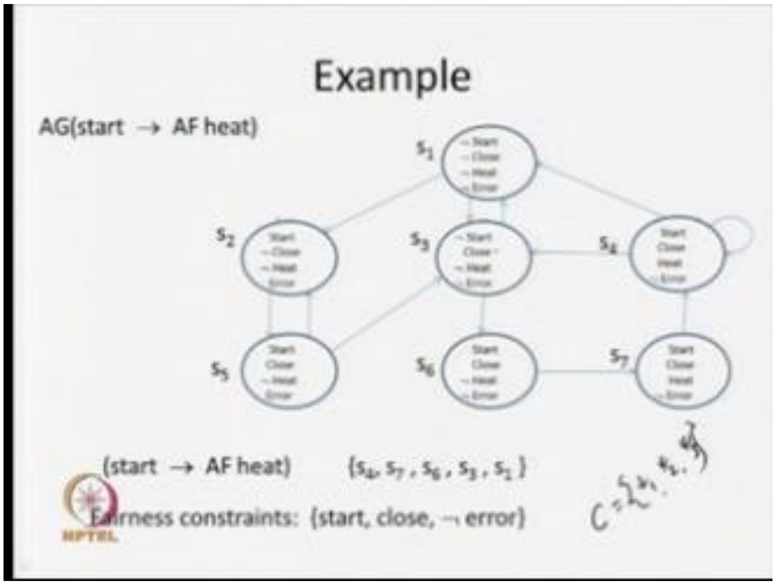
Restrict the graph

(start → AF heat) $\{s_4, s_7, s_6, s_3, s_1\}$

Fairness constraints: {start, close, ¬ error}

Now what will happen in this particular case, now as for our this thing we are going to restrict the graph with the state where this particular formulas start in plus AF it is true so basically we will find that these are the states where this particular formula AF start in plus AF it is true because ion the order to state S2 and S5 this formula is not true now after it follow one we have to loom for those particular fairness constant.

(Refer Slide Time: 40:59)



Example

AG(start → AF heat)

(start → AF heat) $\{s_4, s_7, s_6, s_3, s_1\}$

Fairness constraints: {start, close, ¬ error}

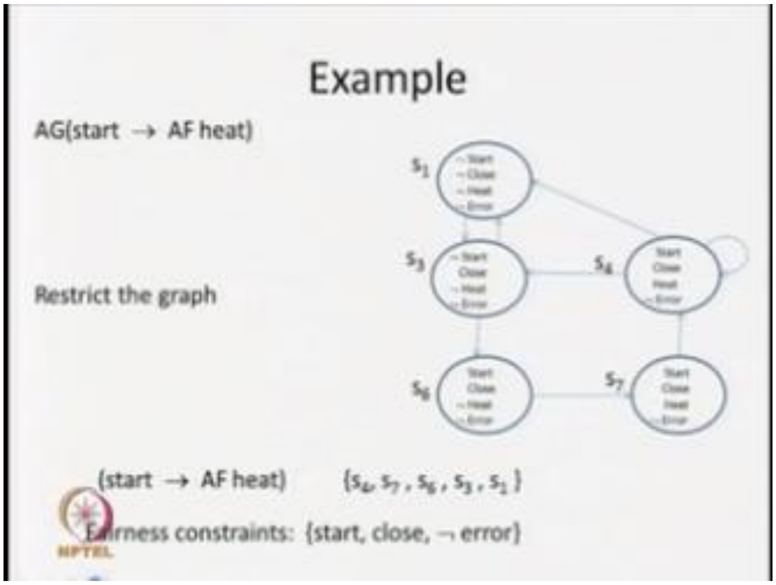
Now as for our example now will see that what are the SCCs over here once we will get the SCC then you will see whether ion this particular SCCs start close and not of errors are true or not you just see that wherever it go you can look for this particular series is I am having this is the loop so wherever you go you can go to any other state so in this particular case what will happen you will find that idea start is coming true close is coming true or not of error is coming true.

So infinity often it is come true on the other hand if you can said that I can be in this particular loop so one of this particular fairness constant will give you so like that we can say that this is the remaining as we say that we are going to have and if you now apply this particular AG start AFE then you will find that in all those particular state where this particular formula is true that means globally in all part globally start in plus EFE that means whenever you start the oven eventually it will heat of the coil.

Because her what happens basically with the help of this fairness constant we are removing this particular unfair point okay so when you are going to check on way on this particular fair parts now you just see that with fairness constant eventually you say that okay our model is going to worked and it is giving to satisfying g this particular formula. But in some point of time maybe user is going to be in this particular loop for ever then this not but we know that.

Usually also use is own intelligent and eventually you will phrases reset button and you will come to this particular point so this is some sort of you unfairness thing here and with constant fairness constant we are removing it and we are try to check on the property in the remaining graph okay, so this is the way that we can check for all this connector one is your re find your model or secondly try to find out a constant fairness constant and model check on the pair parts only.

(Refer Slide Time: 43:00)



Example

AG(start → AF heat)

Restrict the graph

(start → AF heat) $\{s_4, s_7, s_6, s_3, s_1\}$

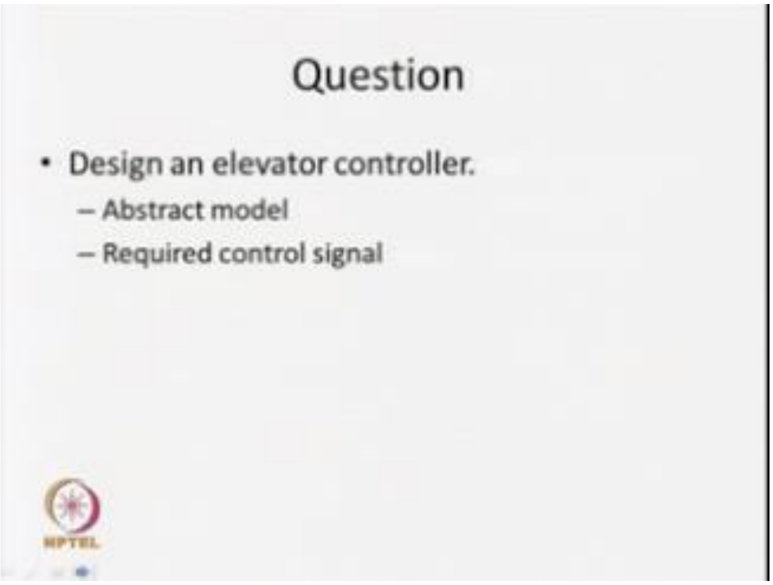
Fairness constraints: {start, close, ¬ error}

So this is basically we have discuss about the model check algorithm we need one model we need a properties those properties will be express in the CTL, so this tool give as a input top the model checker and it will mode checker will give the set of states where this particular formula is true and some time what will happen if we can identify that it is having some unrealistic behavior but it is unbendable then we will try to identify some fairness constant and we are going to look for the correctness of the properties in fair parts on this.

Pair parts will be those part where those particular fairness constants are true and with this you can now proceed for a model second we are going to check whether this properties is true in this particular pair parts or not okay so this is in general or we can say that this is the trance or this is the concept about the model checking in any design we can apply this particular model checking algorithm before proceed further inn our design cycle okay.

So what is this some come up with your design express your properties in CTL and try to check those properties in those models with the help of model checking in algorithm but if the system is having some unrealistic behavior then we try to filter out those particular unrealistic behavior with fairness constant and we will do the model checking on pair parts on this, so this is about the model checking approach for verification of about properties.

(Refer Slide Time: 44:37)



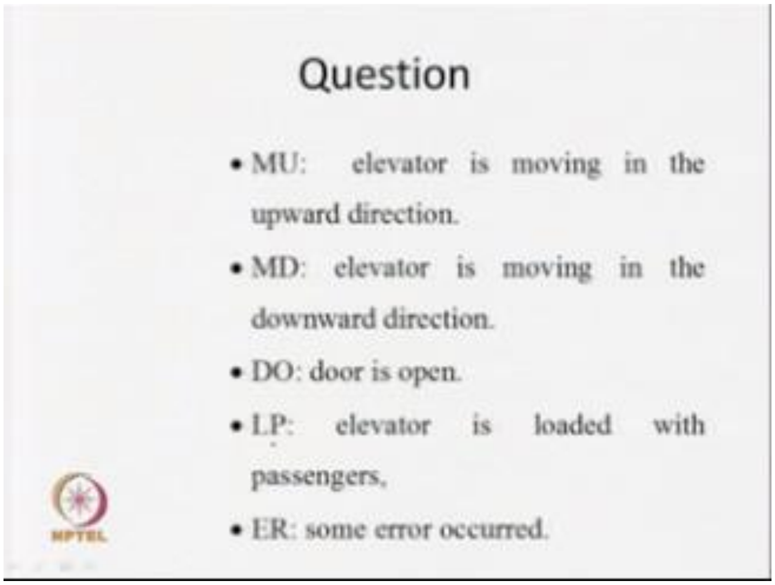
Question

- Design an elevator controller.
  - Abstract model
  - Required control signal

Now we are having now enough information about your model checking algorithm and we know how to do it now just look for some questions now that one question that I am giving you now design an elevator controller I think in a multi store it building all of you have used elevator and all having that you can press button to go to any floor or if you give a request to the lift by pressing as button in this particular floor so further we have to design this controller so we have to identify what are the things or what are the control signal required and depending on that we are going to design a controller.

Now in this particular case again as for all design principle we are going to abstract of the model and we are going to identify the require control signal and with the help of those control signal we are going to have a model once we have the model then we are going to loom for the properties that need to be satisfy by our controller and we are check those particular properties.

(Refer Slide Time: 45:44)



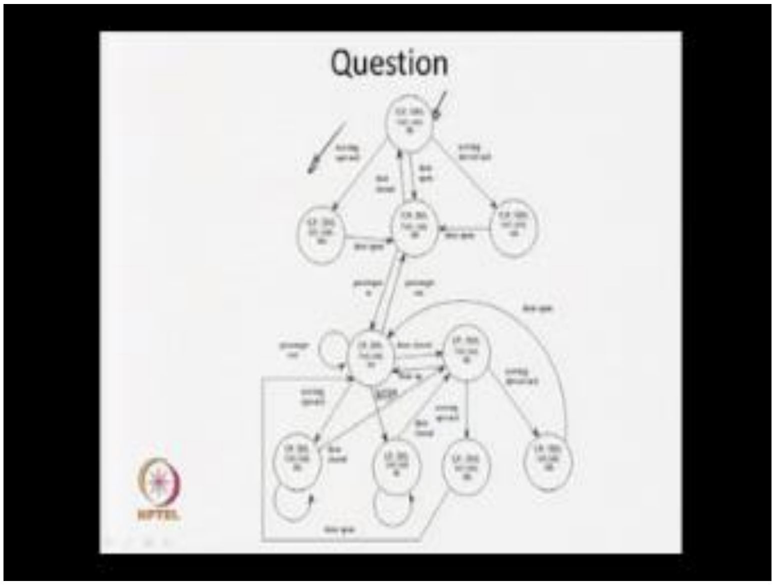
Question

- MU: elevator is moving in the upward direction.
- MD: elevator is moving in the downward direction.
- DO: door is open.
- LP: elevator is loaded with passengers.
- ER: some error occurred.

So in this particular case I am coming up with the very simple solution or I am just abstracting of the controller in such a whether we are having a very less number of control signals so this is the initial design after this you can go for defined the nun put and put more and more information in to our model so in this particular case I am coming up with four signal when I am talking about MU that means elevated is moving in the upward direction MB is a another event I am going to said that elevator is moving in the downward direction so these are the signal indicate in the either the lift is moving upward or if it is moving down ward.

DO basically door is open the door can be either in two condition it is open of floor so I can identify one signal is door open LP elevator is loaded with passengers and ER it is call to some error condition some error id occurred so we have identify this particular five control signals so these are the atomic proposition either this will be true or false now, if with the help of this thing we can come up with the model okay. So say this is simpler model I am saying that.

(Refer Slide Time: 46:55)



Question

This is one particular step what we are saying that this is not of LP not of DO not of MU not of MB and not of error that means it is the ideal scenario now depending on this when the moving upward or moving upward what will happen the signals. So this is a simplified model I am saying that this is one particular step what we are saying that is not of LP not of BO not of MU not of MV and not of LV that means the ideal senior now depending on this when the moving upward or moving upward what will happen the signals some of them will move upward.

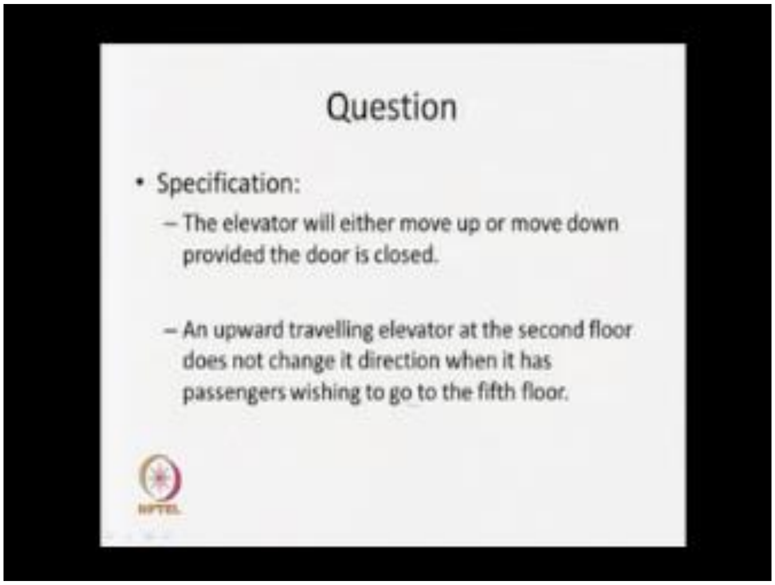
This is moving downward and depending on these things door opened and door closed these are the events depending on the function in arrears in defining steps and is all the states are leveled with the controversy signal which are true in here check. Similarly the presents are keen and the presence are these are the events depending on that you can go to some other step and after that we can close the door on close the door it will move upward or move downward.

And without closing the door if again say time to move on up and move down and will go to the errors in the scenario. So these are the error scenario by which looking into and we are going for even that is going to happen in while operating this particular elevator come up with this particular simplified things.

Once we have this particular model now we can look for the properties we need to be satisfied with this particular model and we can apply this particular model signal algorithm to check for the correctness over this design. If the design is correct then we can pose it for the for verification and other thing if it is not correct then what will happen the model algorithm models give algorithm give a contra. For example and if you say that you follow a particular part or an particular execution test whether it is true or not and as a design what happens.

We redesign or we can rectify those particular error these are all on the other hand, if as a designer if you convenes that this particular behavior is an unrealistic one then I can try to identify some fairness constant and I will refer the model checking we use particular fairness constant. So this is the we are going to proceed now once we have a model then what we can say that.

(Refer Slide Time: 49:05)



Question

- Specification:
  - The elevator will either move up or move down provided the door is closed.
  - An upward travelling elevator at the second floor does not change it direction when it has passengers wishing to go to the fifth floor.

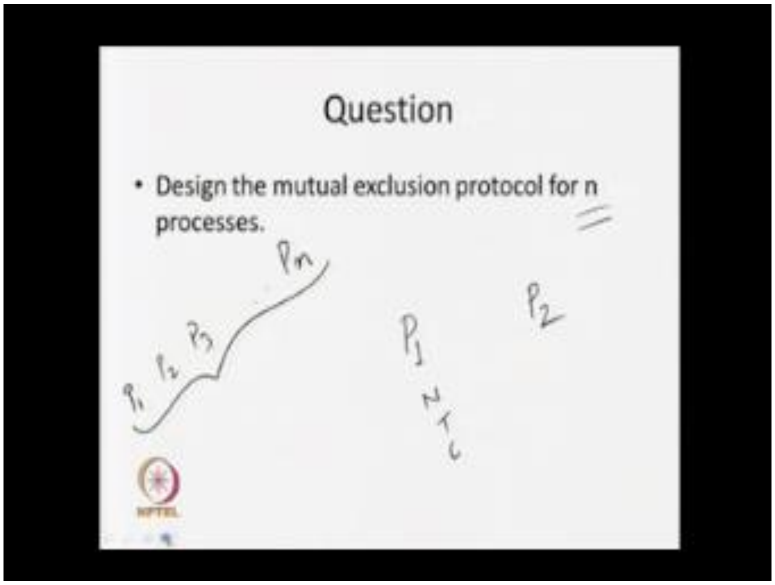
What may be the specificity or what are the properties so one simple algorithm I am saying that the elevator will move out or move down provided that the door is closed that means the elevator should not have any movement if the door is opened. So this properties must be satisfied by or controvert. So another property I can say that an elevator traveling say an upward travelling elevator at the second floor it does not change its direction when it has passengers wishing to go to the fifth floor. So this is basically what are the algorithm that you have in cooperated in our lift control because it should have some protocol to leave the service.

So generally what happens the simple procedure a simple algorithm simple if lift is moving upward then it is going to service the request for all upward direction why because as a designer we know that if it is moving up some floor we are going to come down then what will happen they will sent in the direction of motor is going to power energy. So in generally what happens we are going to leave the surface to all upward directions because if the lift is moving in the upward direction.

Then it sense the direction then lift is coming in the downward direction so that is all type of properties which try to satisfy to check for the correctness of the operators. But if you are in some particular then these property may not be true. So depending upon the protocol data we have implemented or if we are using all the controller we have to identify the property what is that and we have to check those property.

So this specification now what happens you see that we are having a model we come up with a model. We have those particular specifications now we have to represent those specifications on temper logic formula now we are having enough knowledge about a temper logic about CTl, now this is take these parts and try to write down the CTL formula for these particular specification okay. Now another question you see that,

(Refer Slide Time: 51:25)



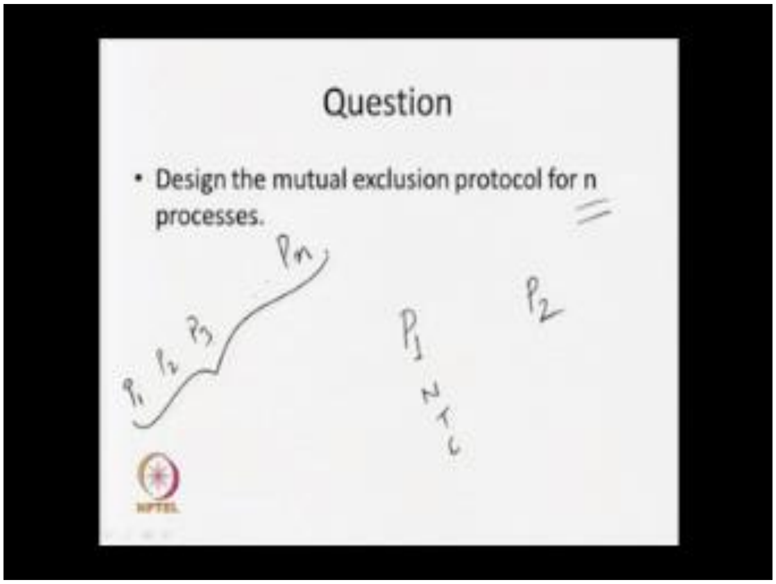
Question

- Design the mutual exclusion protocol for n processes.

What happens already we have discussed about the visual expression protocol okay. So in this particular visual protocol we have we come up with a model and we have seen that or we have discussed that it is having two process so P1 and P2. So we are working with two processes P1 and P2 and whatever steps we vary seeing that it may be in non critical reason it may be time to enter into critical reason. So we have model 8 and we have seen what are the properties you need to satisfy.

Now I am giving a question to you people now you itself check these are the visual expression protocol for n processors, now we have discussed the visual expression protocol for two process now I am saying that you extend this model for n process okay. Now what will happen now we are having two process P1 and P2. And we know the step and eventually what we are doing we say that come up with a more common model and do your job. Now instead of two process now I am doing three process P1, P2, P3 like that Pn. Now come up with a model and see how to model this one hoe to model those particular properties.

(Refer Slide Time: 52:44)



Question

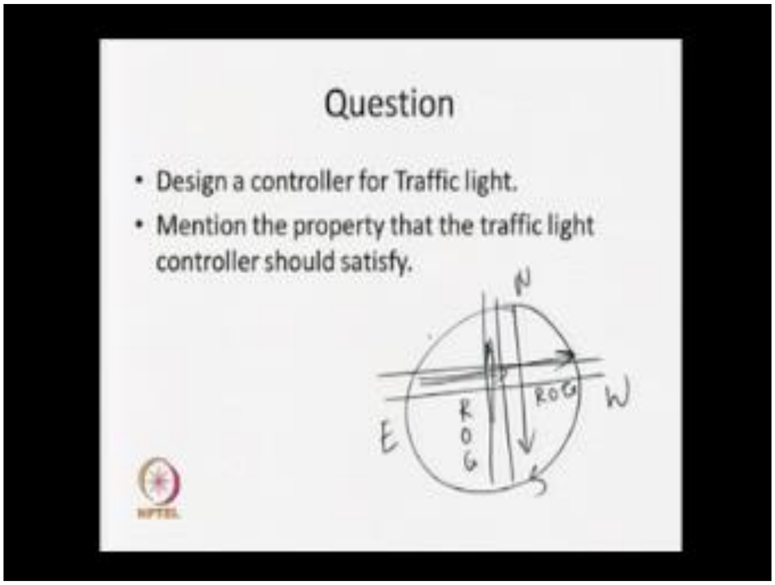
- Design the mutual exclusion protocol for n processes.

Pn
P1
P2
P3
P1
P2
N
T
C

So similarly I can give you some more questions you try to come up with a model and try to look for a specification for the properties needed to be satisfied. So one simple question I can say that design is a couple of traffic values so we are having a traffic light controller. Now we should try to or try into design this particular controller so that we can control that in that junction. And now try to come up with a specification of this particular traffic light controller and represent those particular speciation in your CTL.

And after the CTL formula algorithm to check those particular properties now you itself see that it give me a hint. So in a traffic light controller then what6 we will do what will happen, so I am having a junction road junction. So you can say that it can go east, west direction or north, south direction. Now what will happen here I can have 360 degree signal there, or else I am giving okay. So this is for east, west direction and similarly I can have where Ro n screens for a east, west directions and say north, south direction.

(Refer Slide Time: 53:04)



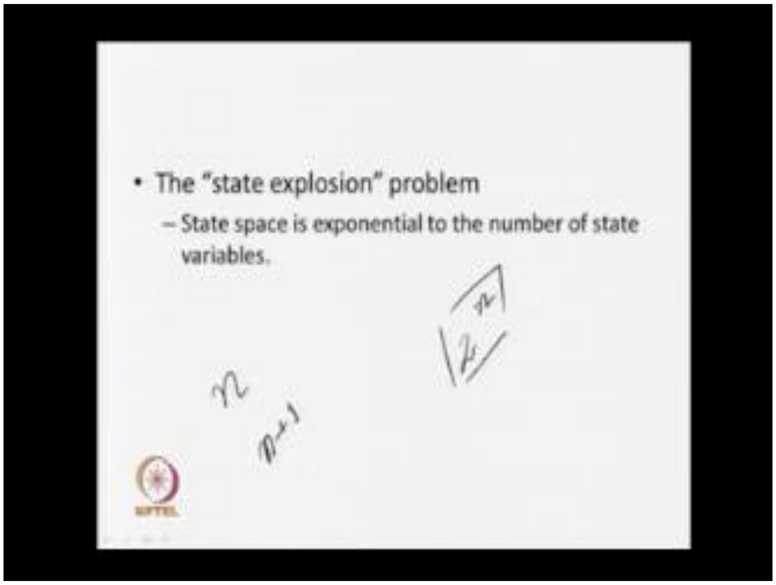
Question

- Design a controller for Traffic light.
- Mention the property that the traffic light controller should satisfy.

Now what will happen, now what is the properties that both are rare like should not close simultaneously okay. On the other hand I can say that both the green light should not glow simultaneously or at the same time because if you glowing at the same time in that when what will happen will move from direction and there may be collusion. So what will happen if one special like that both the light should not glow simultaneously.

Similarly one is red say if not is lot of direction is your red then east ward direction means going to green okay. So these are the speciation now you have to identify the specification and try tom write the particular specification in your CTL and design the controller. So that you can control the correctly in this particular junction okay, now what we have to do we have seen that we are coming up with this model checking method.

(Refer Slide Time: 55:09)



- The "state explosion" problem
  - State space is exponential to the number of state variables.

Model checking algorithm we are having properties you are having a model and we are going to algorithm to check whether those properties are true in the model or not. Now how I am going to get the model we may have a problem because itself see that we are having n control signal these we are having n control signal then what will happen how many different combination we may have, we may have 2 to the power and different combinations. That means we may have 2 to the power n.

Define possible step all may not be reasonable because all we have seen in our visual expression protocol where we are having six controls signal. So total control signal is 2 to the power 6 but all may not be disabling but should be ready for the situation. If I am having non control signal then we are going 2 to the power default step. Now in our new design if we have increased this particular control then the control signal o n-m we need one more extra additional control signal.

Then the number of steps will go to 2 to the power and +1 okay. So if 5 control signals we are having 2 to the power 5 which control signal will be having 2 or 6. So if it is 32 next it will be 64 if we have one more control signal if will go for 2 to the power 7128 so this is stars number of steps returns on the number of control signal and it is exponential in nature so basically we are having a problem with this particular model checking algorithm.

Because they problem is known as your state expression problem because state expression is exponentional to the number of step variable. Now thou we have seen we have an atomic operator atomic procedure algorithm to go for model checking for CTL formulas what problem that we are having with this model checking is state expression problem. So now you are some research works are gaping on how to control this particular state expression problem so in our next class or next module what we are going to see is one particular approach to content this particular step in your state expression okay. I will stop here today.

Acknowledgement

Ministry of Human Resource & Development

Prof. Satyaki Roy
Co-ordinator, NPTEL IIT Kanpur

NPTEL Team
Sanjay Pal
Ashish Singh
Badal Pradhan
Tapobrata Das
Ram Chandra
Dilip Tripathi
Manoj Shrivastava
Padam Shukla
Sanjay Mishra
Shubham Rawat
Shikha Gupta
K. K. Mishra
Aradhana Singh

Sweta

Ashutosh Gairola

Dilip Katiyar

Sharwan

Hari Ram

Bhadra Rao

Puneet Kumar Bajpai

Lalty Dutta

Ajay Kanaujia

Shivendra Kumar Tiwari

an IIT Kanpur Production