**INDIAN INSTITUTE OF TECHNOLOGY GUWAHATHI**

**NPTEL**

**NPTEL ONLINE CERTIFICATION COURSE**
**An Initiative of MHRD**

# VLSI Design, Verification & Test

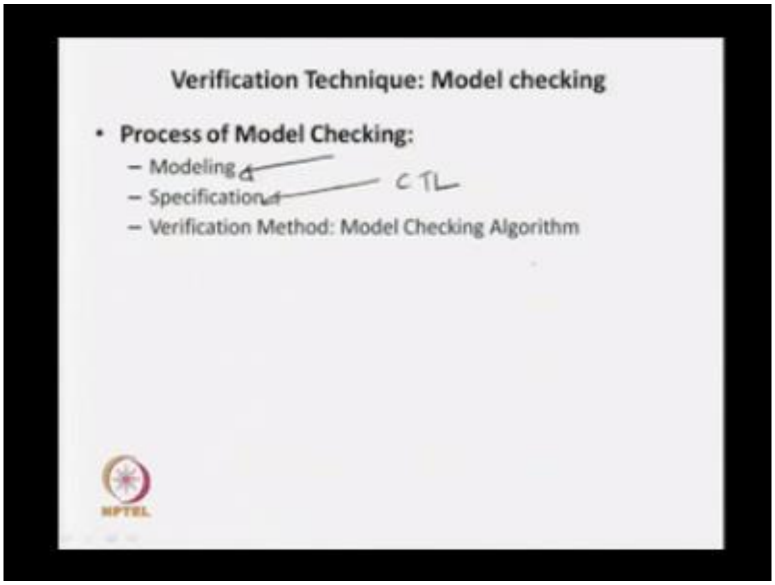
**Prof. Jatindra Kr. Deka**
**Department of CSE**
**IIT Guwahati**

## Module V: Verification Techniques

## Lecture II: Model Checking Algorithms

In last class we have discussed something about model checking.

(Refer Slide Time: 00:33)



**Verification Technique: Model checking**

- **Process of Model Checking:**
  - Modeling
  - Specification CTL
  - Verification Method: Model Checking Algorithm

So what happens basically it is a verification techniques and in this verification technique called model checking we are having three components basically what is your modeling of the system, member to specification basically you have to give the property we have to give the property in some formal languages, so can you, so that we can use some formulation to verify those property in the model.

And the verification method that we are going to talk about your model checking. So in last class I have introduced the concept of model checking with the help of an example, where we have seen that the designing of the mutual exclusion protocol. Now with this particular small example what we have seen that how to model a system that means how to come up with the model of this particular system.

Then we have to see what are the property that system or that mutual exclusion protocol should satisfy we have note down those particular property like safety property, liveness property, then or to talk about non-blocking and note take sequencing. Now these are the property or these are the requirements for our system.

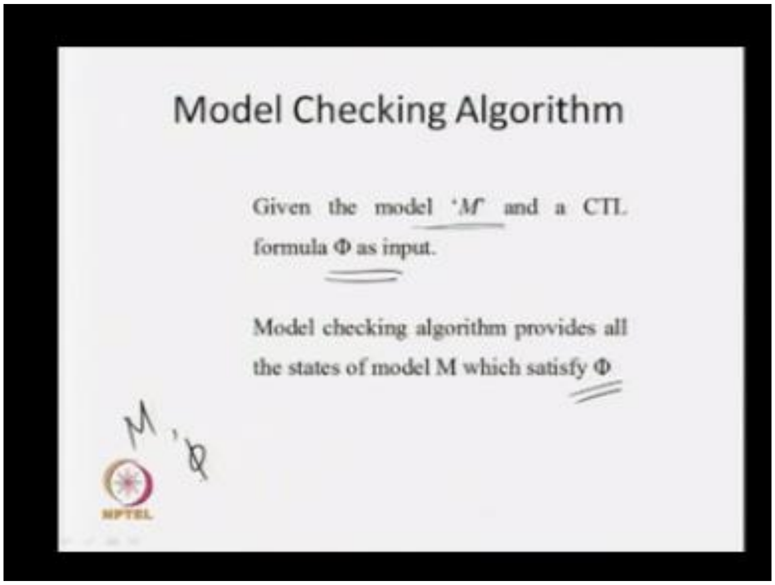
Then what happens those particular specification has to be captured in some formal way or in some formal language. Already we have discussed about CTL computational tree logic and we have seen how they have to be defined or expressed in your CTL. Then we have to see in the model look into this particular specification.

Now we are going to check whether those specification are true in this particular model or not okay. And that we have seen that the model that we have come up is not satisfying the liveness property, but we can modify it so that it can satisfy the liveness property also. Now say with this small example we have seen or given that there were all models you can watch.

Now like that when we are going for a bigger system then we will be having lot of states, the states does not been more. And already we have seen that basically the number of states in our model will depends on the number of control signals that we have. It is basically exponential with the number of signals that we have in the system which is equal to $2^n$.

So basically when we are going to look for a bigger system and we are having a complex formula or complex specification how to check for the correctness of those particular specification in a model for that we need some algorithm, we need some mechanized method and today we are going to see what are the algorithm that we have for this particular model checking methods. So we are going to talk about model checking algorithm today.

(Refer Slide Time: 03:12)



## Model Checking Algorithm

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

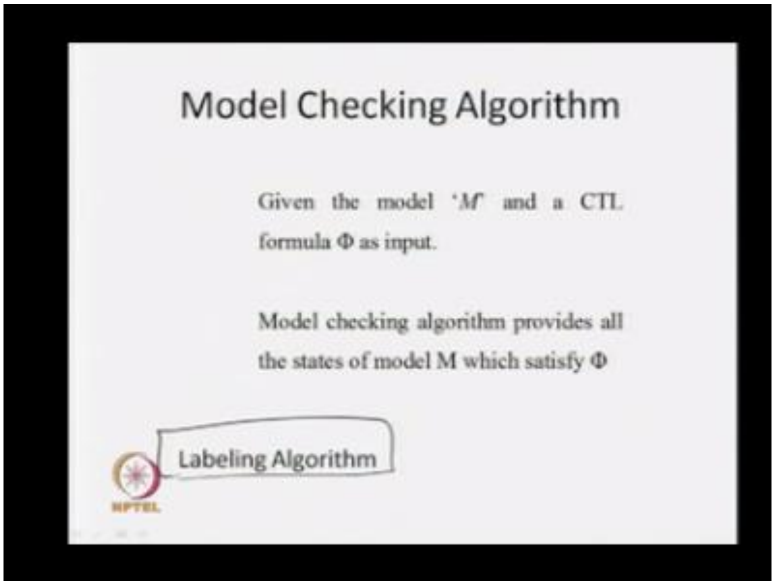
Model checking algorithm provides all the states of model M which satisfy Φ

M , φ

So in last class I have also slightly introduced about this particular model checking algorithms so basically what we have in this particular model checking algorithm we are having a model M, so that system model M that will be having and will come up with this particular system model and along with one CTL formula ϕ that means this is the property or the specification of the system.

Now how model checking algorithm is going to work, so model checking algorithm provides all the states of model M which satisfy ϕ. So basically we just say that we are going to give a model M and we will give a CTL formula ϕ and we are going to collect all the states where this particular formula ϕ is to in the model family.

(Refer Slide Time: 03:57)



## Model Checking Algorithm

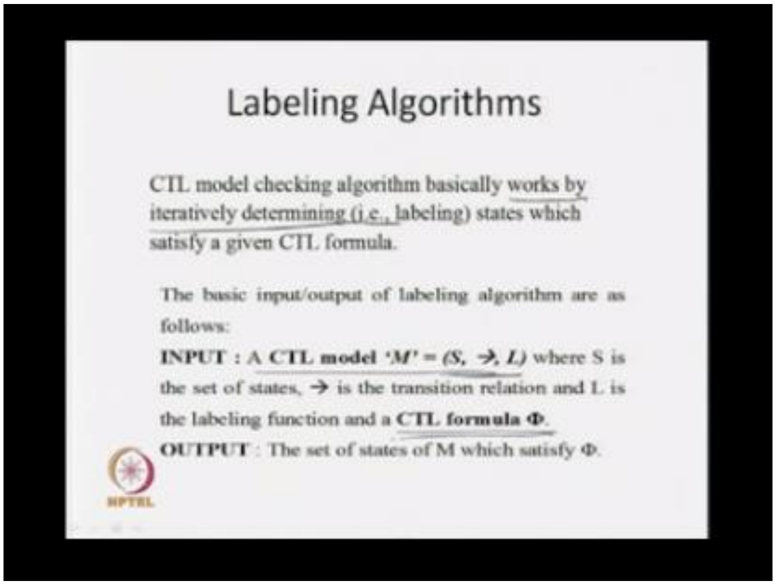
Given the model '*M*' and a CTL formula Φ as input.

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

Labeling Algorithm

So basically ultimately we are coming up with an labeling algorithm. So our basic M is to find out a labeling algorithm, this labeling algorithm is going to label the states of this particular model with the formula if it is true in the particular state and finally we are going to return those particular states where this particular formula is true. So this is the mode of model checking algorithm that we are going to look into it.

(Refer Slide Time: 04:24)



## Labeling Algorithms

CTL model checking algorithm basically works by iteratively determining (i.e., labeling) states which satisfy a given CTL formula.

The basic input/output of labeling algorithm are as follows:

**INPUT** : A **CTL model** ***'M' = (S, →, L)*** where S is the set of states, → is the transition relation and L is the labeling function and a **CTL formula Φ**.

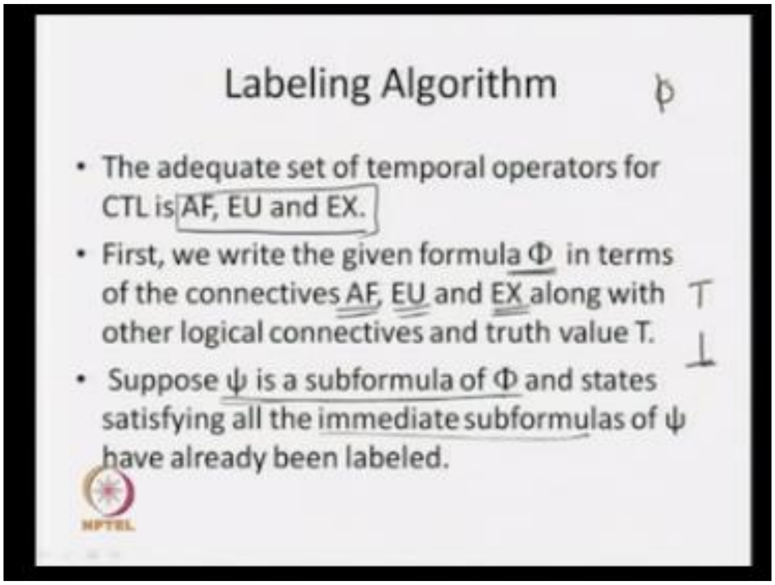
**OUTPUT** : The set of states of M which satisfy Φ.

So basically what labeling algorithm is, now we can look into it in this way CTL model checking algorithm basically works by iteratively determining that is labeling states which satisfy the given CTL formula. So here we are saying that it is worked by iteratively determining which will see or iteratively or having iterative method which is going to label in the states with how I got the formula where it is true.

Okay, after that after completion of these labeling algorithm it will return all the states where the given formula is true. The basic input, output to labeling algorithm are as follows. So what is our input, so we are having a CTL model M and we know that this model is nothing but state of state on condition lesson and the labeling functions.

So these are the input to the model and where as in the state of states that arrow is a transition and L is a labeling function. Along with that we are going to give a CTL formula ϕ, so what will be the output, the state of states of M which satisfy this particular ϕ okay.

(Refer Slide Time: 05:34)



## Labeling Algorithm $\phi$

- The adequate set of temporal operators for CTL is AF, EU and EX.
- First, we write the given formula Φ in terms of the connectives AF, EU and EX along with other logical connectives and truth value T. $\top$ $\perp$
- Suppose ψ is a subformula of Φ and states satisfying all the immediate subformulas of ψ have already been labeled.

So what is the labeling algorithm now same we have seen that what are the operators that we have for temporal operator, so we have in our CTL and they are basically temporal operator called the NEXT state, FUTURE, GLOBAL and UNTIL. And these particular four temporal operators will come up with the path quantifier A and E. So all together we are getting eight different combination.

So we need the method for those eight different combination or eight different CTL operator. But all that we have seen that out of those eight we need only three which are going to form the adequate set of operators. So if we know the method of three operators then we can go for all those particular eight.

So we know that the different set of operators we are going to take one particular different set of operator which includes AF, EU and EX that means in all path in FUTURE there exist a path with UNTIL and there exist a path NEXT state okay. So we will just say that since this is the adequate set of operator we are going to look for the method of these three particular operators a and so once I get the operator for these three operators then others can be expressed in the help of these three operators.

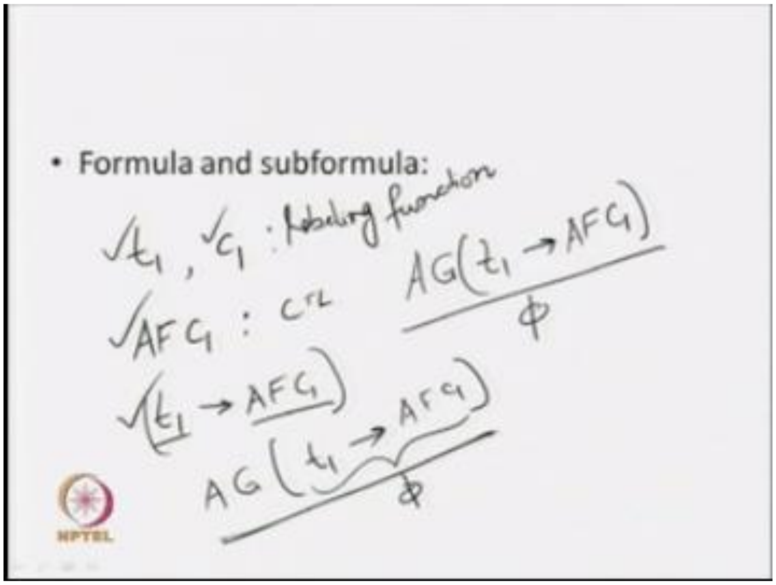
So when we are getting a particular formula $\phi$ say then first of all what we are going to do, we are going to express this particular formula in terms of the connecting AF, EU and EX because we are going to have method for these three operators along with the logical connectives and truth value is true, because we are having the truth values true one is true which is represented by top another one is bottom which is represented by false which is represented by bottom okay.

So we are going to look for the all logical connectives and the truth values true top and true which is top and false which is bottom. So if any $\phi$ we are getting in CTL formula $\phi$ we are getting first we are going to expect this particular CTL formula with the help of these three operators, because you know the equivalence, we are having some equivalence relation in CTL. Now suppose $\psi$ is a sub formula of $\phi$ now we are going to look for some sub formula.

So if $\psi$ is a sub formula of $\phi$ and states satisfy all the immediate sub formulas of $\psi$ have already been labeled. Now what does it means immediate sub formula means all the sub formulas that we have, so basically all level, when we are going to look for a particular formula first of all we know the components and we should know the truth values of those particular components, these are basically sub formulas.

Once we know the truth values of the formulas or sub formulas then we can look for the truth values of the main formula.

(Refer Slide Time: 08:22)



- Formula and subformula:

So as for example I can say that a formula say you know mutual exclusion formula, mutual exclusion protocol example for means we have seen the liveness for what is like that in AG(T1 implies AFC1) okay so this is the properties CTL property in AG(T1 implies AFC1) okay. So that in all path globally if T1 is true it implies that in all path in FUTURE C1 will be true. Now in this particular case you see that when we are going to look for these particular formula ϕ then we have to come up with the all the sub formulas of this particular ϕ.

So here T1 and t2 are your atomic proposition so they will CTL formula so we are going to check say that these are 2 CTL formula so 1 I am having T1 and C1 since these are atomic proposition so we are having that particular leveling function we know that we have the leveling function and with the help of leveling function we level states were the atomic proposition are true.

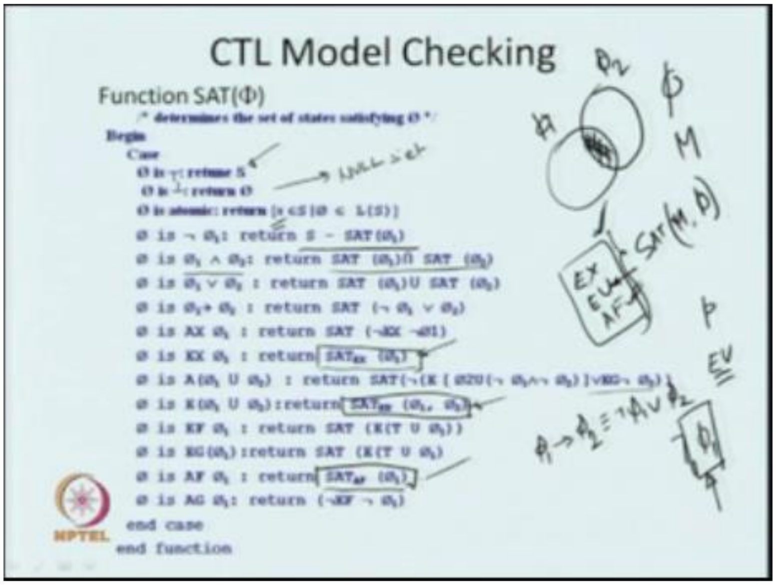
So we know the truth value of these two atomic proposition in our model and we know we would state these are true now once we know that these are CTL formula then we will find that AF C1 is also a CTL formula next what will happen we have to level all the state with this particular AF C1 once we know the truth values of this AF C1 in all the states then only we can for the net level so that is your saying the first we have look for all sun formula once a particular sub formula is leveled then we can do the next.

So once instead a CTL formula now with the help our algorithm we can look for the states where this particular formula is true then by looking into the main formula Ø then we are going to get this formula T1 → AF C1 okay so again this is a CTL formula it is having two component AF C1 and T1 now again with the help of all available algorithm we are going to level the state with this particular formula A T1 →  AF C1 so this is also a sub formula of the given formula now all the states will be level this particular sub formula.

Once we know the level of this particular sub formula then go for the next level next sub formula which is your AG → T1→  AF C1 now we know the level of this particular formula so we know the states where this particular formula is to so once we know this thing then we can go for this particular of given formula Ø so that is we saying that we must know the truth values of our sub formulas that means we should know the state where this particular sun formula is true and where this particular formula is true we are going to level those state with the help of this particular sub formula with the help of our leveling algorithm okay.

So this is the way we are going look into it so when we are going look for this particular Ø we should know the level or we should the truth values of all those particular sub formulas T1 C1 AF C1 and T1 → AF C1 so once we know this thing then we can go the complete given formula now we are going to say what with the algorithms to check for such type of formulas okay.

(Refer Slide Time: 12:00)



## CTL Model Checking

Function SAT(Φ)

```
    /* determines the set of states satisfying $\emptyset$ */
Begin
    Case
      $\emptyset$ is $\top$ : return $S$
      $\emptyset$ is $\bot$ : return $\emptyset$
      $\emptyset$ is atomic: return $\{s \in S \mid \emptyset \in L(S)\}$
      $\emptyset$ is $\neg \emptyset_1$: return $S - SAT(\emptyset_1)$
      $\emptyset$ is $\emptyset_1 \wedge \emptyset_2$: return $SAT(\emptyset_1) \cap SAT(\emptyset_2)$
      $\emptyset$ is $\emptyset_1 \vee \emptyset_2$ : return $SAT(\emptyset_1) \cup SAT(\emptyset_2)$
      $\emptyset$ is $\emptyset_1 \rightarrow \emptyset_2$ : return $SAT(\neg \emptyset_1 \vee \emptyset_2)$
      $\emptyset$ is $AX\ \emptyset_1$ : return $SAT(\neg EX \neg \emptyset 1)$
      $\emptyset$ is $EX\ \emptyset_1$ : return $SAT_{EX}(\emptyset_1)$
      $\emptyset$ is $A(\emptyset_1 \cup \emptyset_2)$ : return $SAT(\neg(E[\neg \emptyset 2 U(\neg \emptyset_1 \wedge \neg \emptyset_2)] \vee EG \neg \emptyset_2))$
      $\emptyset$ is $E(\emptyset_1 \cup \emptyset_2)$:return $SAT_{EU}(\emptyset_1, \emptyset_2)$
      $\emptyset$ is $EF\ \emptyset_1$ : return $SAT(E(T \cup \emptyset_1))$
      $\emptyset$ is $EG(\emptyset_1)$:return $SAT(E(T \cup \emptyset_1)$
      $\emptyset$ is $AF\ \emptyset_1$ : return $SAT_{AF}(\emptyset_1)$
      $\emptyset$ is $AG\ \emptyset_1$: return $(\neg EF \neg \emptyset_1)$
    end case
end function
```

Now you just see that we are in this particular model leveling algorithm what happen we are going to give a formula Ø and we are give a model M so we are going to defined a satisfy with the function which is know that function state and given input formula is Ø and some model is by default or we can defined that it is state can be defined as state satisfy will model M and the formula Ø open now in this particular case now as for the semantics of this particular CTL formula.

What happens we know that all state will be level with your truth value true and non of the states will be level with that truth value false so that is why if Ø true then it is going to return the complete state scape S because true  is true in every where if the Ø given formula Ø is your false bottom then it will enter the null sets so this is basically no where it is true so it is going to return the null set because the truth value false is false everywhere so it will not level with this particular false.

So if my given formula is your Ø then it is going to return the null set now if Ø is atomic so now we are given some atomic Ø is = to say some atomic variable P and what will happen then it will return the states those particular states where that state is a member of this particular leveling function of this particular state because we know that all state will be leveled by your atomic proposition.

Okay so with the help of leveling function we are leveling the states so if my given formula Ø is an your atomic proposition if it is atomic so it is going to return all such type of sets which are a member of this particular leveling function for that particular state S okay now if my given formula Ø is negation of say Ø 1 so it that I am giving a formula Ø which is negation of Ø1 so you just see that I am saying negation of Ø 1.

So when we are going to look for this particular formula so Ø1 is a sub formula of this particular given formula so we must know the level of Ø1 in each and every state so in my given formula is negation of some CTL formula then what it is going to returns it is going to returns the state where this particular Ø1 is not true that means first we are going to have the level of the state Ø1 is true then the remaining state will be your naught if Ø1 so it is going to return S- SAT Ø1 so this SAT value of Ø1 is going to give me the states.

Where the given formula is true so for negation of Ø1 it will be S total state space – satisfactory of Ø1 okay so just see that if naught Ø1 you consider it is having a sub formula Ø1 so first we know the leveling Ø1 then only we can go from the naught of Ø 1 similarly Ø is your Ø1 and Ø2 this is the logical connectives then what happens it is going to return the SAT first we will know the leveling Ø1 and Ø2 so it is going to return with state so the intersection of this state will be the state where this particular formula Ø1 and Ø2 is true.

So satisfactory of Ø1 is going to give me a SAT where the formula Ø1 is true so this is one state I can say that in this particular state Ø1 is true and satisfactory of Ø2 is going to give another SAT it is going to say that where the formula Ø2 is true now for this particular Ø1 and Ø2 that Ø1 and Ø2 must be true in both are true particular state so enter section will give me this particular state.

So it is going to return with particular intersection point so similarly Ø1 and Ø2 Ø1 or Ø2 it will any one of these two state again we know that if it is Ø1 → Ø2 then return naught of Ø1 on Ø2 because this equivalent we know that Ø1 → Ø2 is your equivalent to naught of Ø1 or naught of Ø2 so that is why this we are going to look for the satisfaction of the satisfactory of this thing and when call ultimately we are going to get these particular state now we know this particular formula that AX Ø1 is equivalent naught of EX naught of Ø1.

So we have seen this particular equivalent so if I'm given formula is AX Ø1 then we are going to look it will return going to look for this things satisfied with a naught of EX naught of Ø1 that means first we have express this particular AX in turns of EX because we are going to look for the minimal set of operator which involves EX another operator which is going to have EU and third one is your AF so this is the minimum set of operators that we are going to look for it so AX will be represent that with the help of EX now it is EX Ø1 now say I am going to say that there are exit a path next state Ø1.

Now what happens further it is going to return satisfaction with this particular EX Ø1 now in the other case we have expressing it some acting now for EX now this is a operator now this is the member of my minimal set of operators this is the minimal set or it is the set of operators that means is need a network for EX now we look for this particular EX similarly if it is A Ø1 and until Ø2 then we have discuss while discussing while discussing about the equivalent of CTL that can be represents that A Ø1 until Ø2 can be expressed with the help of EU okay or EG so this is the equivalence formula so why it is A(Ø1 U Ø2) then we are going to look for the satisfy ability of this particular  formula okay now while EØ1 U Ø2 then when my input formula Ø0 E Ø1 U Ø2 then we are to be tern satisfy ability EU Ø1, Ø2.

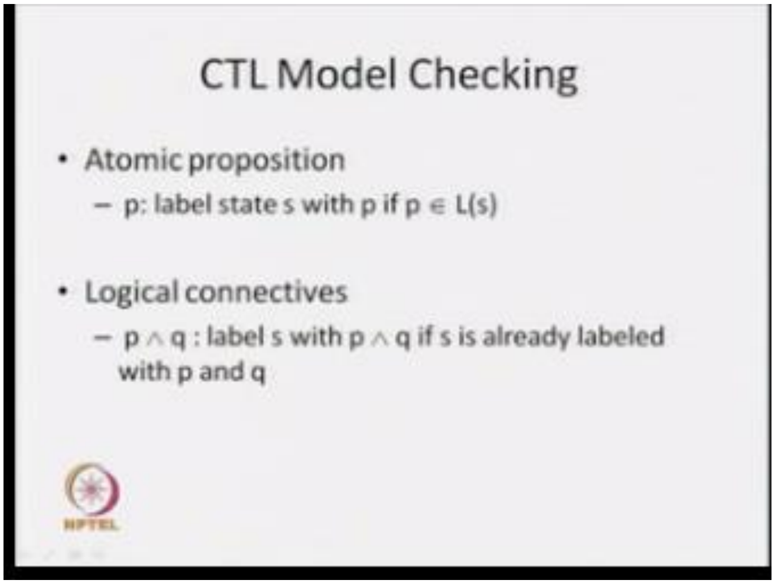
That means we need a method for this particular operator EU this one it was in my minimal set of operators, now when my given Ø is you EEF Ø1 then we are going to look for EP U Ø1 so 2 until Ø1 because we know that futures can be expressed with the help of your until operator so if it is yours EF then we need this particular E2 U Ø1, similarly if it is your AG EG then what will happen.

EG Ø1 this satisfying will be your E2 u Ø1 so AF so we need a satisfying function  like that AF is your satisfying ability AF Ø1 so since it is a member of my this particular minimal set of operator so I need a metal for AF so this is another method and we need one method for set EU okay similarly AG can be expressed with the help of this equivalence ¬P at ¬Ø1 now say now any kind of formula I am giving as an input to this particular  function set and it is going to give me the set of states where this for given formula is Ø.

Given formula is true now further since this is the minima set of operator so we need a metal for this particular three operator EX, EY and AF and in this particular three position we are going to call the appropriate method so what we have seen that what is for v that we are having so if you avenue CTL Formula then we are going to call the appropriate sub formula we will convert it to the previous of formula.

And we are going to get the state of set where this particular given formula is true and since this is the minimal set of operators so we need the method for set of EX and set of EU and set of AF now we are going to discuss about this three procedures, how what will be the alphabet group lacks for this particular EX, EU and AF operators.

(Refer Slide Time: 20:42)



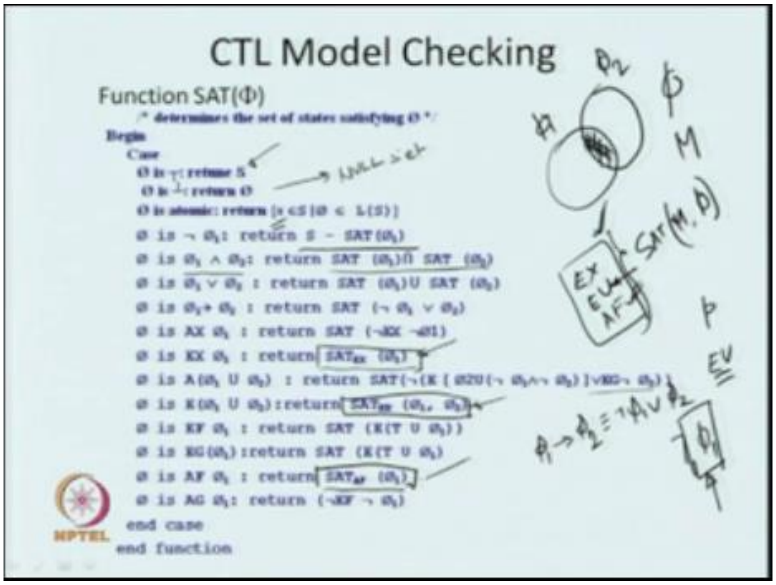
CTL Model Checking

- Atomic proposition
  - p: label state s with p if $p \in L(s)$
- Logical connectives
  - $p \wedge q$ : label s with $p \wedge q$ if s is already labeled with p and q

So it is if we are having an atomic proposition P then it is very simple we are going to level state with P if P is M, embed of LS because you just see that we are having these things if it is atomic then we are going to return all the states where it is a member of this particular leveling function. So we are going to say that given your formula is your atomic proposition then we are going to level the state S with P if it is a member P belongs to your Ls.

Leveling function of that particular state S, now similarly if we are having any logical connecting say P and Q then level S with P and Q if S is already leveled at P and Q because we have seen this particular thing.

(Refer Slide Time: 21:28)

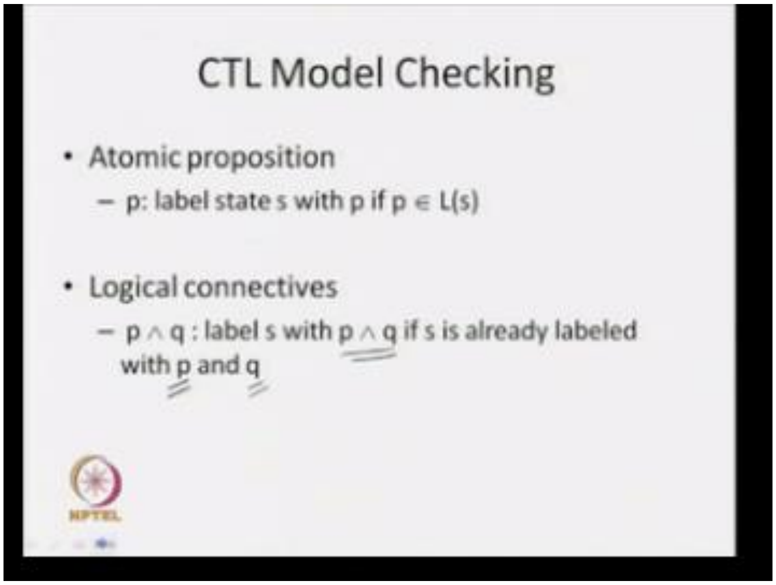


## CTL Model Checking

```
Function SAT(Φ)
    /* determines the set of states satisfying Ø */
Begin
  Case
    Ø is ⊤: return S
    Ø is ⊥: return Ø
    Ø is atomic: return {s ∈ S | Ø ∈ L(S)}
    Ø is ¬ Ø1: return S - SAT(Ø1)
    Ø is Ø1 ∧ Ø2: return SAT (Ø1)∩ SAT (Ø2)
    Ø is Ø1 ∨ Ø2 : return SAT (Ø1)U SAT (Ø2)
    Ø is Ø1→ Ø2 : return SAT (¬ Ø1 ∨ Ø2)
    Ø is AX Ø1 : return SAT (¬EX ¬Ø1)
    Ø is EX Ø1 : return SAT_EX (Ø1)
    Ø is A(Ø1 U Ø2) : return SAT(¬(E [ ¬Ø2U(¬ Ø1∧¬ Ø2) ]∨EG¬ Ø2))
    Ø is E(Ø1 U Ø2):return SAT_EU (Ø1, Ø2)
    Ø is EF Ø1 : return SAT (E(T U Ø1))
    Ø is EG(Ø1):return SAT (E(T U Ø1)
    Ø is AF Ø1 : return SAT_AF (Ø1)
    Ø is AG Ø1: return (¬EF ¬ Ø1)
  end case
end function
```

It is your PØ1 and Ø2 that we are going to look for the intersection of these two states.

(Refer Slide Time: 21:36)

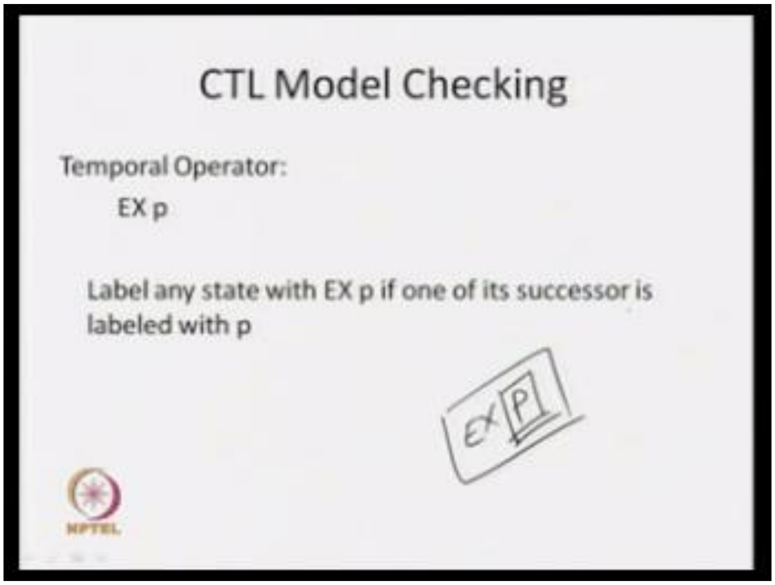


## CTL Model Checking

- Atomic proposition
  - p: label state s with p if $p \in L(s)$
- Logical connectives
  - $p \wedge q$ : label s with $p \wedge q$ if s is already labeled with p and q

So what will happen, if it is we are going to look for this particular P and Q then we are going to look for the states where both P and Q are true so if S is already leveled with your p and it is also leveled with Q then we can leveled this particular state S with P and Q, so like that we can go for any other logical connectives.

(Refer Slide Time: 21:57)



## CTL Model Checking

Temporal Operator:

EX p

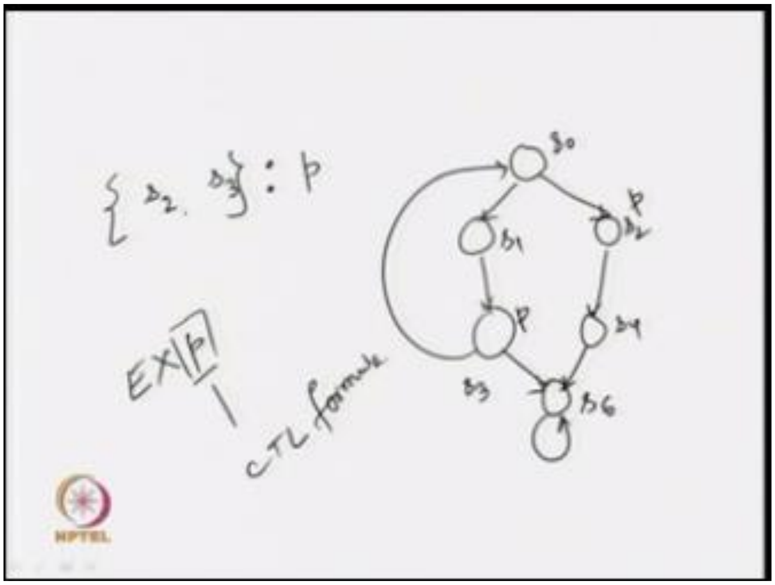
Label any state with EX p if one of its successor is labeled with p

$\boxed{EX\,\boxed{P}}$

Now temporal operator we are going to look for say EX this is one of the temporal operators all over minimal set of operator so all the adequate set of operators, so in this case it is their exist a part in next step P is true so level any state with EXP if one or with successor level with P so level any state with Exp if one or with successor level with P, so we are going to see the entire state space.

So in case of EXP so P is a sub formula first you know the leveling of this particular P if you notice thing then you can go for this particular given formula, so level any state with EXP if one or with successor level with P, okay.

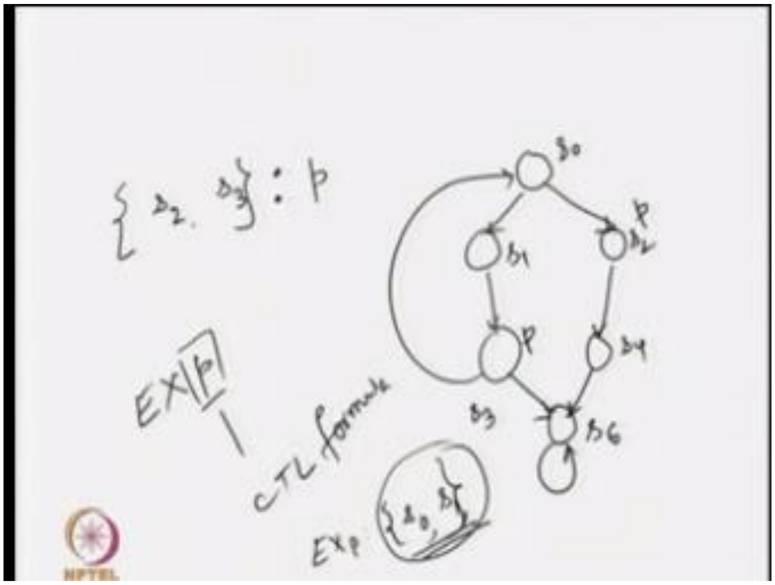
(Refer Slide Time: 22:44)



Okay so let us see one example. So this is a model you can say that this is the come if gets are ser so it is at a state of states so we are having form as 0 to S6 okay, S0, S1, S2, S3, S4, S6 now say that with the help of leveling function these S3 levels with P and say S2 is level with P that means we are going to get a set where S2 and S3 so in this set of states P is true because these are level with this particular P.

P may be atomic proportion or it may be any sub formula because when you go for EXP we must know the level of P so P may be any CTL Formula, okay.

(Refer Slide Time: 24:00)

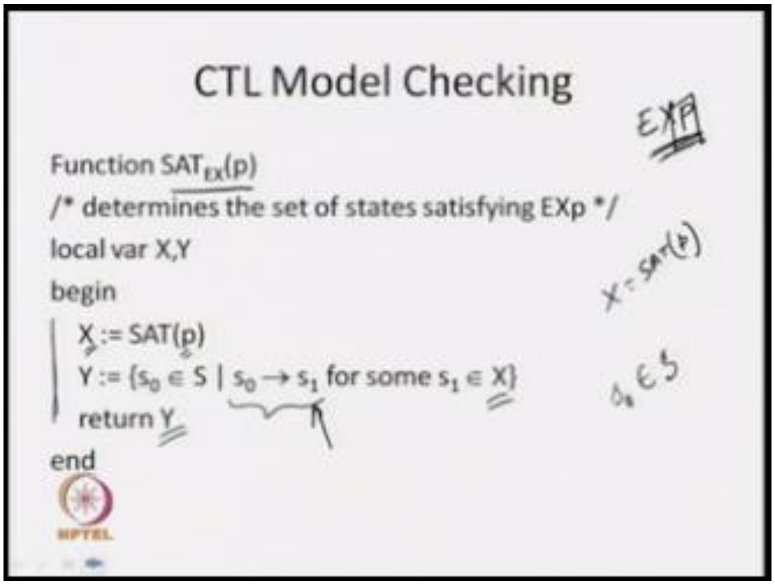


Now what you are saying that level any step with EXP if one or with successor with P so in this particular case if you see this things now if you come to S0 then it is having two successor S1 and S2 so one of the successor level is your P so it is the member of this particular state so for EXP what will happen we will get a s0 will come into these thing  when we come to S1 we will find that it is having one transition and this particular state S3.

Is having this mark with leveled with P so you can say that S1 will be leveled with your EXP, now when we come to your S2 it is having one successor S4 which is not have leveled with P so E S2 will not be leveled with EXP, when we come to you SØ, S3 then what will happen, you just see that it is having two successor one is S6 and second one is your S0 so we will see both the successor.

None of the successor is leveled with your P so S3 will not be leveled with your EXP when you come for S1, S4 it is having one successor s6 and it is not leveled with P, so it will not leveled with your EXP, when you come for your 6 it is having successor S6 itself, it is not leveled with P so it will not leveled with EXP, so basically we are getting these two steps S0 and S1 where EXP is true. So my, the satisfy with the algorithm or my the leveling algorithm will retain this particular state of S0 and S1 and it will say that in this two step my EXP is true.

(Refer Slide Time: 25:42)



## CTL Model Checking

```
Function $SAT_{EX}(p)$
/* determines the set of states satisfying EXp */
local var X,Y
begin
    $X := SAT(p)$
    $Y := \{s_0 \in S \mid s_0 \rightarrow s_1 \text{ for some } s_1 \in X\}$
    return Y
end
```

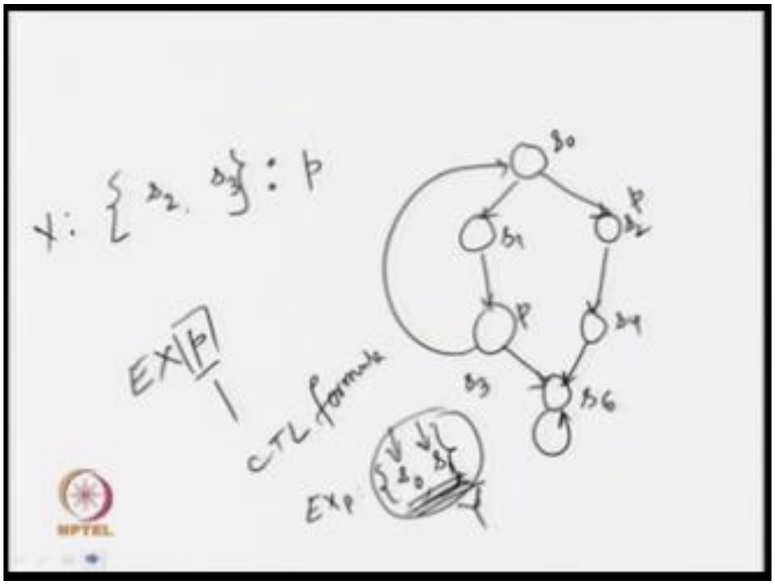
Now this is the algorithm you just see that what is the function it is function EX of this priority as set T EXP that means find out the state of stage where the formula EXP is true, so we are going to look for the formula EXP, it having the state of stage that is prime EXP, now here we are taking two local variable and Y now what is X, X first we are going to say it is calling this particular function satisfying ability.

With the sub formula so it is going to retain the state of step where this particular P is true so like that if you are going to look into these things, so this is the set X, S2 and S3, now what we are going to do, we are going to collect all those particular state S0 so it is some sort of SS0 whose belongs to that particular state of S of my model, in such a way that there should be transition from S0 to S1.

For some S1 belongs to X so in X whatever we are having all the states where P is true, so now we are going to collect all such type of state S0, where from this particular state S0 we are having a transition to the member of the this particular state X1 that means you are going to get some your distance state where in next step this particular formula p is 2, then after looking into it then we are going to return this formula state of step Y, so basically this particular step is going to give me a set of steps where the formula Exp is true so this is a very simple algorithm so giving the formula.

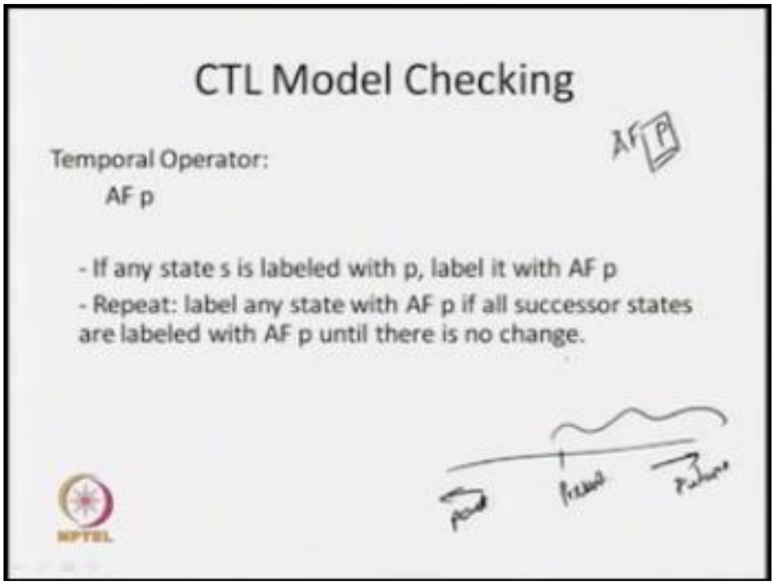
And if we know what is the sub formula p so you must know the leveling of this particular sub formula p so we are first going to collect those steps and we say that this is my x, okay. because we are going now calling this particular states probably say function with the p only or Exp so once you get this thing then now we can collect the other state where that EXp is true, okay so we are going to look for all such type of transition where s1 must be a member of this particular step x and what is the state this set x this is the steps where the formula p is true, okay so this is a very simple method.

(Refer Slide Time: 28:08)



So like that in previous example here we can say that first the x is your s2 and s3 now from here from those particular step we are going to look all the predecessor so it is going to be it may as 0 when it is your s3 then from here we are going to look all the predecessor where it is going to give me a s1 so these are the two state where this particular formula EXp is true and this is the Y that we are having set Y that we are have define as a local way about.

(Refer Slide Time: 28:40)



CTL Model Checking

Temporal Operator:

AF p

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

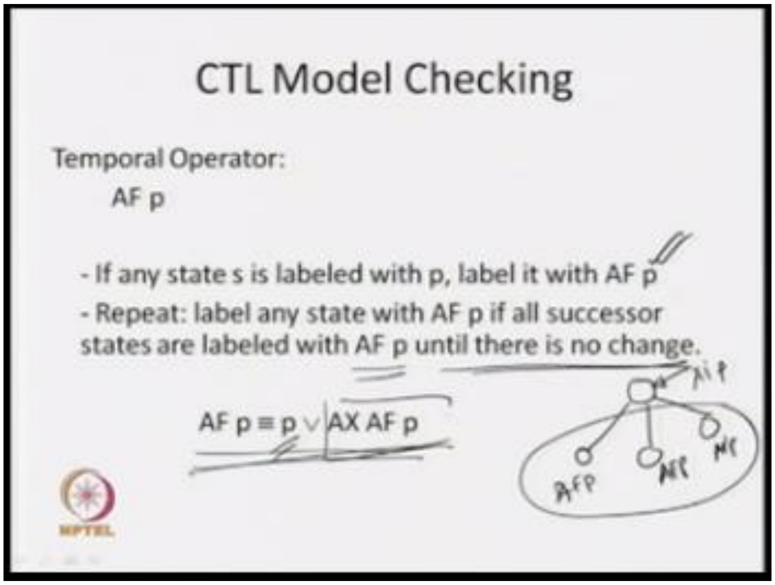
So next operator is your AFp okay, what is AFp basically in all part in predecessor p must be true so wherever you go we must get a state in peruse where p is true. Now how we are going to get an algorithm for this particular operator you just see that we are saying that if any state s is level with p level it AFp, now say if we are going to look for AFp then p is the sub formula we must have the level of p, so we are saying that if any state s is level with p then level it with AFp.

Why we can do this things, already we have defined a semantics and in our semantics what we have seen that peruse be have a includes the present scenario also, so basically if you look into the time line say this is my present and these are my future direction and that is my past, okay now while defining our semantics what we have seen that, the future behavior of the system includes the present behavior also so that means my future is from this particular present scenarios.

So this is as for the semantics that we have defined, since if p is true in presents that we can say that it will be true in the future also so if any state as it is level with p we are going to level with AFp, okay so this is the first step we are doing then we are doing to repeated like that level any state with AFp if all successor steps are leveled with AF p until there is no change. So now what is happen first this is our best condition is that state of that where p is true we are going to level those particular states with the formula AF p then we are going to repeat opposite to level any state with AF p if all successors states are leveled with AF p until there is no change.

That means you are going to collect the state selectively and if you cannot include any more states then we can stop at that particular points. So this is that saying that until there is no change, so see why you can do this things.

(Refer Slide Time: 30:50)



CTL Model Checking

Temporal Operator:

AF p

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

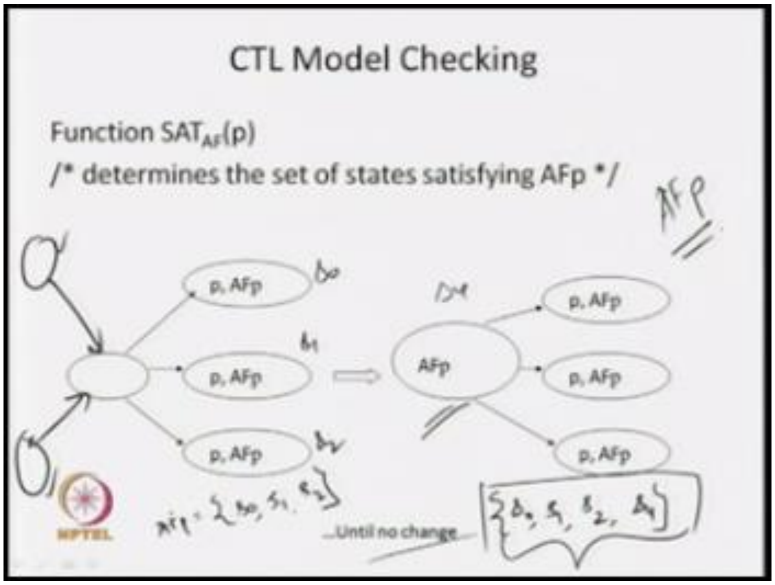
AF p ≡ p ∨ AX AF p

Because we have seen one equivalence that AF p we can write AF p with the help of this equivalence that p or AX AF p what it says that if p is true in a particular state then we will say that AF p is true so this particular p is captured by this particular my first condition, if any state as it is level it will be leveled it with F p and this is possible because my future includes the present behavior.

And what the next component we are saying that either p is to in this state or we are going to say that in all part next step AF p is true so that means if this is your either p is true over here or whatever states we are going to get here in all state AF p is true, so in all part next step F p is true, so like that if it is that then we can include this particular state of steps where this given formula AF p is true so that is why I am saying that level any state AF p if all successor states are leveled with AF p, okay we are gettable going to level it.

So if one leveling those particular states with your AF p because if p is true in all those particular state then we can very well level with your AF p. So in this particular case we can level this particular state with AF p so that is why we can say that like that we are going to collect each and every states that is why we are saying that we can repeat this particular about peruse until there is no change, no change of the state of state that we have collected. So this is done with respect to this particular equivalent say p is true that means this is the first state and we are repeating this particular second state for those particular position, okay.

(Refer Slide Time: 32:41)



CTL Model Checking

Function $SAT_{AF}(p)$

/* determines the set of states satisfying AFp */

...Until no change

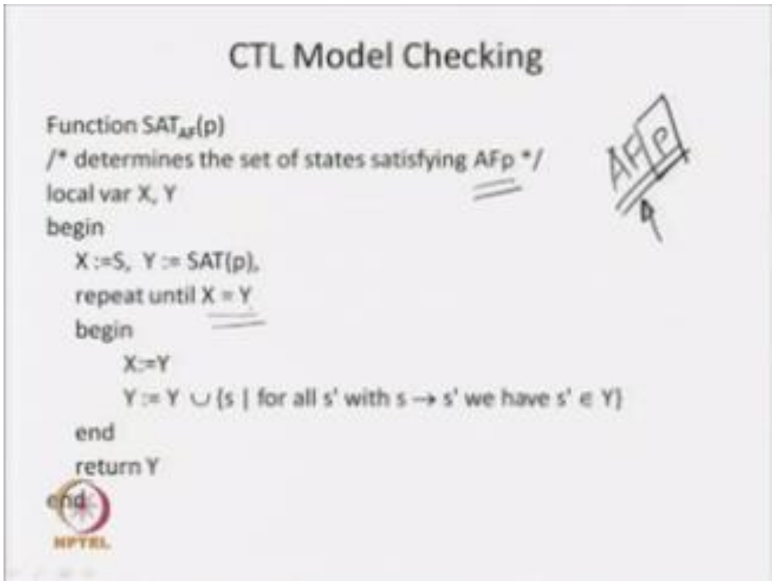
Now you just see that this is the given level so we are saying that just as an example, we are saying that these are the three states where p is true that means it is level with p and this one is not leveled with p. Now according to this particular first one we are seeing that if state s is leveled with p leveled it with AF p so since p is true in this particular distance so we are leveling with AF p, okay this is the first state then we are going to repeat our procedures so we are going to do it until there is no change.

Then what will happen since we know that AF p is true in this particular three states we are going to look for its predecessors since this predecessor we are getting on and from here we are saying that in all those particular three state AF p is true so we can level this particular state with also AF p, okay so this is the way that we are going to learn, so if we have some more states over here then what will happen so like that you can go backwards say if I can have some more state so once we know the level of these things then we can go for the level of those particular states also.

We will proceed in this way in backward manners and until there is no change that means we cannot collect any more state, that means what happen you just see that initially my this particular where is AF p is true I can say that it is true is your s0, s1, s2 so these are the three state s0, s1, s2. Now when we are done this particular state so this is my s4 that means the set of state where this particular formula is true becomes s0, s1, s2 and say s4 like that we will proceed and unless we cannot include any more states in this particular state then we will stop there and we are going to return this particular state of states but saying that these are the states where it is particular formula AF p is true, okay.

So this is the basic concept, now after having this particular concept now what we can do we can look for the algorithm.

(Refer Slide Time: 34:49)



CTL Model Checking

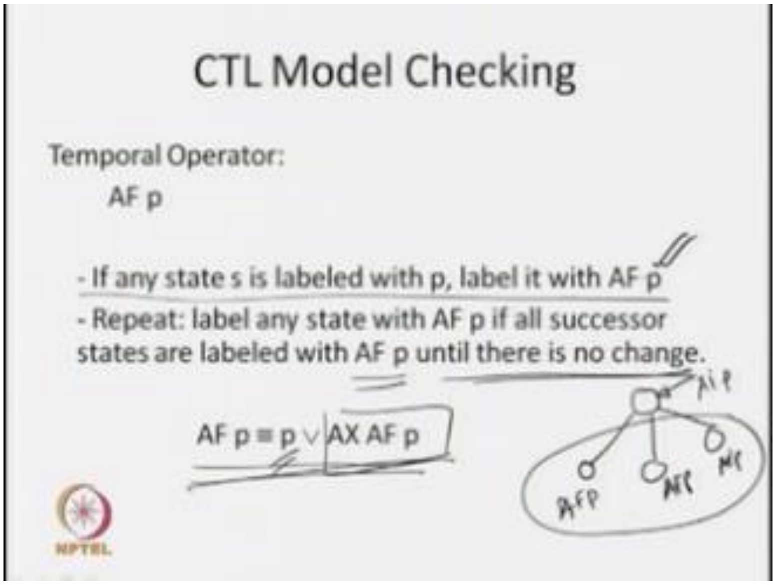
```
Function $SAT_{AF}(p)$
/* determines the set of states satisfying AFp */
local var X, Y
begin
    $X := S$,  $Y := SAT(p)$,
    repeat until $X = Y$
    begin
        $X := Y$
        $Y := Y \cup \{s \mid$ for all s' with $s \rightarrow s'$ we have $s' \in Y\}$
    end
    return Y
end
```

So this is the simple algorithm that we are having so it is state of AF p is going to determine the set of state where this particular formula AF p is true. Now like your previous case we are going to take help of two variable x and y and what is this x for initially we are initialization x to all the states basically this is the complete as we are taking we are saying that this is the complete set states we are initially saying to s, okay and what is your y, y is your calling this particular satisfying the function with this particular formula ϕ, okay ϕ is a NCTL formula and it is the sub formula of this particular given formula so basically we know the level of this particular formula AF p.

That means when you call this particular procedure that is probability with this particular sub formula of p it is going to return me those particular states where the given formula p is true. Now what happens so, wherever p is true we are going to say that AF p is true, so what we are doing so we are saying that now y is assign to x, okay so just saying that initially in all those particular state y the given formula f is true so this is in one step we have getting some step where this particular formula is true so you just keeping this particular information x okay. Now I am where going to do this particular loop until x = y so initially we having some step y where this particular formula p is true and this is the holds steps y.

So this is may not be equal at the verify in provide a in all steps are marked with p okay so until x = y we are going to repeat it so what we are doing we are keeping the previous information because in all state y AFp is true this is according to my first condition.

(Refer Slide Time: 36:47)



## CTL Model Checking

Temporal Operator:

AF p

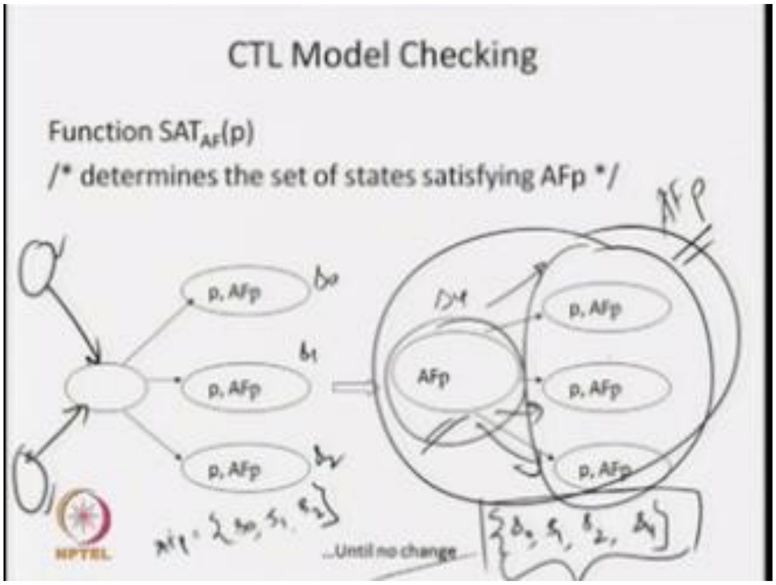
- If any state s is labeled with p, label it with AF p
- Repeat: label any state with AF p if all successor states are labeled with AF p until there is no change.

$AF\ p \equiv p \vee \boxed{AX\ AF\ p}$

AFP
AFP
AFP
AFP

If any state s is leveled it p and leveled it with AFP. So this is my first condition.

(Refer Slide Time: 36:55)



## CTL Model Checking

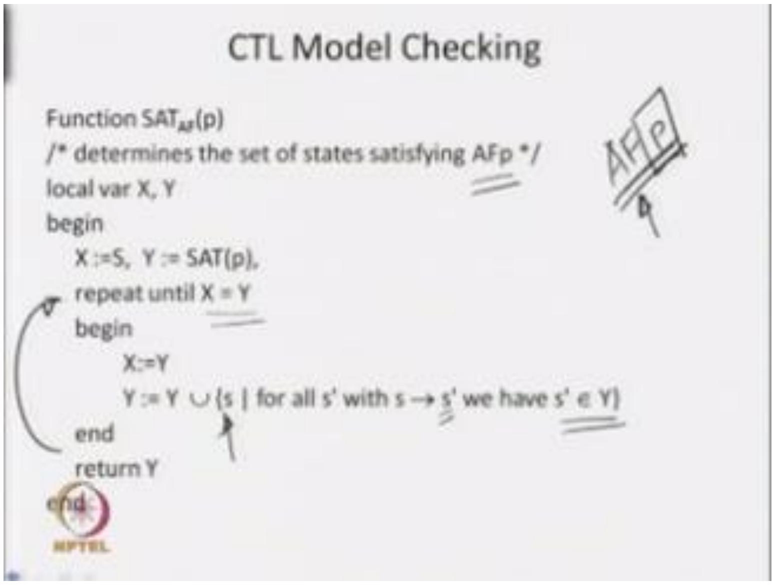
Function $SAT_{AF}(p)$

/* determines the set of states satisfying AFp */

So according to first condition in all the steps of this particular y are leveled with AFp so we are getting this initial step and what we are going to check now y will be update now, now what are the steps we can include in this particular state y you just see that if we are having such type of scenario if all the next states are leveled with you AFp and we can very well include this particular state to the set of state where AFP is true.

So that is why we are having this particular loop sentence why will be update in the y union what are this is things we are going to look for all such that of state as where we are having a transition from this particular s to all such type of s this and where s thus is a member of y okay, so this is the basic signal we are going to look for such type of step s4 and we are going to look for all the transition and all the next states that we are going to have they are AFP must be true over here so that means they are the member of this particular step y.

(Refer Slide Time: 37:59)



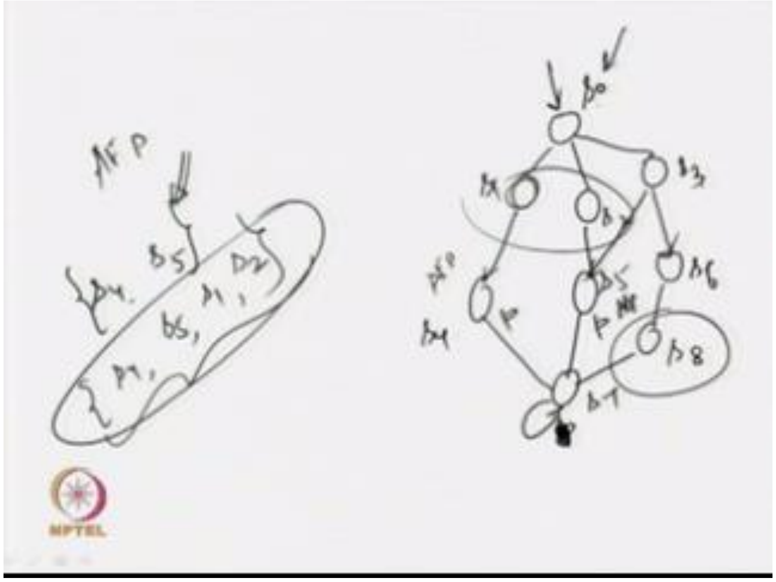
## CTL Model Checking

```
Function $SAT_{AF}(p)$
/* determines the set of states satisfying AFp */
local var X, Y
begin
   X :=S,  Y := SAT(p),
   repeat until X = Y
   begin
        X:=Y
        Y := Y ∪ {s | for all s' with s → s' we have s' ∈ Y}
   end
   return Y
end
```

Okay so that why we are saying that all such type of s thus we are going to look and those all s thus must be a member of y that means in all those particular s thus AFP is true so here now happening this particular y with those new states okay again we will repeat this particular loop and we are going to check whether x = y or not. If in any of this particular scenario y is not updated then x and y remain same.

So if that particular there we can turn in this particular procedure, you just see that we are collecting going from the particular step of state where p is true and they where seen the whole graph and collecting all the same. If there any point of time we cannot include anymore states it is going to say that now you stop here itself because now there is not possibility we are including anymore steps because you just see that any one example say.

(Refer Slide Time: 39:00)



You are looking for in all part includes of this 2 do not say that p is 2 her this two are marked it so these are step s0 s1 s2 s3 s4 s5 s6 s7 s8, now s for this things we are noting for a fast step we are going to start with your s4 and s5 so these are the two steps so now it is the basic step now we are starting with s4 and s5 now we are going to look for those particular prestigious step now we are coming to this s1 wherever you go in all so this will be well did your AFP this is also leveled with AFP.

So in all step I am coming to this particular step and we are saying that we are getting one particular step s4 only one step which is leveled it to FP so I can include this one s4 s5 and I can include now s1 also okay. Similarly when s5 is also member over here then what will happen I am going to get one prestigious step so I will give you, so we are giving coming to s2 so there is only one successor step where it is leveled with you FP.

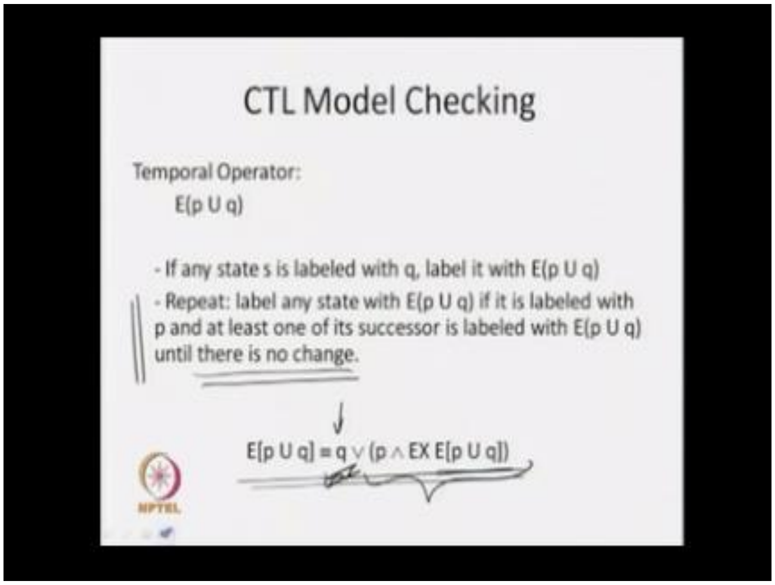
So I can include this particular s2 over her now this particular s5 and prestigious s3 no when we coming to s2 will find that it is having two successor one is leveled with AFP one is not leveled it AFP so s3 cannot be included, so we are coming to this particular scenario. Now in this particular case what will happen now y is updated form s4 s5 s1 s2 now will go in to the loop now what will happen in this particular case now these two members of you need that we are going to look for the producers of, so s1 s0 now in s0 we have increase successor s1 s2 and s3 s1 s2 are leveled it your AFp.

But s3 is not leveled with your AFP so we cannot leveled s0 with AFP so in this particular case my steps remains them so there is no same so at that particular point we can stop our procedure and I said that these are the four step were we are going to implies, because in this particular case since no more steps are including over here so there is no possibility of getting anymore step because you are going to check all the particular step, that means if someone is not member over here that means it is not a member of here like that say s8 if you say this things.

So what will happen from s8 you will go to s7 and p is not true over here so we are not going to get any part where in future in p is true otherwise if  p is true here than in the very fast case p would have include over here also.

So there is no chance that anymore steps will be included over here so if there is no chance in this particular state of step that we can stop the procedure her itself okay so this is why you are saying that until there is no chance so eventually we have coming up with the very elegant procedure so this procedure can be with not implemented. First collect all the set of set where p is true then repeat this particular loops okay.

(Refer Slide Time: 42:52)



## CTL Model Checking

Temporal Operator:

E(p U q)

- If any state s is labeled with q, label it with E(p U q)
- Repeat: label any state with E(p U q) if it is labeled with p and at least one of its successor is labeled with E(p U q) until there is no change.

$E[p \; U \; q] \equiv q \vee (p \wedge EX \; E[p \; U \; q])$

No another operator that temporal operator or CTL operator that we need is your EP and el q now how we are going to see that Ep until q they are exist the part where p remains true and until q becomes true now again you see that according to all symmetric in future includes the present behavior so that is why the past there is coming if any state s is leveled with q leveled it will ep until q.

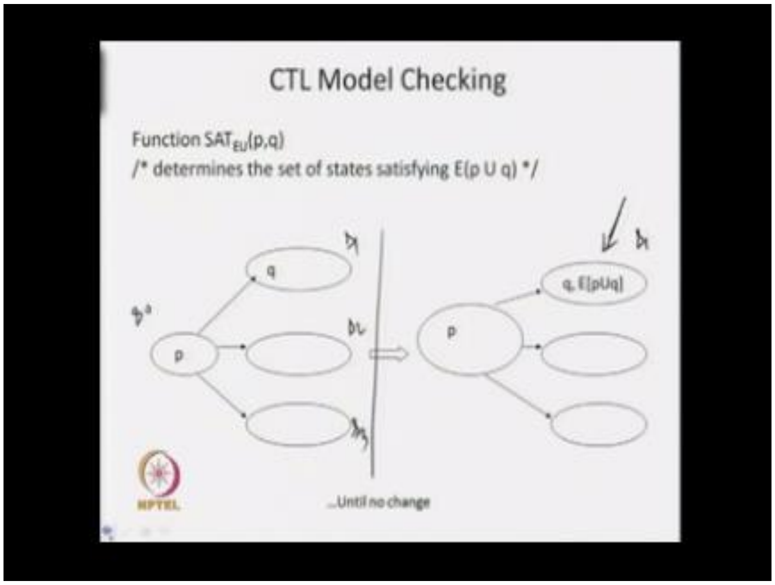
Because future includes the present behavior so if any states is leveled with your q we can say that ep until q is true over here so we are going to level this particular states with ep until q, now we are going to repeat this particular procedure level any state we ep until q if it is leveled with p and at least one of we successor is leveled with ep until q okay. So p should remain to until q become and we need at least one part so that is why I am saying that at least one or which successor is level with ep until q.

And p must be true at that particular part so if the state is leveled it p and if we will find that any of the successor is leveled with ep until q then we can leveled this particular state it ep until q because if anyone of the successor is leveled with ep until q at least you will see that in future somewhere q is true and till that particular point p remains true so they try you can said that past is any s is leveled with q you are going to leveled with ep until q and we are going to repeat this procedure.

Why this is possible because we have seen this particular equivalent okay already we have seen in this equivalent what we said that ep until q can be expresses q or p and they are the exits step part next step ep until q so this is you see that if p is q is true in as we are going to leveled with p until q so this is the first part first result if Q is true then we are going to level it EP and then if Q is true then fine we are going to get those particular step and if is a or we are going to have this particular condition this is the second clause order. Repeat one we must be true in that step and we are adjustable for the in next I will do with P and Q. So from this particular equivalent we are developing this particular method.

Now we are going to repeat it on until there is no change so we are going to collect step one after another and we are going to stop in a particular point when we cannot include anymore steps. So we are going to step until there is no change this is the equivalent so we form this particular equivalent in the we have got a method for Ep and q.

(Refer Slide Time: 45:55)



CTL Model Checking

Function $SAT_{EU}(p,q)$

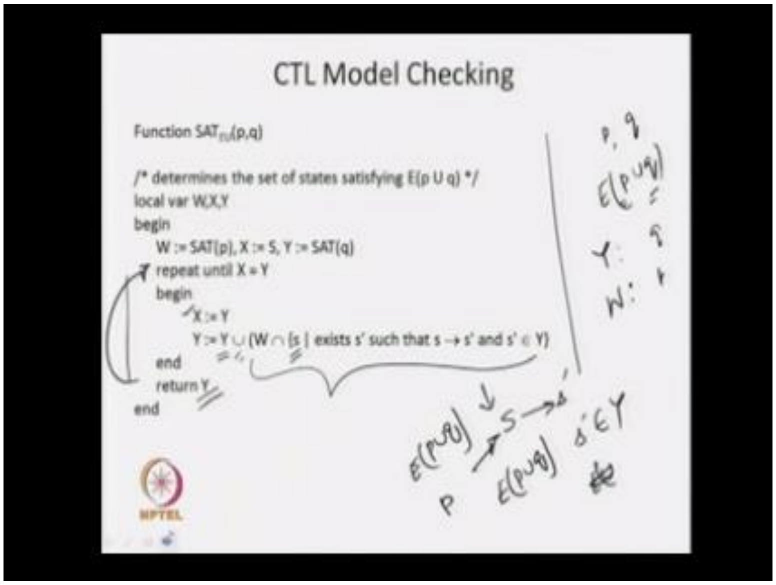
/* determines the set of states satisfying E(p U q) */

...Until no change

So we see that simple example I am saying similar example I am taking say these are the three steps initially we are having say s0, s1, s2, s3, now as per the first condition wherever Q is true we are going to level it Ep and q so by looking into this what will happen we can level this particular step s1, with p Ep and p q okay. Now this is the state were q is so we can level it p and q then what will happen once you know the level then we can refer the graph and we can say that we are going to follow this particular step.

We are going to look for the predecessor here and if this is leveled with p then we can level it with p and pq because it is leveled with p and anyone of the particular successor level Ep and we say p remain until q becomes true. So since K is true you are leveling with Ep and q and we are coming to this particular step and we are seeing the p is true so we can level it with Ep and q, so this is the way that we are going to reverse the graph and until there is no change okay.

(Refer Slide Time: 47:09)



## CTL Model Checking

```
Function $SAT_{EU}(p,q)$

/* determines the set of states satisfying $E(p \cup q)$ */
local var W,X,Y
begin
    $W := SAT(p), X := S, Y := SAT(q)$
    repeat until $X = Y$
    begin
        $X := Y$
        $Y := Y \cup (W \cap \{s \mid$ exists $s'$ such that $s \rightarrow s'$ and $s' \in Y)$
    end
    return Y
end
```

Now look into the procedure so we are going to get the elegant procedure for this particular Ep and q also so we are going to functions which is for E until p and q so we are going to take three variables WXY now we are having sub formula p and q because you are going to look for the E p until q. This is one formula this is another formula so in W we are going to set that we are going to call this particular set of function p because we know we must know the level of p. This is the set W to X is basically we are going to send that under the step X=S and y is your set stability of this particular q.

So we are going to collect all the steps were q is true we are going to say that this is my particular set Y. So Y is set of set where q is 2 and W is your set of square were this particular formula is P is true. Now we are going to repeat this particular loop okay. Now what will say that Y if you are going to look at Y, Y is a state where this q is true and we have seen that wherever q is true we are going to say that Ep and q is also true. So if any state that level with q then we are going to level with Ep until q.

And this is because of our efficiency of a system is because we saw in a crescent behaviour okay. So that is why this is a set of set where Ep until q is true. Now we are going to collect this step were it is true and we are going to apply in this particular step one. Now what will happen if the set of step now we are going to set up these are the previous condition we are going to keep so in set of step, Y P and until q is true. So we just are having this particular step acts as a set the particular step.

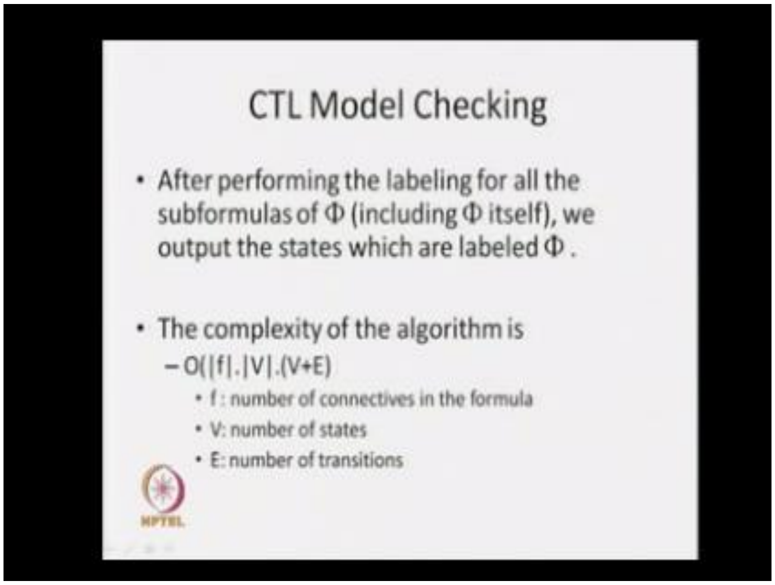
Now we are going to apply this particular Y so how we are applying it this is your Y union that means we are going to look for the steps where our given or required condition is satisfied. So it should be suitable like the W this is a set of step where P is true so W intersection we are going to collect all sub set of s so from s to s thus we are having transition. And this particular S dash is a member of Y so that means in as the we are going to see is already if until if Ep until q already is true in this particular state of step.

So if we are having the transistor form of set s to the set as well Ep and q and p is true we are having a consist from as and we can say that s p and q is true okay. So that is why we are applying it so we have and second condition is in as p must be true that is why whatever steps we are collecting along with that we intersecting with the W. W says that it is a set of sets where p is true so that step P must be true and whatever steps we are getting we are applying with this particular point.

And we are going back to this particular step we are going to check whether X=Y or not what we have in X this is a previous set of sets where we are Ep and q is true we are going to check whether any new more states are added or not if it is that means these are not equal again we will go into this particular loop we are going to correct. So  it is a set of procedure we are having and this is nothing but the graph has got here cause we are going to start from some stage of this particular graph and we are going to look for all the predecessor if my given condition is true or given situation is true in this particular predecessor.

We include this particular in my set of steps then again I will going to repeat this whatever news gets we are going to form this particular step we are going to look for the predecessor and if now any new steps cannot be added this particular step then we are not going to get anymore step and this is the final state of set we are going to get and that is why we returning this particular step Y okay.

(Refer Slide Time: 51:48)



## CTL Model Checking

- After performing the labeling for all the subformulas of $\Phi$ (including $\Phi$ itself), we output the states which are labeled $\Phi$ .
- The complexity of the algorithm is
  - O(|f|.|V|.(V+E)
    - f : number of connectives in the formula
    - V: number of states
    - E: number of transitions

So these are the steps we are going to see so in your CTL model checking algorithm we are going to have we are saying that after performing the level for all the steps for φ including φ we output a set of states which are leveled are required finally we are going to give those particular step were bit is true so use a set we have an minimal set of operators we are cosideri8ng Ex AF and AU. So we are simply seeing how this is implemented and have got very elegant algorithm and elegant procedures to implement these three operators.

So once we need know the algorithm for this three operators now others can be expressed with the help of these three okay. Now what will happen now we have to go for look for each and every step for this formula and once we know the sub formula then only we will get the main formula so that is why we can set the time required to do this particular method while checking if you find that you can say the time complexity of the algorithm will be your order of F and set of seta and this is the complete graph V+E. In set of set and set of condition that we have,

This is basically I have said the number of connective in a formula that is the number of formula we have basically we see that if I am going to do that φ 1 and φ2 algorithm in two steps. Because first I have to look for the φ1 and then we have to look for the φ2 we know the leveling of this then only I can go for the conjunction of these two things that is why I say it is a number of connectives for the length of the formula that we have because when I give this particular formula we must know the true values of φ1 and φ2.

Then only we can loop for this one and instead of some we are collecting some steps we need to to see all the predecessors steps in X this is only one level in case of AX Eu we have to repeatedly look for this particular thing so from all step we have to look for this predecessor and yield the number condition that means in was we have to complete graph so that is why this is the complexity of my leveling algorithm so this is your liner to the length of the formula and we say that the quadratic to the numbers of nodes we have in this particular graphs.

So this is a simple graph complexity algorithm and complexity depend on the length of the formula plus for every sub formula all this particular process in your and we have we have to look for each and every step of this particular problem. So this is basically we are going to have an linear time algorithm of this particular procedure so this is the beauty of the CTL algorithm so0 and it is easy to auto map.

So these are the basically algorithm we have seen and in next class we are going to see some simple example we will explain and we will see how we are going to or we will see the execution of this par5ticuar algorithm in next class again we are going to discuss about these three operators of the wi9tyh the help of some examples okay I will stop here today.

**Centre For Educational Technology**
**IIT Guwahati**
**Production**

**Head CET**
**Prof. Sunil Khijwania**

**CET Production Team**
**Bikask Jyoti Nath**
**CS Bhaskar Bora**
**Dibyajyoti Lahkar**
**Kallal Barua**
**Kaushik Kr. Sarma**
**Queen Barman**
**Rekha Hazarika**

**CET Administrative Team**
**Susanta Sarma**
**Swapan Debnath**