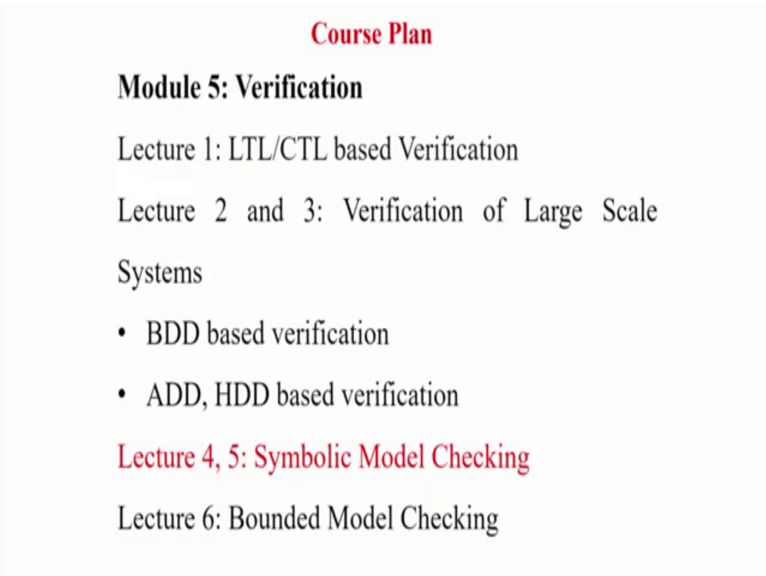
Optimization Techniques for Digital VLSI Design
Dr. Chandan Karfa
Dr. Santosh Biswas
Department of Computer Science & Engineering
Indian Institute of Technology Guwahati

Lecture – 23
Verification: Symbolic Model Checking

Hello everybody and welcome to the next lecture on verification, so if you recall we are in the double lecture series on symbolic model checking.

(Refer Slide Time: 00:35)

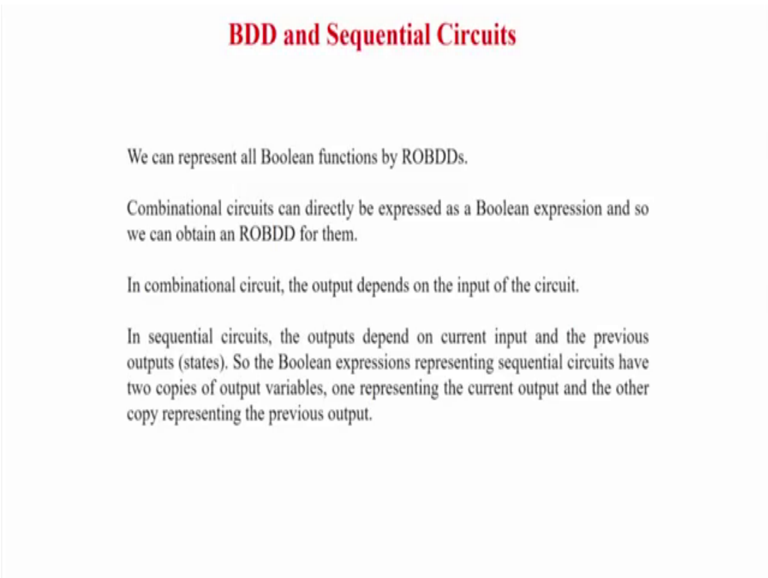


So, just quickly recollecting what we have a discussing is that if we have to look at the with model based verification, so we already found that it is actually the modeling of the system which is actually taking a lot of complexity and on that if you are want if you want to model check any of this CTL or the LTL formulas, then what we are been killed about in the exponential complexity is the size of the model.

So, what after that basically we have seen that BDD which is one of the very key role playing mechanism to help us to elevate the complexity problem, to eliminate the complexity problem is the BDD that; with BDD or symbolic mechanism we can handle the complexity up to a large extent because if you are using symbolic modeling and verification we are not exactly modeling the system in terms of explicit state space, rather we are modeling the system in terms of binary decision diagrams and then we are using those binary decision diagrams itself to model check the formulas.

Because we have already seen that model checking means just the labeling algorithms which we have a linear or you can think about non exponential complexity, the exponential complexity is basically incurred in the size of the model. So, what here we are dwelling in a symbolic model checking we were representing the system itself in terms of BDD, rather than the explicit state based enumeration then all the labeling algorithms we were doing in the symbolic model checking. So, in the first lecture basically we quickly covered.

(Refer Slide Time: 01:59)



That is lecture number 4 we quickly covered about.

(Refer Slide Time: 02:02)

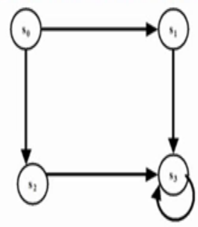
BDD and Sequential Circuits

- Sequential circuits are expressed with the help of state transition diagrams.
- In the case of sequential circuits, we have some inputs, some outputs and some state variables.
- The state variables can either represent current states or next states. The next state depends on current state and input variables.
- Thus, the next states can be represented by OBDDs.
- Even the transitions can be represented by OBDDs.
- Thus, we can represent all sequential circuits with the help of OBDDs.

How BDDs can be represent in the usual represent sequential circuits.

(Refer Slide Time: 02:05)

Representing subsets of sets of states using OBDDs



Here, the states s_0, s_1, s_2 and s_3 can be distinguished using two state variables, say x_1 and x_2. Let us represent them as follows.

$$\{s_0\} = \overline{x_1}\,\overline{x_2}$$
$$\{s_1\} = \overline{x_1}x_2$$
$$\{s_2\} = x_1\overline{x_2}$$
$$\{s_3\} = x_1x_2$$

(Refer Slide Time: 02:06)

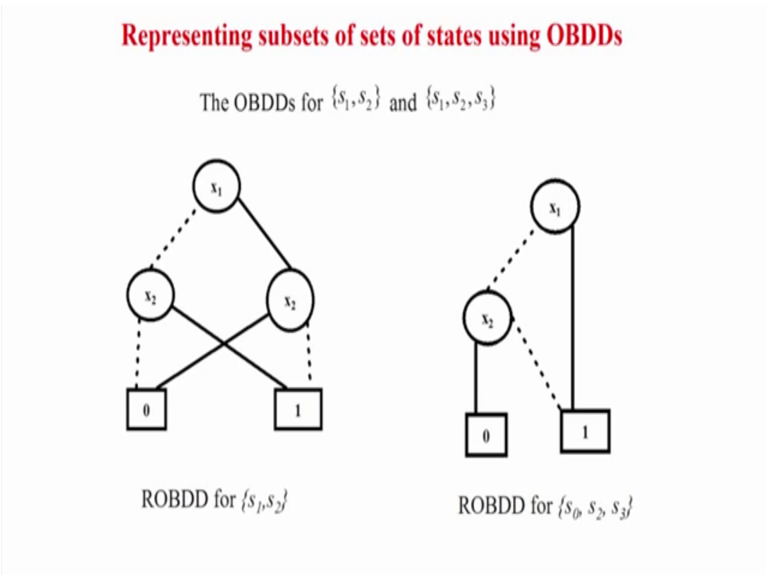
Representing subsets of sets of states using OBDDs

Since we use Boolean functions to represent subsets, we can represent all the subsets of states (s_0, s_1, s_2 and s_3) using OBDDs.

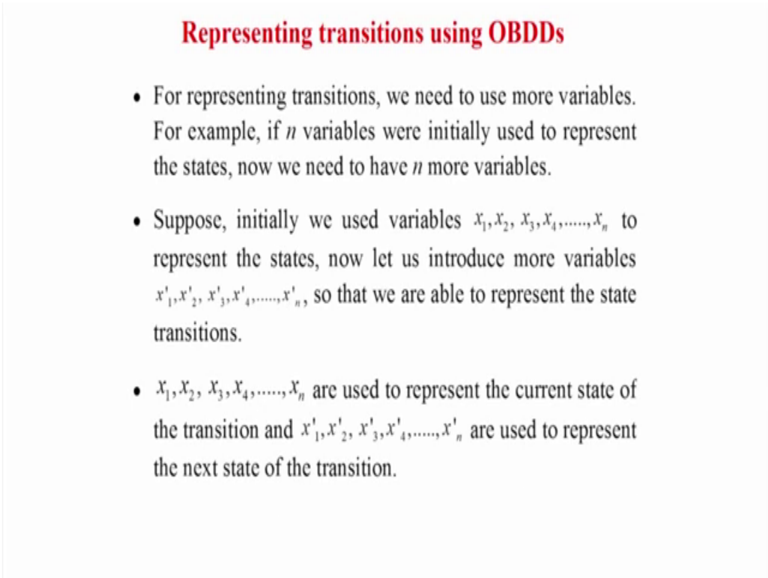
$$\{s_0, s_1\} = \overline{x_1}\,\overline{x_2} + \overline{x_1}x_2$$
$$\{s_0, s_2\} = \overline{x_1}\,\overline{x_2} + x_1\overline{x_2}$$
$$\{s_0, s_3\} = \overline{x_1}\,\overline{x_2} + x_1x_2$$
$$\{s_1, s_2\} = \overline{x_1}x_2 + x_1\overline{x_2}$$
$$\{s_1, s_3\} = \overline{x_1}x_2 + x_1x_3$$
$$\{s_2, s_3\} = x_1\overline{x_2} + x_1x_3$$
$$\{s_0, s_1, s_2\} = \overline{x_1}\,\overline{x_2} + \overline{x_1}x_2 + x_1\overline{x_2}$$
$$\{s_0, s_1, s_3\} = \overline{x_1}\,\overline{x_2} + \overline{x_1}x_2 + x_1x_3$$
$$\{s_0, s_2, s_3\} = \overline{x_1}\,\overline{x_2} + x_1\overline{x_2} + x_1x_3$$
$$\{s_0, s_1, s_2, s_3\} = \overline{x_1}\,\overline{x_2} + \overline{x_1}x_2 + x_1\overline{x_2} + x_1x_3$$

Basically these states and then we have seen.

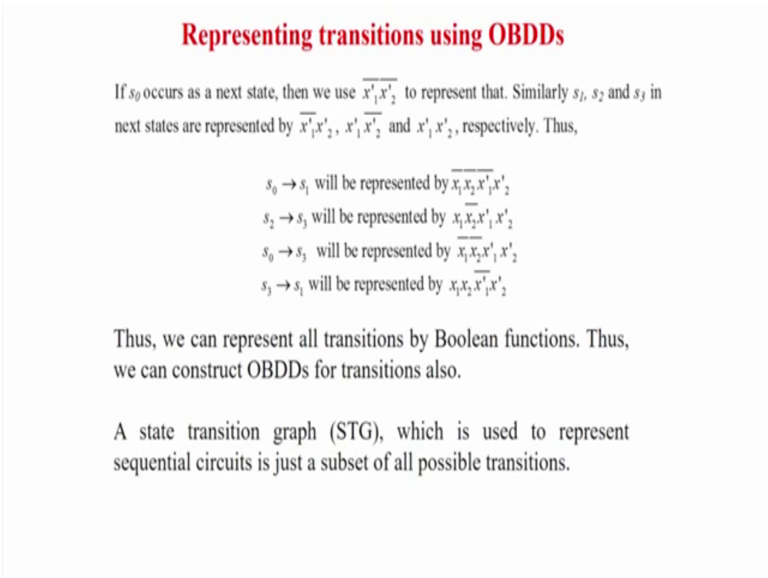
(Refer Slide Time: 02:07)



(Refer Slide Time: 02:08)

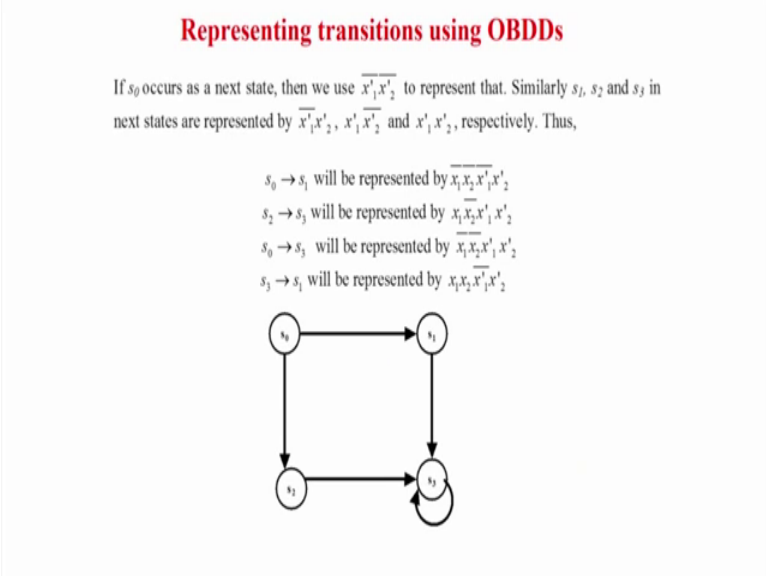


(Refer Slide Time: 02:09)



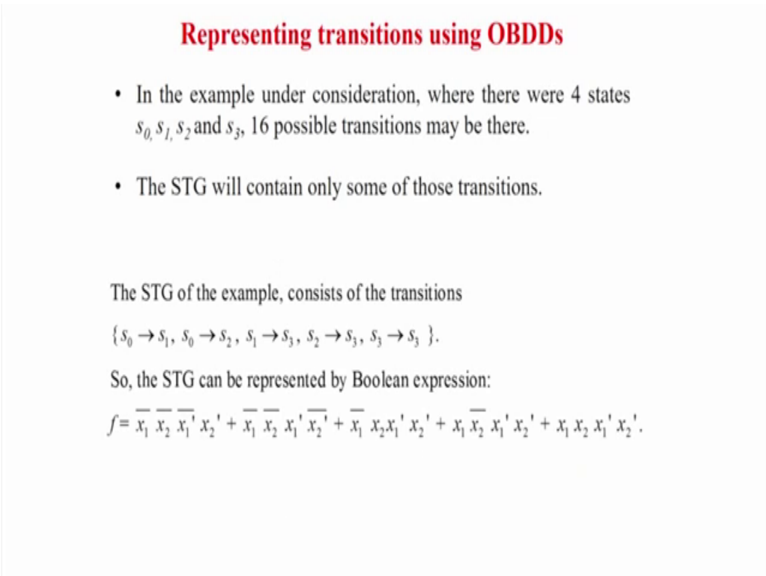
Basically quickly how transitions can be represented.

(Refer Slide Time: 02:11)

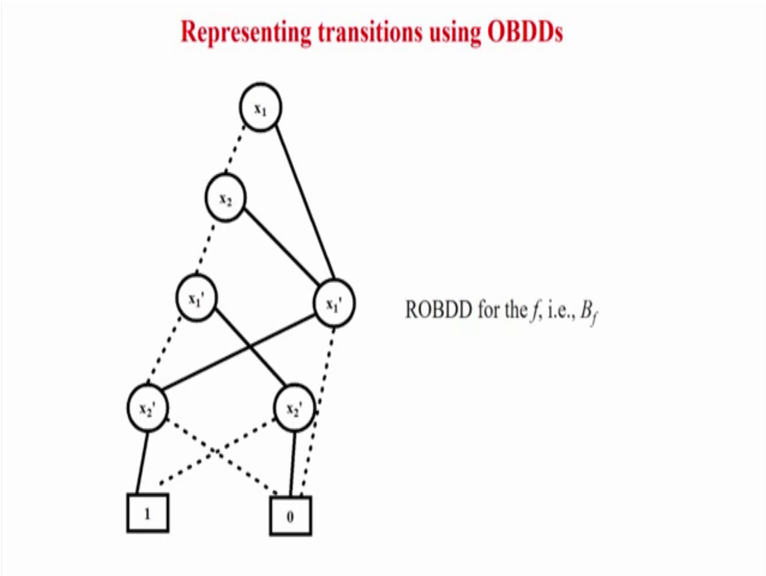


And then basically there is some kind of ROBDD, which represent the states substrates of these and the transition relation then what we had seen.

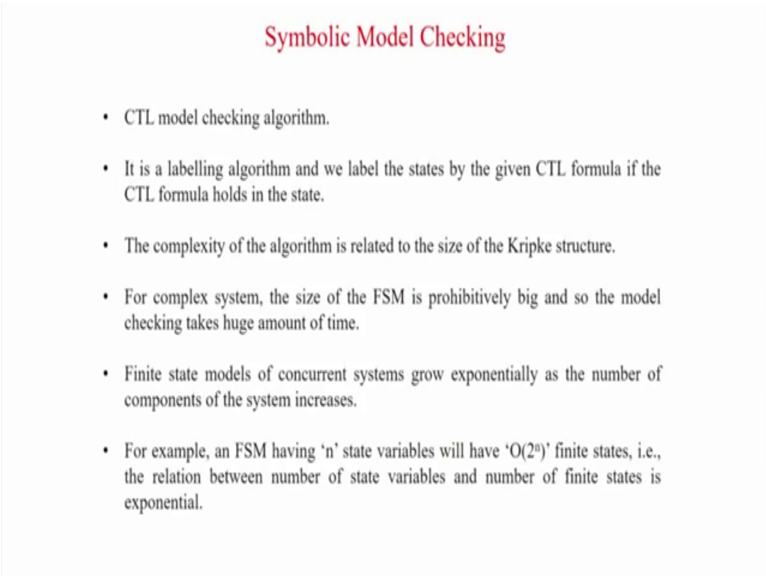
(Refer Slide Time: 02:11)



(Refer Slide Time: 02:13)



(Refer Slide Time: 02:19)



Then basically we had gone to the symbolic model checking in which case we have given you an idea, rather by taking a Kripke structure then this model basically we represented.

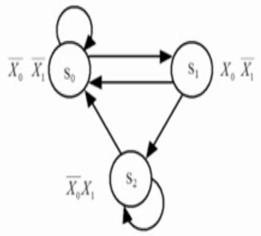
(Refer Slide Time: 02:24)

Symbolic Model Checking

- It is known widely that the state space explosion problem in automatic verification has limited its use to small systems.
- To make the model checking more effective, we need some methods to content the state space explosion problem.
- One possible way is to represent the state space of the FSM in a compact way so that the model checking becomes effective
- With the help of ROBDD we can represent the Boolean expression in canonical form. Also we have seen that ROBDD can be used to represent the Finite State Transition system and the set of states of transition system.
- In model checking algorithm we use ROBDD to represent the Kripke structure and the method that we get is named as Symbolic Model Checking. It says that the states are not enumerated but represented symbolically as a set with the help of ROBDDs

(Refer Slide Time: 02:25)

Symbolic Model Checking



Model checking using OBDDs is called symbolic model checking. The term emphasises that individual states are not represented; rather, sets of states are represented symbolically, namely, those which satisfy the formula being checked.

Given a system, we must assign unique Boolean vectors to each state before we construct OBDD. The easiest way to do this is to assign a Boolean vector based on the propositional atoms true in that state.

(Refer Slide Time: 02:30)

Symbolic Model Checking

- Generally, in model checking algorithm, the Kripke structure M and a CTL formula Φ are given as input and the output of the algorithm is the set of states of the model which satisfy Φ.
- While checking for Φ, we know that the model must be labelled by the sub-formulas of Φ. Also it is clear that the primitive sub-formulas are the atomic propositions.
- From the labelling function of the Kripke structure, we know the states in which an atomic proposition p is true. This set of states can also be represented by an OBDD.
- The basis of model checking algorithm is to find out the previous state of a given state or a set of states.
- Consider the labelling algorithm of EX Φ. Label any state with EXΦ if one of its successors is labelled with Φ. While going to label the states with EX Φ, we must label the states of the Kripke structure where Φ is true.
- Therefore, we know the states where Φ true, and this set of states can be represented by OBDD. The job of model checking algorithm for EX it to find the predecessor states of this state.

(Refer Slide Time: 02:31)

Symbolic Model Checking: EX

$\text{Pre}_{\exists}(X)$ **:** takes a subset X of states (of OBDD) and returns the set of states which can make a transition into X.

$\text{Pre}_{\forall}(X)$ **:** takes a subset X of states (of OBDD) and returns the set of states which can make a transition *only* into X.

How to Find $\text{Pre}_{\exists}(X)$:

Given,

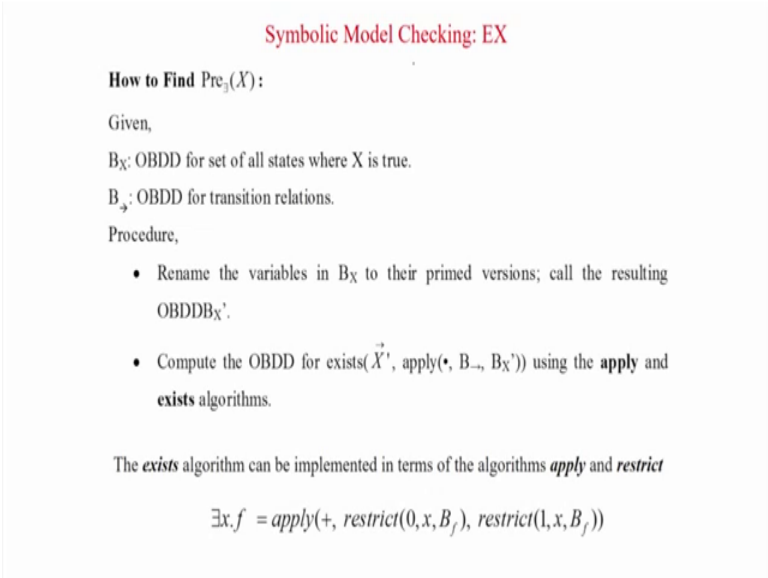
B_X: OBDD for set of all states where X is true.

$B_{\rightarrow}$: OBDD for transition relations.

Procedure,

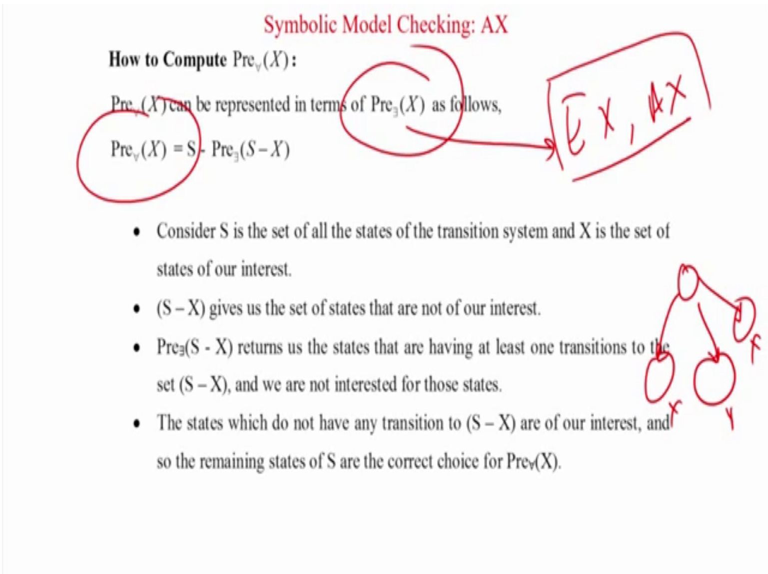
- Rename the variables in B_X to their primed versions; call the resulting OBDDB_X'.
- Compute the OBDD for exists($\vec{X}'$, apply($\cdot$, $B_{\rightarrow}$, B_X')) using the **apply** and **exists** algorithms.

(Refer Slide Time: 02:32)



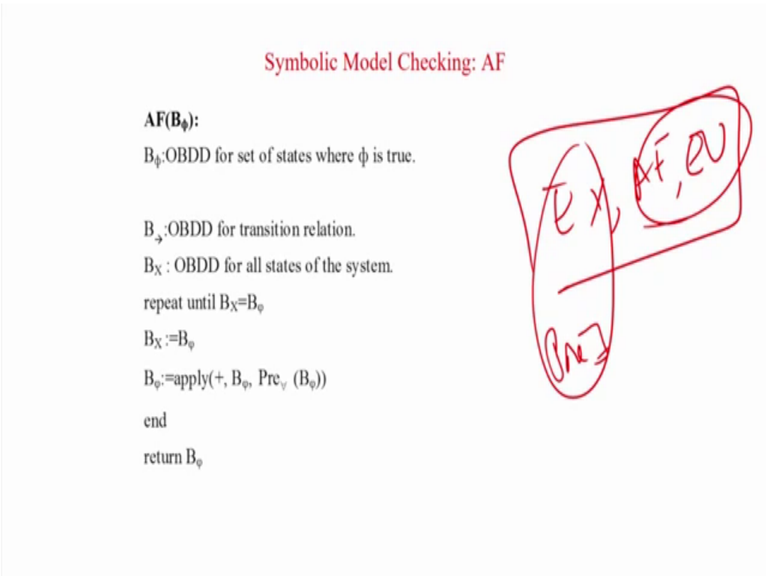
In terms of basically the BDDs and then we tried you have shown elaborately in the last class, that how symbolic model checking can be done using such a Kripke structure which are symbolic and we are very elaborately seen two formulas temporal formulas that is E X that is there exists a by using this model checking algorithm and also we have seen basically another one that is the A X.

(Refer Slide Time: 02:59)



That is from a given state in all paths in future the formula should hold. So, basically till now we have seen E X and A F sorry E X and A X the two formulas, how we can do our model checking that was covered in lecture this is last lecture in lecture number 4. Now, as you know they are basically we are also shown that by given any model any Kripke structure we do not explicitly model them in terms of states, rather we model them in terms of BDDs and then we have seen how the labeling algorithm can be done symbolically itself on in those BDDs, using this we can do labeling symbolically on the BDD itself rather than the state space rather than the state based Kripke structure and as an example we have seen two formulas that is E X and A X.

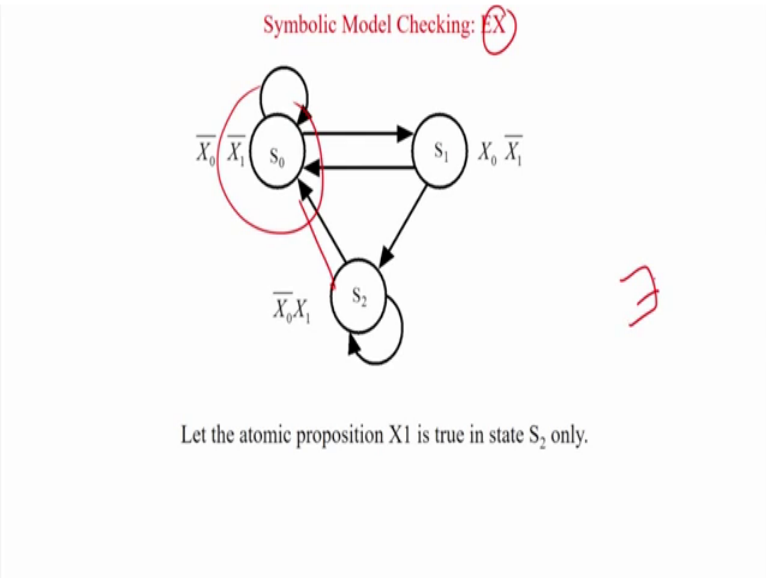
(Refer Slide Time: 03:46)



Now, basically we are going to see we already know that we know that E X AF that is the first lecture and eu. So, in the first lecture if you re recollect on verification we told you that basically the three basic formulas E X AF and eu. So, within this three formulas basically all the other CTL formulas can be modeled or you can write other CTL formulas in terms of these formulas and so therefore we can call them as the basic or building blocks or if or we can say that if we know symbolically how to model check for these formulas in any other any x complex formula can also be written and model checked using symbolic model checking.

So, E X already we have discussed last day E X means just before, she see basically E X is nothing but what we mean to check is that from any state say this one in the next state that is because of x some formulas would not hold.

(Refer Slide Time: 04:30)



This is nothing but your existential quantifier that is if you look at last time we have done this one. So, we generally call it as existential quantifier this 1. So pre of x, so pre of x is nothing but from any state x means what maybe then this some states in terms of x from where basically there exists a path, where this is this states some state x over the formula intended to be true and means there should be a state from which there should be at least one path leading to a state in the next.

So, this may be a path and there should be at least one path to a state x where this formula in x will hold, then we can label it as existential of x. So, this is nothing but you can call it as E of X of some formula X. Now why is it because from this state there exists at least one path to a next state where the formula corresponding to this capital x holds basically. So, that is the idea so E X means what I mean to say is that, E X means basically directly we can have basically that their existential a pre of existential x is directly mapped to the formula of EX, that is all states from which there is a path and in the next stage basically you should call it the path this is basically the edge.

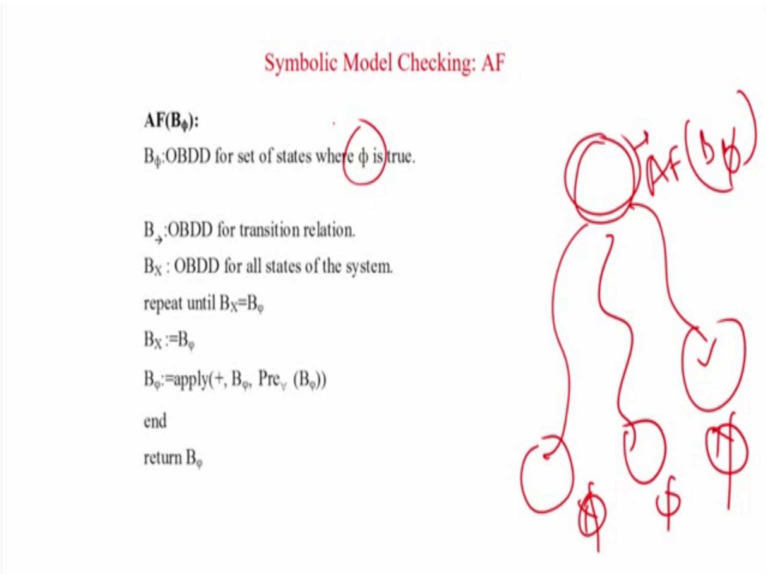
So, from a state like this the edge and the state here the property basically which is required to be true has to hold there should be existing a path. So, that is E X and basically the formula that is pre their existential x actually caters to that, now as I told you that is what you are going to do today basically that 3 formulas are building formulas E X AF and eu.

So, E X we can directly say that that is pre of pre existential actually caters to this and now in this today's lecture we will elaborately discuss or how we can go for model checking for formulas AF and E U using symbolic model checking so and for this will be very elaborately using these two functions.

So, you have two things already we have seen that is one is pre of x and one is for all existential pre; that means, from this state their exist at least a path where x is holds and in case of for all that is a pre for all clause, then for all paths from this state your property of a whatever property you want to satisfy in x should basically hold so and you already have discussed in lecture 4 that is last lecture elaborately how we can model check for this pre all and pre x.

Now, because as a as already I told you pre x is directly iterate to ex, so we dropped the detail lecture on this because E X means directly you can apply pre of x existential x and your job will be done.

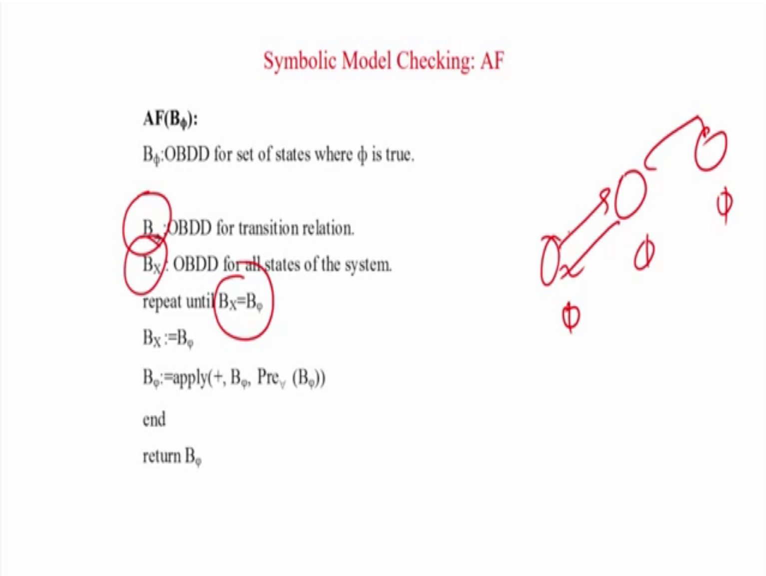
(Refer Slide Time: 07:14)



But the two other formulas is AF and E U equals elaborate discussion that how can we model check on them. So, let us start with AF, so what is basically AF means there should be a state say and then all paths from either state in future some property x should hold, then only we can say that AF of x is true that is for example, it is saying that AF of phi.

So, it is saying that that is this says that this relate in future this some phi formula is true in some states, then only we can say that AF of phi or phi actually is satisfied by this state; that means, from one state if you take all paths or if you take any path and in future in a state the formula phi basically holds true that is actually AF formula mean; Now, how to go for a symbolic model checking using this one.

(Refer Slide Time: 08:09)



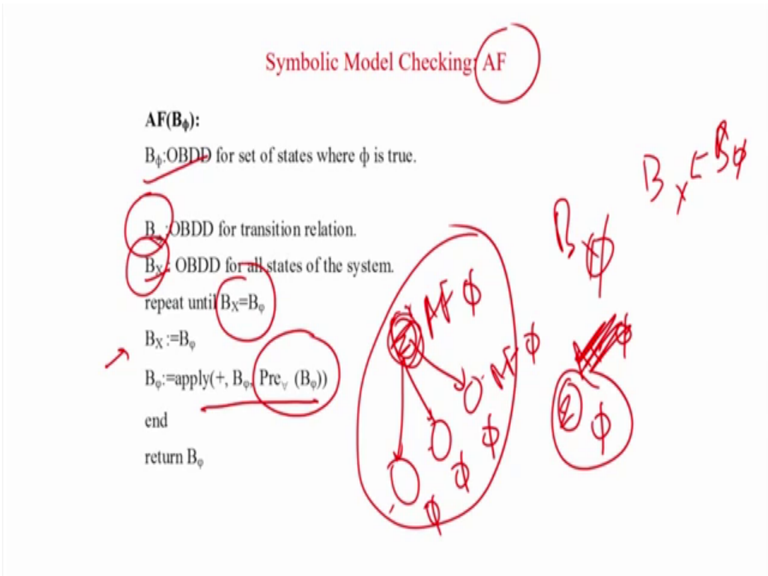
So, this is the algorithm, so B of B this B with an arrow sign basically is the BDD representing the transition relations, already we have seen in the last lecture in elaboration that how basically OBDD can be used to represent the all transitions of a of a state machine on the bookie automata in this case sorry, the Kripke structure in this case and then B of x is the BDD for all states of the system.

So, initially you will say that you just have a BDD which represent all the states of the system just an initialization, then what we do we repeat till B of x is equal to B of phi. So, B of phi basically is the OBDD of states, where the formula phi is equal to 2 that is very important for us Bx is some arbitrary initialization. So, initialize initially what they have done they have taken all the states of the system and they have made basically a BDD for that.

So, initially what we do we repeat till B of x equal to B of phi. So, as of now basically Bx can of course, if it happens that this phi is true everywhere. So, in a very case it may happen that there are three states in the system, exact say for example and everywhere the phi is equal to true, then we should not do anything actually we can stop the algorithm there itself because, in all states your property is true. So, in all from all states of all paths in future phi is true. So, of course if in this state phi is true in this state phi is true and the in this state also phi is true.

So, of course from that state all paths basically in future phi will be equal to 2, because a present is also considered as future. So, I mean that is actually true kind of a thing that without even starting from that state, your formula is holding true at that state itself and if it holds for all the state. So, far or of the Kripke structure or your system model, then of course you need not do anything you can stop there itself.

(Refer Slide Time: 09:59)



So, what we do initially B of x you assign all these states of the system and then you repeat until B of x equal to true; that means, if it happens that in all these they are the defect case that all the states have the formula to stop there itself.

Else what you do you assign Bx equal to B of phi, now in this state basically your Bx will now not have all the states of the system. Now it will have only the states of the system where phi is true here actually it is a, now we have become a temporary buffer what do is wherever states actually phi is true that now Bx is holding as a temporary buffer, now B phi will be updated.

So, what we are now having Bx is now holding the all the states phi where at the present iteration phi is equal to true, so now B is a temporary buffer.

Next we will update B of phi, so how do you update, if you recall in the first lecture we have shown how AF is levelized. So, what we do basically we say that any state here when phi is true basically we have to mark as AF phi right. So, any state where phi is true is AF phi AF phi is true, because present is future so in that state phi is true. So, of course in all paths from the state phi is in future is going to be true because, if this is the state where phi is true. So, any path if you take obviously phi is going to be true in future because, present in future that is before starting itself your formula is satisfied.

So, you have to mark it next what you do next basically now the this is done then you have to take you have to see this form this formula or this way of calculating symbolically, because pre of all that is pre existential and pre for all. So, these are the 2 basic structures which will be used for symbolic model checking. So, already we have discussed in the last lecture how pre of existential and pre of all is calculated. So, what did we saying pre of all B phi. So, what it means that means, there may be say some 3 or 4 states in way in which case say this is phi is true this is phi is true and this is where phi is true and this one state where these I have only 3 pages no other is available.

So, from this state basically you know that all the paths these two states where phi is equal to true. So, you can say that here basically this formula is satisfying over here. So, now what did he say this for now you say that you take all states where B phi is true like this. So, in these states basically obviously your AF is AF phi equal to true because, phi is true in their state now you take all such states also. So, all these states will be there you take all by this formula it is same and apply you can see B of phi pre is something like this, where basically this state is such like there is one edge from each on this from the and there are three edges for example all the possible edges no more it exists and in that is phi is equal to true.

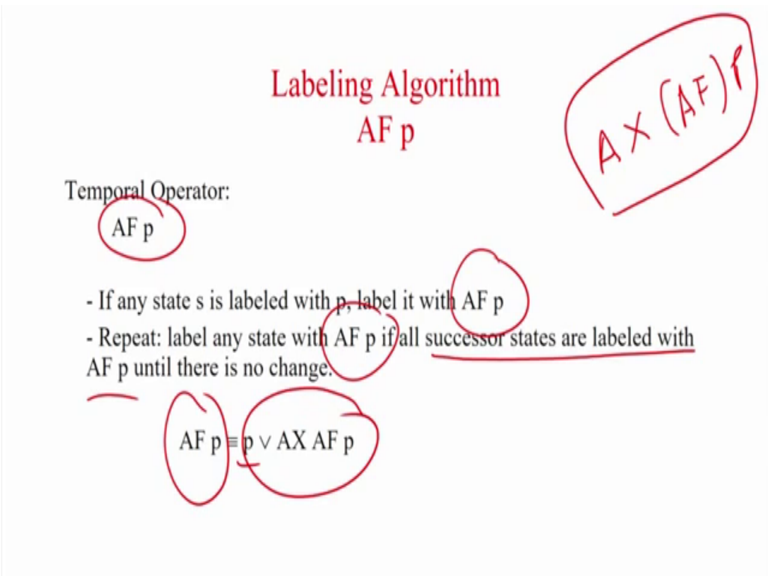
So, here also B of phi that means, A of phi has to be true and then you say that you take states like all states, where A of phi is equal to true sorry you take all states where phi is equal to true and all such cases like this when basically next a state phi is equal to true for all edges and you take this union of this and union of this and you are going to apply this one; that means, this one actually what it is doing it is taking all such states where phi is equal to true and this is pre of all B phi is taking states of this nature and you or them basically these are the states in which AF phi holds and you have to keep on doing it till you are not going to give any more states, then basically what you are getting you are getting AF of phi.

So, that is the very simple idea that is first you label all the states where phi is true, that is where AF true is true as well as you have to also consider this one in iteration that is take a state where all the next states phi is true, for all the edges that you also mark as a AF phi take or of this and this is one iteration that you keep on doing it and then what and whenever your algorithm will stabilize you are going to get basically AF of phi and we have to do it symbolically.

Of course, this can be done symbolically because if you see already we know that this one can be done symbolically already we have discussed and apply the or operation or operation can also be done using a BDD.

So, this is the symbolic model checking algorithm for A of F just recollecting what I have told using a diagram.

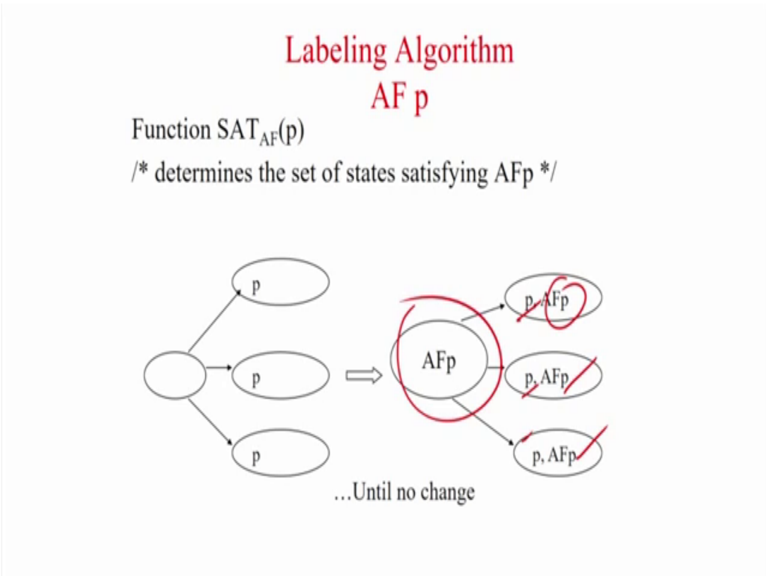
(Refer Slide Time: 14:18)



So, what is AF of p or phi in our case? So, if any state S is labeled with p label it with AF p that is true as I have already told you and then you have to repeat it in a loop label any state with AF p, if all it is successor states are labeled with AF p under there is no change. So, whatever I told you diagrammatically is written in this slide that is AF p is formula is p, either p is true in that state or in all states from x that is all states in future in all states in that is Ax AF means or for all states there exists a path such that in all states in future p is equal to true, that is this one corresponds to the case the states where phi is true and this part of the formula corresponds to the fact of this 1; that is there is a state from which in all paths in future phi is going to be true in the next state that is this one correct.

So, as I told you this one already we have discuss in the initial lecture, so you can just recollect it so this is the formula and just pictorially representing.

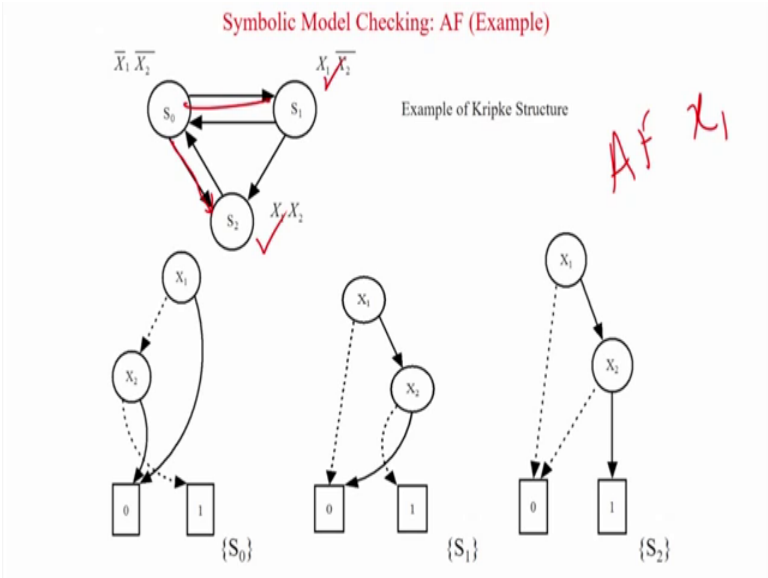
(Refer Slide Time: 15:23)



So, these are the states in which p is equal to true. So, of course, you have to mark them AF p and AF p by default they are going to be marked and then basically as this all these states are such that p is 1 is equal to 2 over here you also make it as AF p. So, that is 1 way of or the way of actually labializing the for labializing the states for AF p. So, this already I have shown you this pictorial diagram basically you already we have shown you in this first lecture, but I am just repeating you.

So, that you can get a recollect the means recollect the lecture, but this is not symbolic so this part actually is a state based representation.

(Refer Slide Time: 16:03)



But now we are going to do it symbolically, so this one was a symbolic manner that is use using BDDs oring them pre calculation that is a symbolic way of doing it, but this example I have told you is a non symbolic way of doing it, but just to read mean correlate the fact I have shown these two slides.

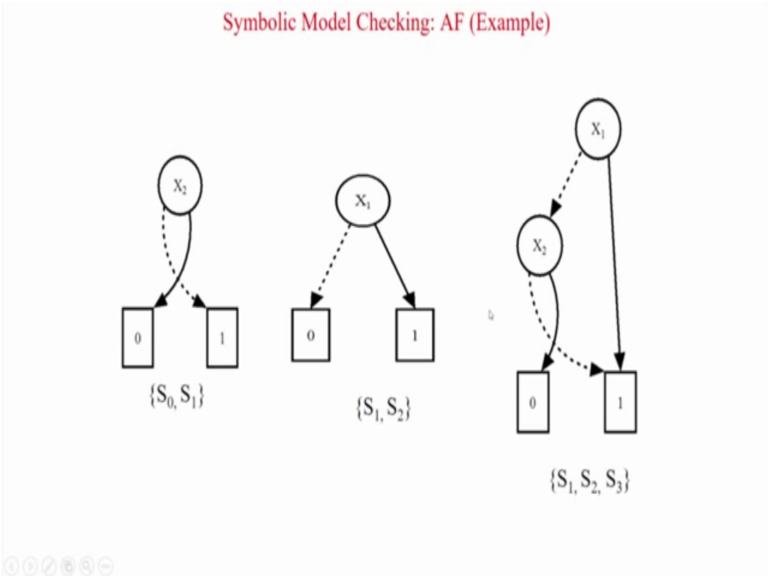
But now again let us come back to our real business which is on AF and we are going to do in a symbolic manner. So, this is an example of a Kripke structure I am zooming it for you, you have to always keep this structure in mind because throughout today's lecture we are going to use this example heavily.

So, if there are three states in this Kripke structure and then here X1 is false X2 is false S1 X1 is true X2 is false and in x S2 both X1 and X2 are true correct. So, now symbolically as I have already discussed in details I am not going to again revise that again your last class we have done.

So, this one is the BDD corresponding to the state S naught, this 1 is the BDD corresponding to S1 and this 1 is the BDD corresponding to S2, you can easily find it out just X1 X2 means both are true. So, then only it will be true, so if both of them are true that is why you getting the answer as 1.

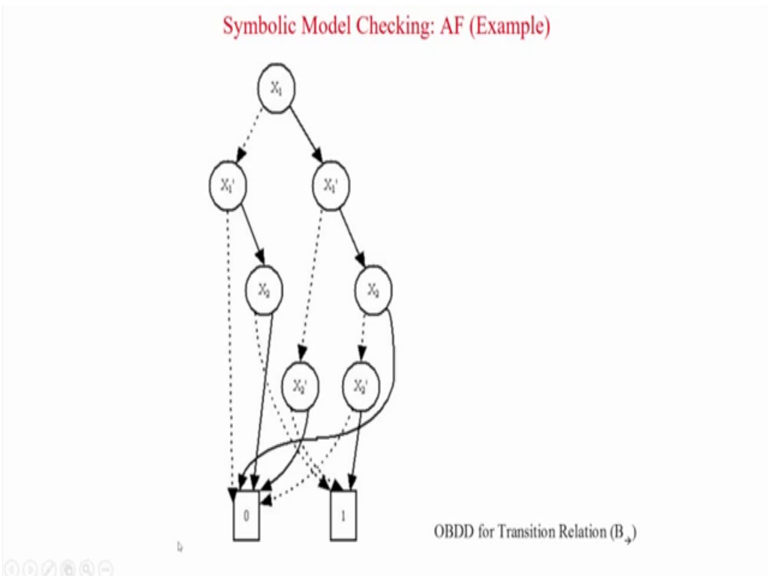
If any one of them is a false the answer is a 0 that take S naught, similarly or others can be easily explained last class we have done this elaborately.

(Refer Slide Time: 17:21)



Similarly, S1 S2 is the BDD for this set BDD S1 S2 for is the BDD for this and all these states if you take together S1 S2 and S3 this is the BDD; all other combinations can be drawn, but I have not done it here because basically for the example we are going to consider in this case these combinations will survive and this is the transition relation.

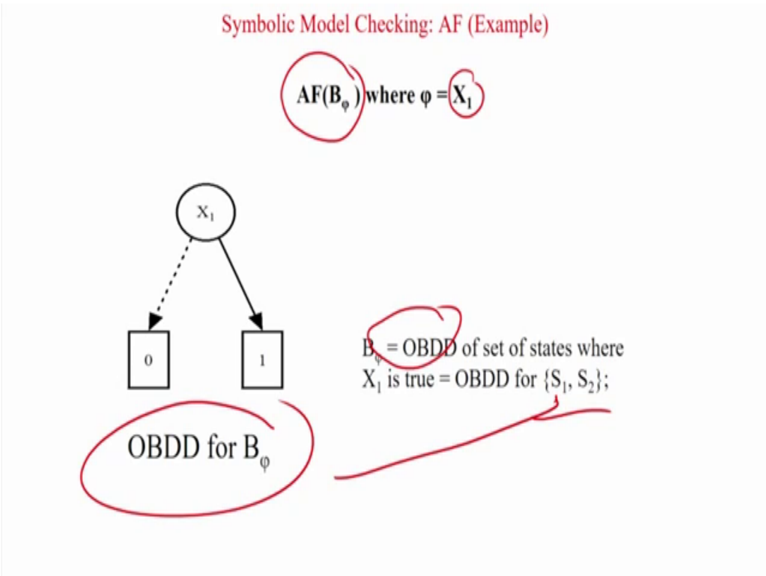
(Refer Slide Time: 17:39)



So, transition relation also we have found out how to deal with, so that we also put some prime states like there are the next states. So, these are the next states you can see say along with the normal states, we also have the prime states and we can easily model the transition relations of a Kripke structure of any in that manner any state machine in a symbolic manner or using a BDD.

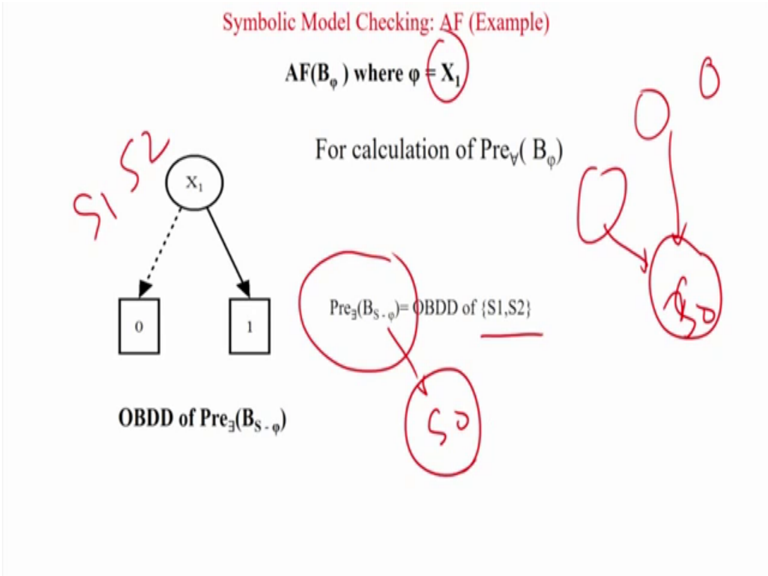
So, that also you have discuss in the last class, now for this example sorry this example we are considering this example we are considering. So, this is the set of BDDs for states and this is the BDD for the transition relation. Now, you can easily hand draw them again and to see if the whatever we are telling is correct and that also we will give you a confidence of drawing the BDDs for giving different state diagrams now coming to real business.

(Refer Slide Time: 18:24)

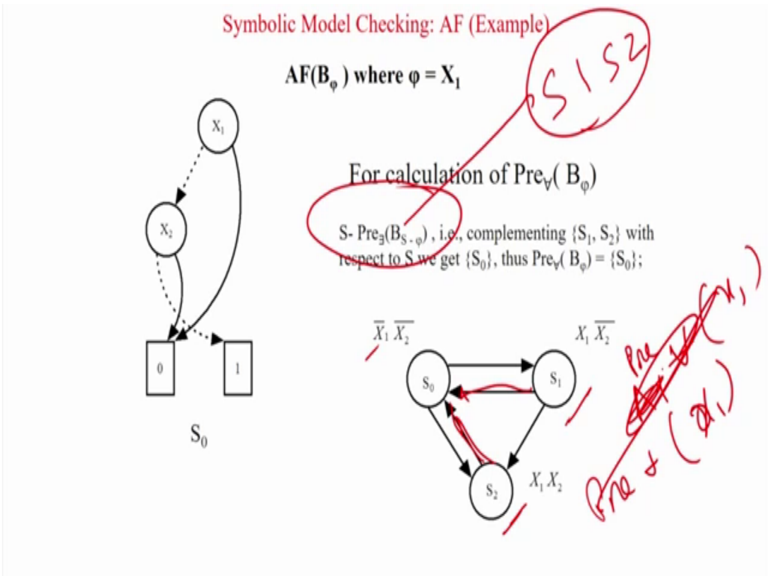


Say for example, we are going to verify all paths in future B of phi, so what is phi of phi here this is basically your variable called X1 we are saying all paths in future let us again go back to the structure it will be easy for us seeing it ha.

(Refer Slide Time: 18:37)



(Refer Slide Time: 18:38)



What you are going to say all paths in future X1 is going to be true that is what you are going to verify by labeling that is from the states, first of all paths in future X1 should be equal to true. So, if you can see over here before doing it symbolically let us take a what do I see Layman's approach to see what is going to be the output.

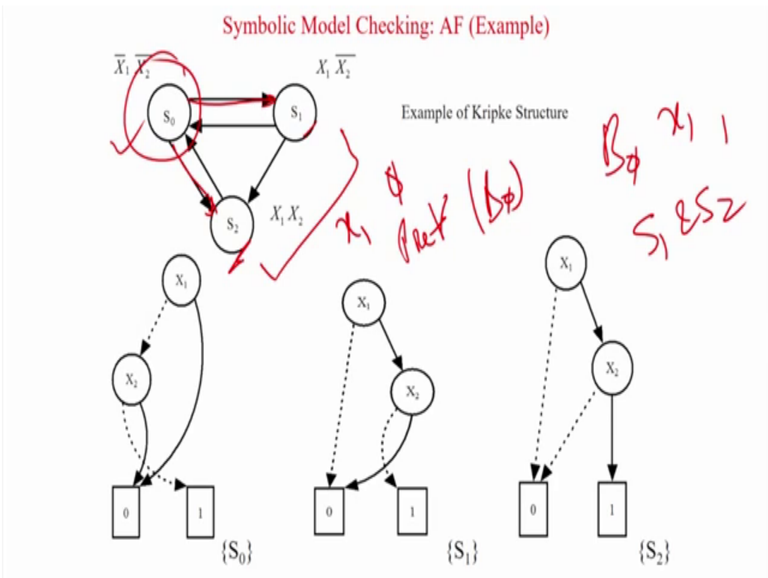
So, if you look at this thing if you look at this structure is very obvious that from all states this formula is going to hold, the formula is all paths in future from a state all paths in future X1 is going to be true. If you can see from state X1 if you see this is the path X1 is going to be true this is the path again X1 is going to be true.

So, from S naught all the paths in future will give you X1 true, if you take S2 so there is 1 path here this is this path is this and the 2 hop path where X1 is going to be true that is for S1 and there is there is only 1 path from S1 out going from S1. So, if you take S2 S0 S1 X1 is going to be true.

So, S2 will also satisfy this formula and S1 these states itself X1 is going to be true. So, we can see that in all paths in future so from S1 X1 is true. So, by default it is true from S2 if you see here also X1 is equal to true default it holds and also you can see that this one is also if you take this path also X1 is going to be true in future any one you can take and from S naught in the present state X1 is not equal to true, but we requiring all path in future X1 is going to be true.

So, if you can see this is one path where X1 is going to be true there is one path where X1 is going to be true in the next state. So, of course by a simple intuition you can find out that in this Kripke structure in all paths in future X1 is going to be true right and within which states X1 is going is true that is S1 and S2.

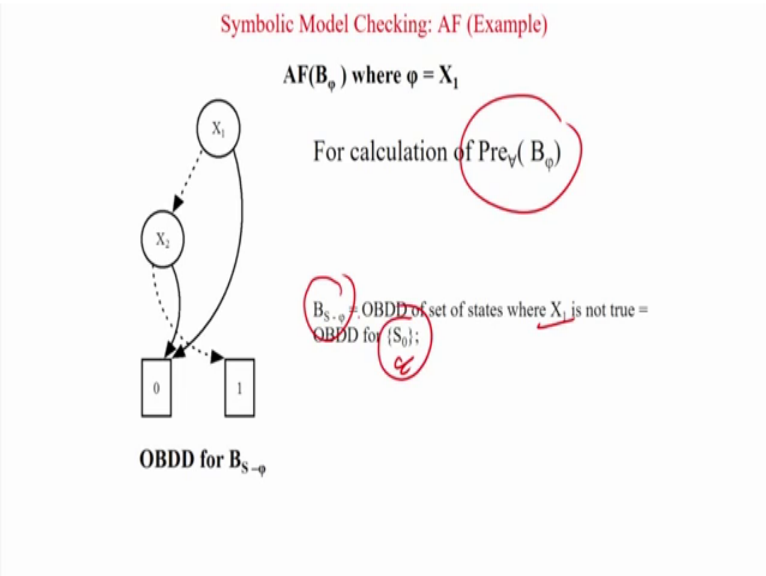
(Refer Slide Time: 20:31)



That is B of phi that is X1 is true in the 2 states that is S1 and S2 correct. So, now we are going to inherently we have in our mind that all paths in future X1 is satisfied by all the states of that Kripke structure, but let us do it symbolically.

So, first what we require the first step we require a set of all states that is B of phi where the X1 is true. So, already we have shown you that this a X1 X2 states S1 and S2. So, you have to first draw the BDD for S1 and S2 because, if this X1 and X2 X1 is true in the states S1 and S2. So, this is the BDD for B of phi that is actually the BDD corresponding to S1 and S2, was you already discussed you can easily find that that this represents the BDD for this state subset S1 and S2.

(Refer Slide Time: 21:21)



Next what we have seen next you have to do the calculation of pre, we are just going by this algorithm if you look at you are just going by this algorithm stepwise we are going by this algorithm.

We have to first calculate the transition relation is anyway there B of phi we are calculating that in set of all the states, where phi is true then we are going to finally go for this apply or B of phi that is in this case S1 and S2 and we have to go for pre of all B of phi, that is next you have to find out that is pre of all B phi.

So, in this case S1 and S2 so we have to find out basically some states such that all paths from that states leads to S1 and S2, these are the states called these are the two states where your X1 is true that is your phi is true. Now, I will be find out such states from where all paths will lead to either S1 and S2 that is nothing but pre of all B of phi.

Now, if you look at state S naught obvious this path and this path. So, all paths leads to the states where X1 is equal to true, but if you see in the next state means for all means in the next not in future basically next state it is true. So, S1 obvious S0 obviously, satisfies because there are 2 paths from S naught. So, 1 is leading to S1 11 is leading to S2 and in both they of both of them your X1 is basically equal to true, but if you look at S2 ok. So, in a X2 basically if you see the next state is S naught where phi is equal to false.

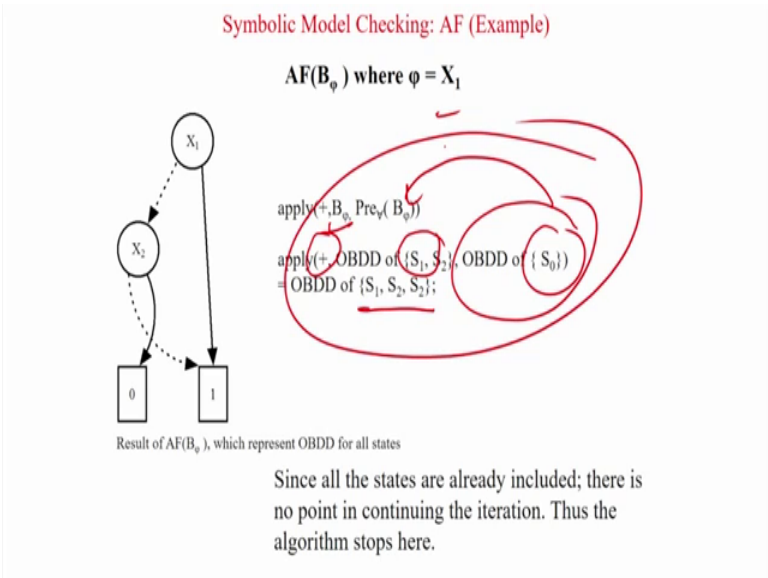
So, that is not going to satisfy and similarly for S1 also there is S1 path to going to S naught. So, in the next state phi is not equal to true or phi that is where X1 not equal to true. So, pre of B phi is only going to be this state S naught, that means from state S naught all the transitions if you take they are going to lead to states next states from S naught where X1 is equal to true. So, basically so that is equal to that will be going to be the answer for pre of B of phi, but if you recollect of all B phi we actually do not go in a direct manner we go for in an indirect manner.

First we go for B of S minus phi that is the set of all states where X1 is not equal to true that is equal to BS naught. So, last lecture if you recollect we have where we are showing the algorithm, where you have showed the algorithm for pre of all we do not directly hand calculate as we have seen from these on the Kripke structure, whether we are taking a negated way of doing it first we find out all these states where phi is equal to false and then we will try to find out all the states from where there is at least 1 transition through the states where phi is false and then we will subtract it from the state space entire state space to get the answer.

So, anyway going by steps so B of S minus phi that is the set of all states where X1 is not true that is equal to nothing but S naught. So, this is the BDD for S naught now we are going to find out the set of states. So, we are having a states called x S naught right that is the last 1 S naught where your formula is false. Now you have to find out any state where this exists at least 1 path to S naught, so if there is at least 1 path to S naught; that means, all paths from these states these 2 states are not leading to states where X1 is equal to true. So, that we are going to so that will not be considered.

So, as we are going to take in a negative way, so at least there are the 2 states for example which are at least 1 path to S naught. So, they have to be negated from the answer. So, if you calculate if you look at the Kripke structure here so and what we will see that basically S naught is the statement is false. So, if you just look at here so S naught in the state where X1 is equal to false. Now, all states which are at least 1 transition to S naught will actually not satisfy the has to be computed now. So, from S1 you can see the at least 1 path to S naught and from S2 also there is at least 1 path to S naught.

(Refer Slide Time: 25:13)

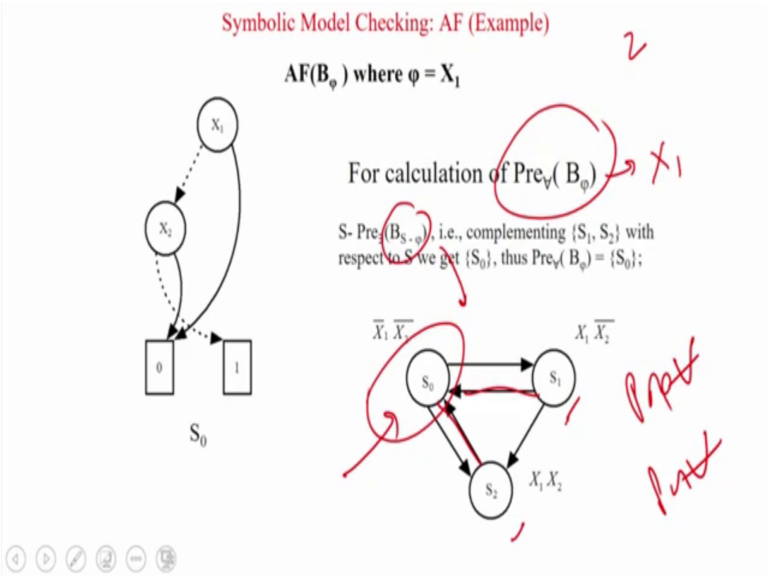


So, what does it mean basically if you compute there exists a path this 1 is basically now but S naught. So, what it is saying pre of S naught means there will be 2 states S1 and S2, now what that indirectly mean that indirectly mean that so this is the B naught this is the BDD for S1 and S2 because, S1 and S2 are the two states from which there is at least 1 transition which these 2S naught and S naught in this case where your formula phi is equal to false.

So, now what you are going to do now we are simply going to say S minus pre of Bs minus phi; that means, this is the state these are the states S1 and S2. So, these are the two state S1 and S2 means, there is a path from this S1 and S2 which leads to state S naught where your formula phi is false; that means, obviously these two states cannot satisfy the formula of all paths pre for all pre for all of your phi that is X1, because there will be at least 1 transition in this case to S naught when it is false.

So, these two states will be not satisfying your clause of for all phi pre for all phi x x1, so this will be eliminating. So, now basically what we do we will now subtract this S1 and S2 basically from S naught and you are going to get the solution answer as S naught.

(Refer Slide Time: 26:43)

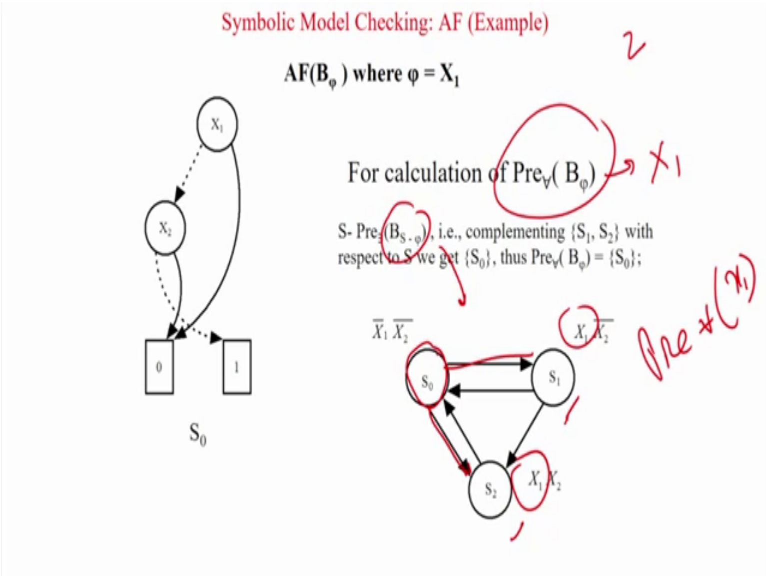


So, basically pre of all B phi that is equal to pre all B phi that is equal to X1 we are going to only have this thing that is actually S0. So, now intuitively what we have done first we have taken the set of states where this is false, so X1 is false. So, that is nothing but your S naught that is the AF that is equal to this.

Next what you are trying to do next to try trying to find out all these states like this 1, which have at least 1 transition to the states where B 1 is false. So, in this case S1 and S2 will be there because there is 1 S2 S naught that directly implies that for all existence that is for all pre cannot be satisfy that S1 and S2 because, they are having at least 1 transition to a state where the value is false.

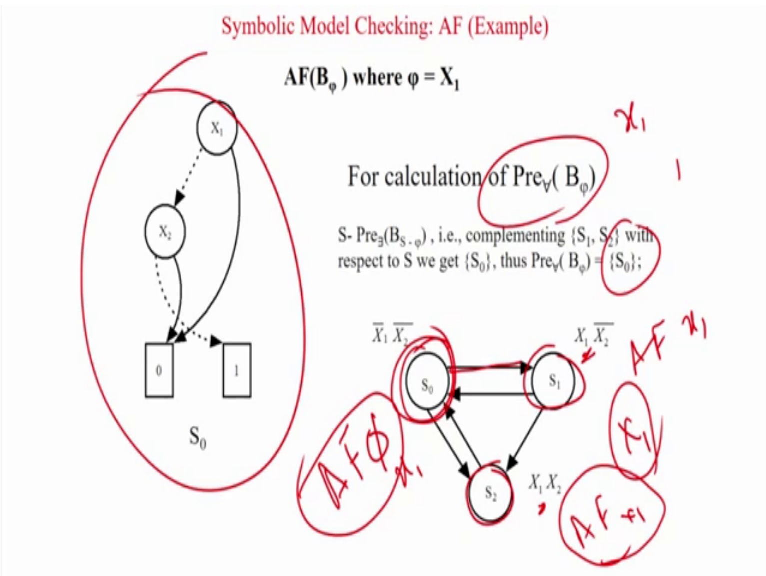
So, if you subtract S1 and S2 from the entire state space, we are we are getting the value S naught a state S 0 that is actually satisfy the case that is the state from where the next state phi is equal to true, that is in this state the property which holds true is basically for all true basically; that is from this state if you take transition all transitions from this state, they need to transitions lead to states where X1 is going to be true.

(Refer Slide Time: 27:46)



So, pre for all X1 is basically satisfied only in this state S naught. So, this is how we actually symbolically calculate the value of pre for all X1 that is B phi. So, that is actually now we are in the case where we have got symbolically the value S naught that is this BDD which corresponds to pre of all B phi that is equal to x1.

(Refer Slide Time: 28:16)



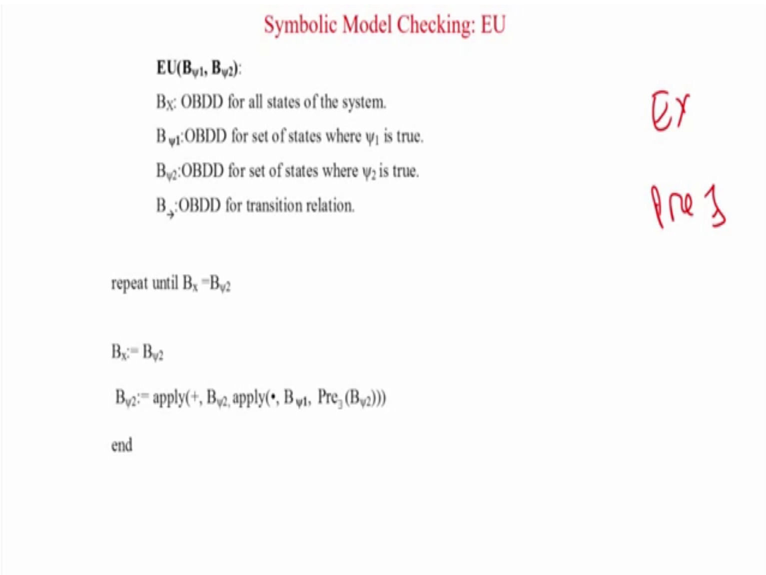
So, now what we have at hand, so now we have at hand the symbolic representation for this set of states such that, in the next states that is this some states like this where in the next state your formula X1 or your property X1 is true. So, this in this example it is S naught, but in the general case you will have a set of subsets of states, which will be represented by basically your a binary decision diagram that will be done symbolically.

So, now we have a set of states like this where in the next state this property is true in all the paths. So, you can very easily say that all paths in future phi is definitely going to be true over here in this case X1, because next is also a future so this 1 will be basically labeled. Now we have to also label these states where X1 is true by default itself like in state S1 and S2 X1 is true here also X1 is true.

So, in this states obviously AF X1 is true here also AF X1 is true, now why it is true because present is also future. So, in this state X1 is true itself so obviously, even before I start from this path or in all paths in future x is going to be true ok, so that is what is the case.

So, now what they are going to do is that they are going to take the or of B of phi which what we have already gone. So, OBDD for S naught stands for this and OBDD S1 and S2 stands for the case, the states where itself X1 is going to be is true; basically this is nothing but these 2 states where themselves are 20 states. So, you take OBDD of S1 S2 OBDD of S1 and you go for apply operation apply is all, so we are going to get the answer S1 S2 and S3. So, which we have already shown you intuitively that in the Kripke structure for all paths with in future X1 is truly satisfied by all the 3 state S 0 S1 and S2, also we are getting the same answer symbolically.

(Refer Slide Time: 30:12)

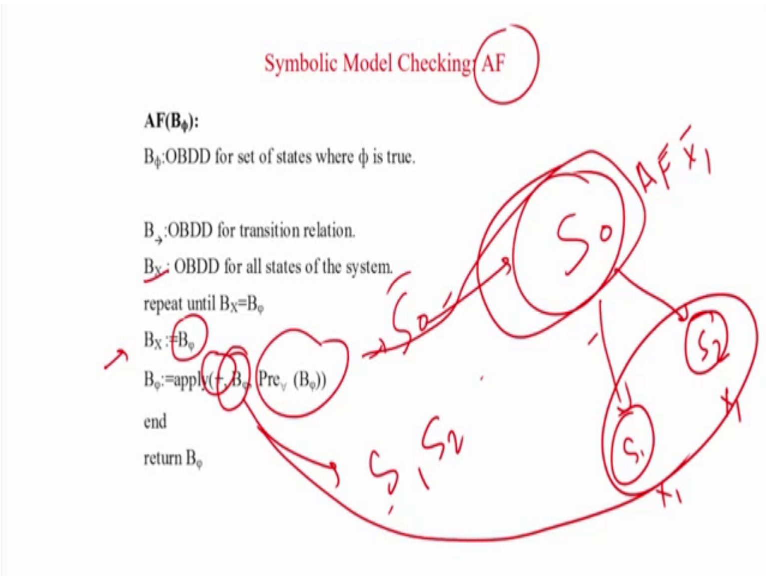


That is again this S1 and S2 are coming into this class because X1 is true in this state itself and the state S naught is coming, because all the paths from S naught next state if

X1 is true or X1 is true in that part. So, if we make an union of this we are going to get the answer which is S1 S2 and S S 3 sorry, basically S 0 S1 and S 0 S1 and S2 typing error. So, all these states will be basically included over here and all these states are coming as the answer, so no more repetition is required no more iterations are required if you stop and your model checking will tell you symbolically that yours formula AF X1 is true in all the states.

So, again just recollecting again by this algorithm what we have done this is where is the algorithm, so again just a quick recap. So, what we have done basically we represent it B of phi as a BDD and then basically this pre of all be calculated in an indirect manner, that is first we calculate we call compute phi minus that is phi negate it is false; then we find out existential pre and then we subtract whatever answer we get we subtract it from the whole state space and then we get the answer.

(Refer Slide Time: 31:33)

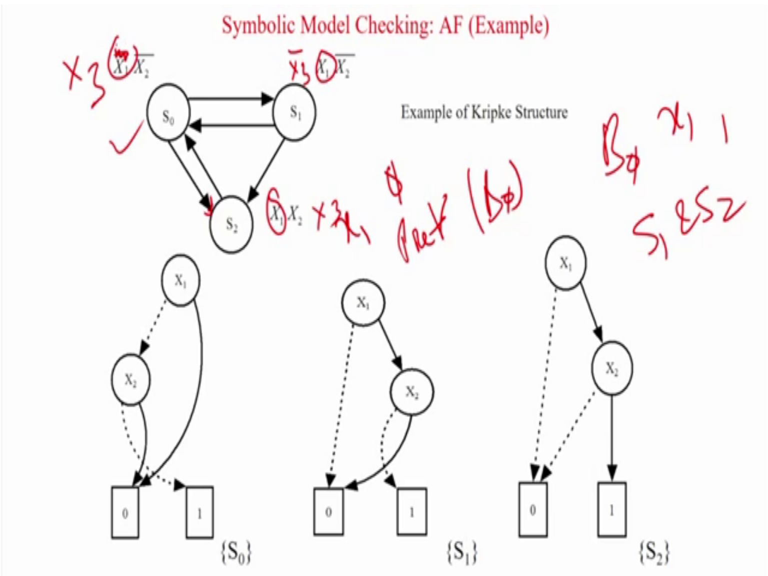


So, I think in this case we got the answer of S naught and basically and then what we have done then B of phi are the states, basically where the variable is the formula to be verified is true by explicitly itself explicitly true in the state itself. So, we got S1 and S2, so these are the two states like S1 and S2 where X1 was true itself and but for the state S naught which we had found out over here, these are state in which case in the next state the formula AF phi or a AF X1 is true. So, of course we are also labeling that Ax phi sorry AF phi we are labeling it as X1 here by the virtue of the next states having the value of x1.

So, this one is actually computing for this and then we are going for a or operation. So, or means so this one is going to I will give you this state and this B phi is going to give you states S1 and S2 and then finally we are going for an or operation and so we are also going to get the value S 0 S1 and S2 and basically why do we end over here then the iteration will stop over here because Bx will be equal to B phi the whole state space will come over itself and of course, if you would have a very straight case like something like if you would have something like just a stray example if you would have something like in this case if I again negate the bar.

So, any anyway I cannot this will not be holding this example properly.

(Refer Slide Time: 32:52)



So, if somehow I could have done a case like maybe I would have some x 3, I could have over here and then I would have eliminated this and I also x 3 here x 3 bar over here and maybe x 3 here I would have just put it and in your variable; I am just adding a dummy variable here to just give you the fact that this is the structure in which all these states X1 is going to be true, in such a case your algorithm basically will stop here itself because Bx will equal to become B phi.

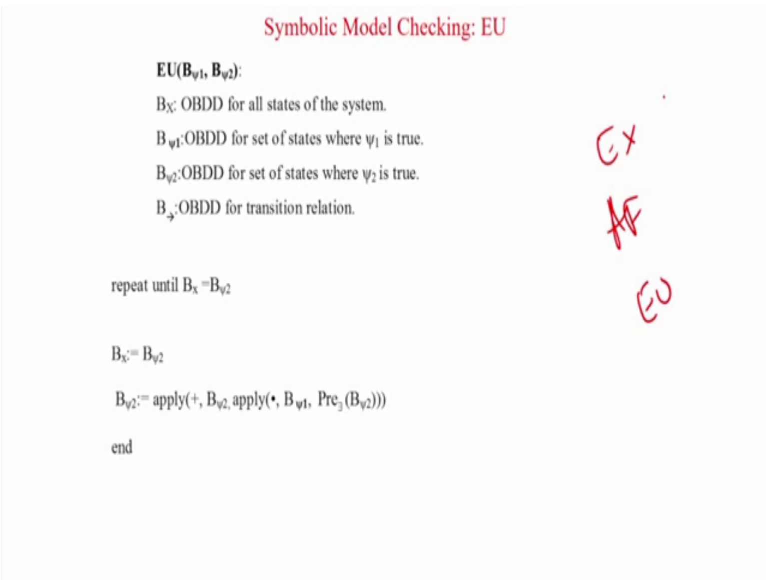
So, that is not very straight case in which case in all the states basically any of this system this formula X1 is true.

So, in that case the machine will stop or the algorithm will stop there itself and also in one more case as it happened here sorry, if you look at 1 thing also happened that after the end that B phi is after you and sent the answer by oring the set of states where phi is equal to true or in the next state phi is going to be true for all the parts then basically here we are finding out that all these states have to come into picture the S1 S2 and S3.

So, it will again become equal to the set of all states or in any iteration if you find out that all these state space of the states have been included. So, of course there is no point in eternity your answer stops and you are going to get the answer, that is all the states of the system satisfies the given formula.

So, what we have seen we have seen 1 very important formula for model checking in AF, how it can be handled symbolically we have given the algorithm and then we have explicitly analyzed using a very simple example and how it can be done symbolically. So, till now we have learned about E X AF, so E X as I again I am repeating E X is nothing but equal to pre of extension that is equal to E X because, it says that from a state there exists at least 1 path where in the next state it is true, that is nothing but pre of x; then using pre of x and a pre of all we have shown how can you model AF.

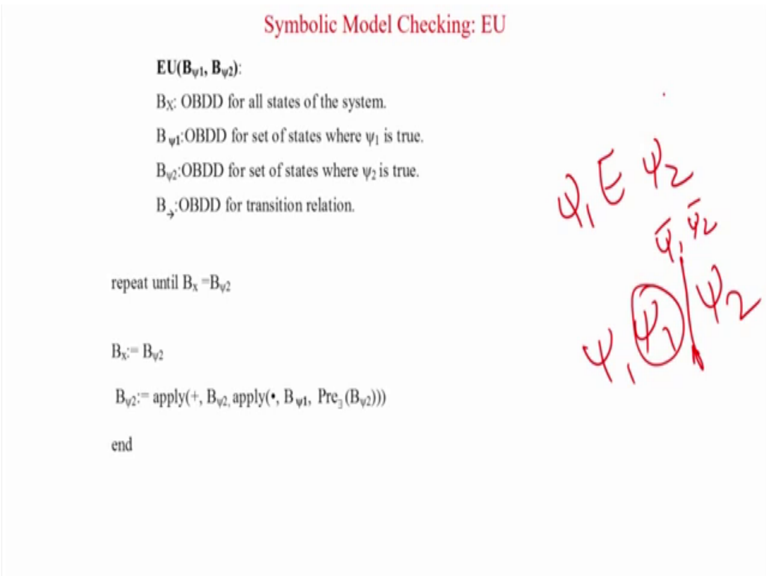
(Refer Slide Time: 35:01)



So AF we have learnt we actually already know from the last class, then what only more atomic version remains or I should not call atomic the a basic formula which remains to be explored is E U, with all this you can generate any other CTL formula.

So, now we will see how to compute E U that is existential until using this how we can go about modeling any other formula. So, we will now concentrate symbolically how to verify eu. So, what do you require here just similar approach quickly we can cover this the same way as we have done for AF, we have the set Bx is the set of all the states in this system, but there are two formulas because we know that means it is always only psi until psi 2 or psi until phi.

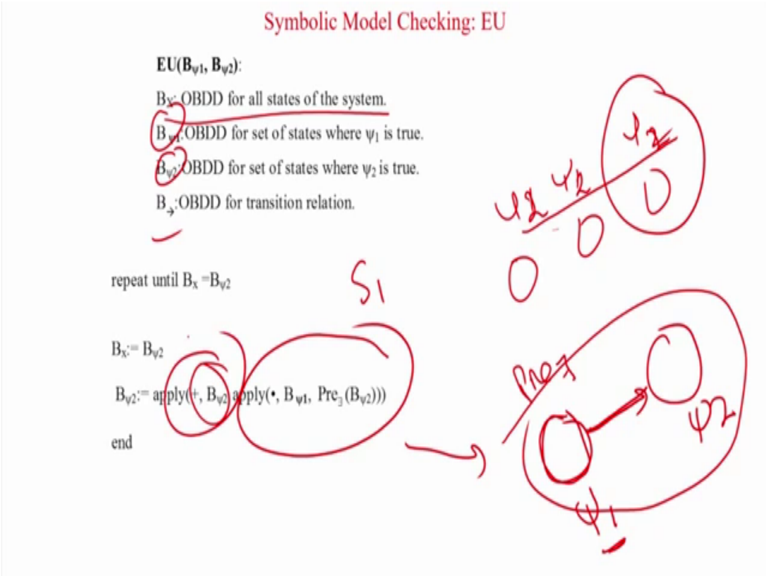
(Refer Slide Time: 35:46)



So, we have two formulas basically, so this 1 should hold until this that is what is the idea. So, if you forget basically your psi 1 should hold until your psi 2 should start holding, that is there should not be any state in between where psi 1 is also false and psi 2 not true; that means, you cannot have any state here where maybe psi 1 is bar is true and psi 2 bar is true that is in this state both of them are false, you cannot have anything like that otherwise the formula will be 4 false.

Basically psi 1 should keep on holding until psi 2 or in this case yeah psi 2 starts holding right. So, anyway so there will be 1 BDD for psi1, 1 BDD for psi 2 and this is your transition relation now what we have to do basically.

(Refer Slide Time: 36:24)



So, this is the initialization step, so initialization whatever will be the value of psi 2 formula will be damp to Bx that is the temporary storage as we have already done and you have to repeat till the bf of bf is equal to psi 2. So, of course what is formulas sorry the formula says that psi 1 has to be true until psi 2 correct.

So, if what happens if in the system all this psi twos are true, then of course the formula is true without any change. So, therefore, what we do is that basically Bx basically we dump B basically we repeat this 1 then basically we check by this formula, we are just going to check that when if Bx is equal to psi of B 2 that is dx is the all the states of the system. So, if all the systems states are equal to B of psi 2 we stop there itself. So, that is again the similar check like AF, so anyway and we have to keep on repeating till there is no more changes.

Now, what we do how the labelization that is very very important that is by this formula. So, we just see an intuition and then we will do is symbolically. So, what we do? So it is E X so if you look at it is eu; that means, there exist a path where psi 1 until psi should should hold. So, in this case by intuition it should be a and operation so and operation means psi 1 should keep on holding sorry it will be psi 1 psi 1 psi 1 until your psi 2 should start holding.

So, what we do so first is you can see B of psi 2. So, a B of psi 2 holds that means, you need not worry at all. So, all the states where psi of B of psi 2 holds you can directly level it.

So, we are having a all the set of states where psi 2 is true you are intruding and we will do a or operation with this, but this apply if you look at so this part basically, this part this psi 2 and or this just simply corresponds the case where psi 2 is true because our main job is to wait psi 1 psi 1 psi 1 and psi 2 should start holding; this psi 2 starts holding from any state that state can be directly taken you know psi 1 psi 1 psi 1 until psi 2. So, whenever psi 2 starts holding your life is safe kind of a business. So, basically so psi 2 we are just keeping all the states where psi 2 will be there we are going to take it.

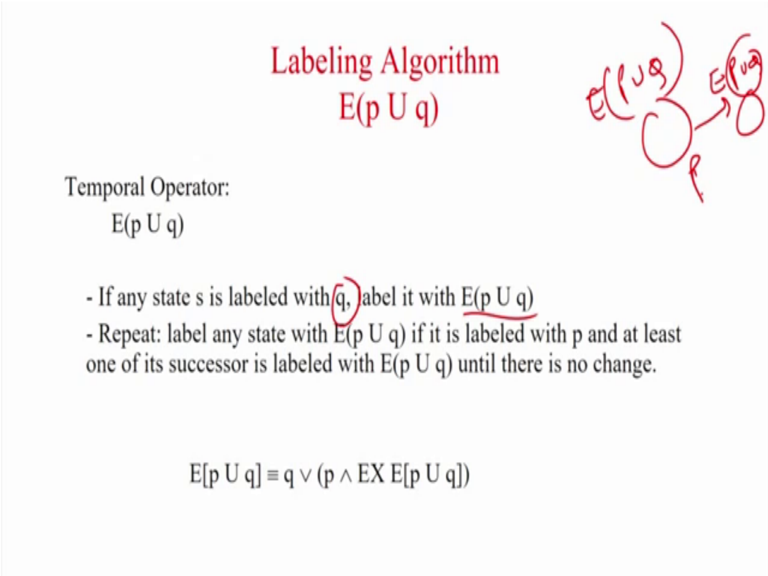
So, that part is simple or with a psi 2, now this 1 is particularly important. So, what we are doing over here in this case we are taking pre of B psi 2; that means, there is a state say x there is some state here, so here your formula psi should stood hold. So, that this pre of B of psi 2 tells that there is a state say x and this is a state where psi 2 is holding.

So, this state I can this state is actually nothing about your pre existential of psi of psi 2. So, this is the state where pre of x holds pre of psi 2 because, this is from this there is an h where psi 2 is true you take all such things and then you and with a state of B of psi 1 that is what that is you take this apply means apply dot; that means, that this is a and anding.

So, what we are anding with it you are anding with a state where psi of psi 1 is true; that means, you are doing something like this. So, again repeating so pre of this 1 is going to stand for the states like this, where in the next step psi 2 is true; along with it we are not going to return this along with this you have a and condition there is a applied dot what is that and but in this state psi 1 is going to be hold. So, psi 1 next state psi 2 is going to be 2; this is the set of states you are going to labelize of course, because psi 1 is true and the next step basically psi 1 is psi 2 is going to hold.

So, psi 1 until psi 2 very easily is holding in the situation such states you will report, along with that we will also report some states like this where psi 2 holds itself like something like this where psi 2 is true. So, that is going by the apply and the second part this diagram as I have explained you correspond to this and if you have to keep on doing this iteration there is a fixed point right.

(Refer Slide Time: 40:49)

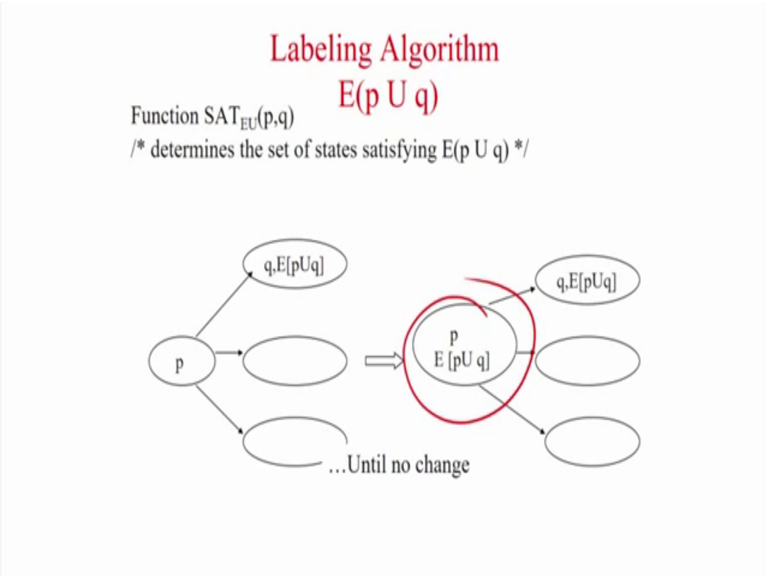


So, again as I have told you again non symbolic way of doing it because, we have already discussed in lecture number 4 just recollecting. So, what it does E p Uq, how do we do it label any state S is labeled with q label it with p and q p and q because q is holding true.

So, we need not worry at all life is safe you apply it you hold with you apply a labelize with p Uq, then label any state with Ep Uq if it is labeled with p labeled with p and at least 1 of it is successor is labeled with E of p and q here that is the set state, so what that is the second clause. So, the first clause is very simple the first clause is add or of B of psi 2 if q is there you labelize it otherwise also you can labelize this 1, if there is a state like this where p is true and in the next step E of p until psi holds that is then only you will also label it as p until q right.

So, that is what is being told by this algorithm you know which you have already seen in Lecture 4 pictorially.

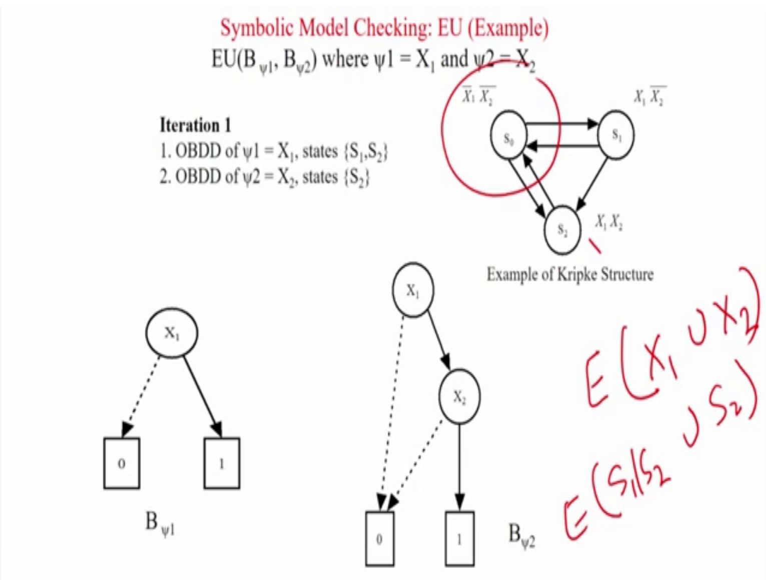
(Refer Slide Time: 41:57)



So, q is true over here so your life is cool. So, here you can write E of p until q now next step very easily, here also basically this is p is holding over here and already you know that p until q is holding over here. So, here also you can elaborate or you can labelize that E equal to p until q, whatever I told you is showing any diagrammatic fashion. So, now you have to keep on doing it there is no until no change.

So, now we will do this symbolically. So, the symbolic formula also we have discussed it will be collecting a BDD of phi 2 you will be oring it with this formula, you have already seen which corresponds to this figure.

(Refer Slide Time: 42:38)



Again taking an example and finding an out how we are going to do it. So, there will be 2 iteration 1 iteration will do it slowly the other iteration you can just look at this slide and compare and compute yourself again the same Kripke structure you are going to take.

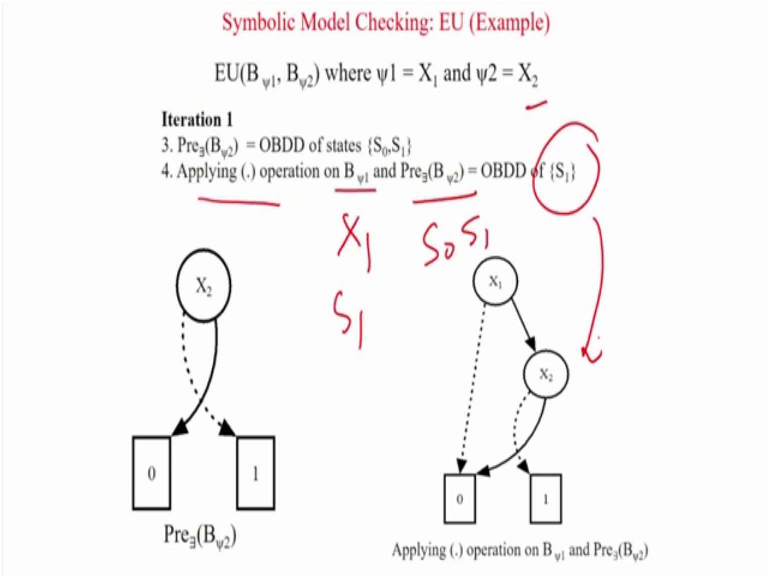
So, we have to keep it in mind, now what you are going to say here we are going to check E U there exist a path where psi 1 is equal to X1 and psi 2 is equal to x 2; that means, we are going to find there exist a path where X1 until X2 that is what we are going to put. So, what is where is X1 is equal to true X1 is going to be holding to be in state S1 and S2 and when state X2 is true state X2 is true only in state x 2.

So; that means what existential of S1 or S2 until S2, so that is what is going to be and there should exist a path where this 1 should hold. Now, you can see intuitively we can see that look at the state x naught. So, what is the problem with x naught, you can see in this state what you say X1 should hold until X2 in this state you say X1 is also false and X2 is also false; so of course, S naught is not going to satisfy the formula.

But if you look at this 1 X1 is true X2 is false fine, but in the next state what is happening X2 is going to be true and X1 is also holding. In fact, even this would not have been a problem, so what do you saying X1 is true until X2 starts becoming true. So, in this case X1 is 2 and X2 false no problem just in the next state you can see that is in future may be a no problem that X1 is true, I do not bothered even if X1 is false here, but X1 is becoming a true over here. So, X1 is holding as long as X2 is not holding.

So, x S1 is satisfying in S2 if you see both X1 and X2 are true over here. So, in whichever state X2 is true directly we can enumerate it. So, intuitively and also we will find out towards the end that x naught will not satisfy the formula, but X1 and X2 will satisfy the formula ok. So, now again let us coming back to stepwise analysis, so stepwise what is that OBDD of psi 1 is S1 S2 this 1 OBDD of psi 2 is X2 which is this BDD this BDD.

(Refer Slide Time: 45:02)

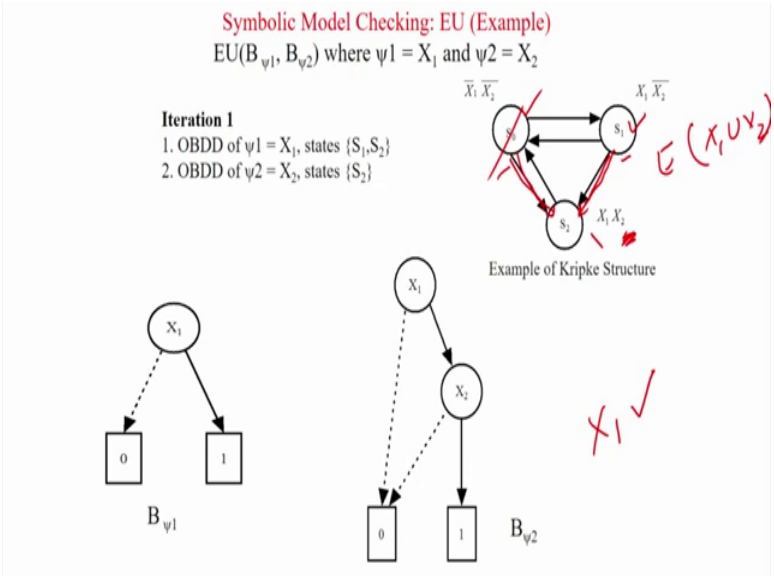


Now, we are going for iteration, so if you look at the formula so what we have to do we have to first find out the pre of existential of the states of psi 2, this is the formula we are going to go by this formula.

So, if you look at pre of psi 2 that is equal to the psi 2 means in this case it is x 2. So, wherein x 2, so S this 1 is X2 this 1 is x 2 so ok. So, what are the pre of S 2. So, a pre of S2 is nothing but S1 and S 0 because that is true that is obvious. So, basically pre of S2 is S naught and S1. So, what is reflected over here so these are symbolically you can compute the value of pre of S1 that is BDD of S naught and S1 that is this states psi 2 is x naught that is state S2. So, what is the pre of S2 is S 0 and S1 because there is a direct edge to this states and we are going for 1 existential operation 1 path is enough.

So, basically S naught and S1 this is the B corresponding to this 1, then what we have said then these are the states Basically where there is at least 1 path where x naught is becoming equal to true, there is a at least 1 path where x naught is true.

(Refer Slide Time: 46:15)



So, S 0 and S1 is done so these are the states where the next state X1 is going to be true at least 1 path is there. So, such states with of obviously, hold the value of e X1 until X2 because, in from these two states at least one path is there where in future x naught is true along we along with that so in this case ps naught and S1 qualifies.

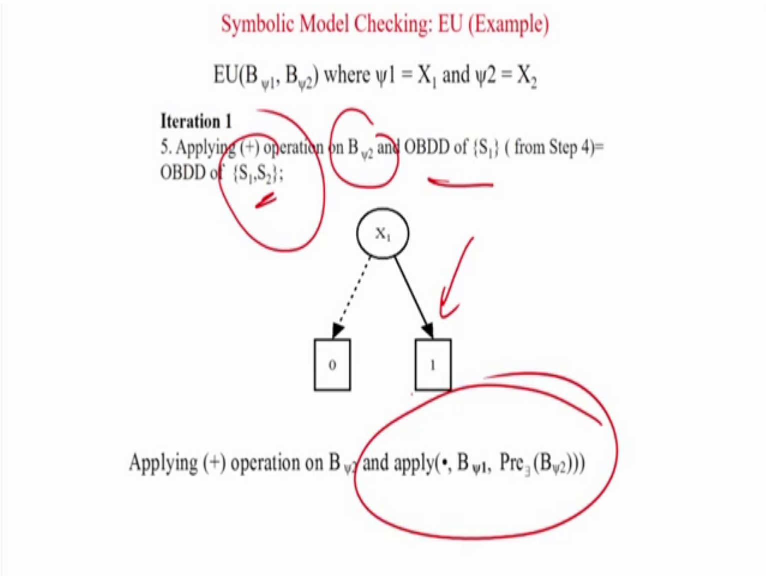
But along with this condition we cannot take both the S1 and S2, now sorry S 0 S1 cannot be taken right away even though both has a path where X1 sorry X2 is true in the next structure, we have to ampersand it with the case where X1 is 3 true in the step right this is the next. So, from S2 pre you are directly going to get sate x naught and this 1. Now, you have to ampersand it with the cases where this take X1 is true, so we are going to have state where X1 is equal to true as well as the states from where in the next X2 is going to be true.

So, of course we will find out that S naught will get eliminated because X1 is false over here, but we will keep retain S1 because X1 is true over here. So, now next 1 is something like this operation applying operation dot on B of psi 1 psi 1 is nothing but X1 and pre of this 1 is S 0 and S 1. So, this 1 is only satisfied by state is 1 because at state S naught x X1 is false. So, you are going to apply this operation and we are going to get the BDD of S 1.

So, the step for what is going to return step 4 is going to return these states in which case X1 that is psi 1 is equal to true and their existence is 1 part where in the next state X2 is going to be true. So, that is what is being returned by S1 and which is this BDD represented by this BDD.

Finally, we have to or it with so finally, we will get the answer S1 which is satisfying the clause that in the present state X1 is true and in the next step X2 is true that is phi until phi 1 until phi 2. Next basically you have to take all such states where X2 or psi 2 is by default true that is basically your Or operation of B of psi 2 and whatever we have gone B of psi 2 in the state where by default X2 is true.

(Refer Slide Time: 48:28)



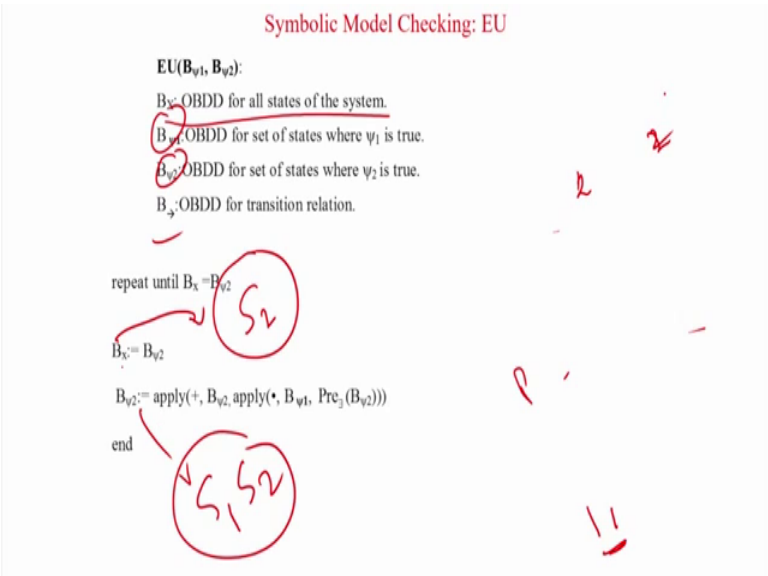
So, that actually happens in state S1 we are going to get a or operation on this and finally we are going to get the answer of S BDDs S1 and S2 which is this BDD, so this is the answer of applied.

So, now what we have got so in this iteration you have got S1 and S2. So, what is that? So, now in this iteration we have got S1 by the virtue of the fact that here psi 1 is true and in the next states psi 2 is true, that is by this complex by this part of the formula we have got this one; by this part of the formula we have got the solution of S1.

Now, you have to do this or operation and this 1 will be done by this 1 S2 is simply where X2 is true, so we are getting by this iteration we are getting the value of S2 over here there is just a simple operation.

Finally this is the BDD where E U psi 1 until psi 2 satisfies as we the answer is state S1 and S2, now they just stored intuitively we have seen that S1 and S2 surfaces; but algorithmically low because algorithmically your you have to stop only when there is no change there is a fixed point or all these states has been taken into picture. So, now in this case initially you have stored if you look at B of psi is all the states of the system right. So, now initially what we have done B of psi 2 if you see we had what B of psi 2 in this case was S2 right.

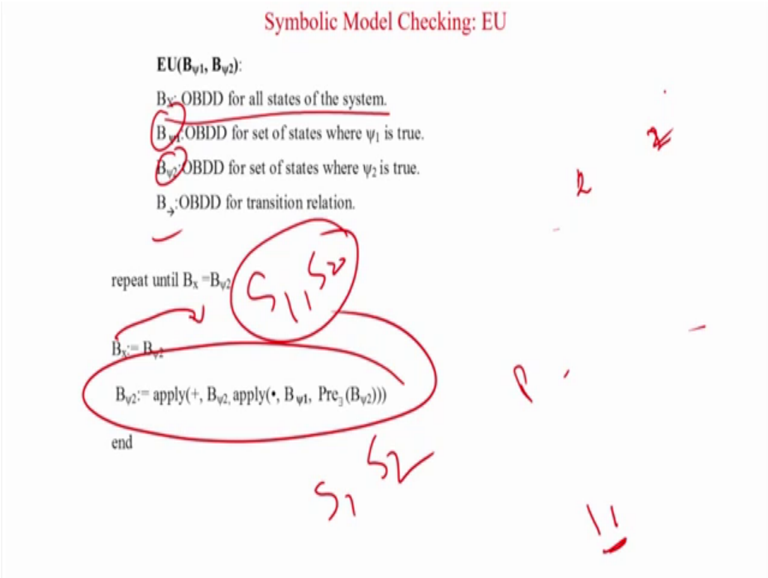
(Refer Slide Time: 50:00)



So, now we had store X2 over here and now sorry S2 over here because, initial psi 2 was S2, so we have stored S2 over here then we have updated this. So, what we have got it over here we have updated and we have got here was S1 and S2 ok, now you have to repeat this until this 2 are equal or not; so in this case they are not equal so there will be another more iteration.

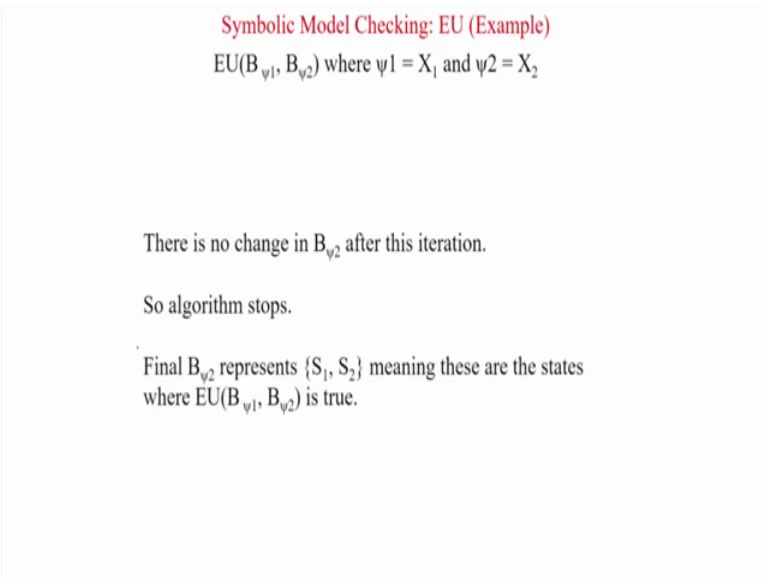
So, now again in the next step what is going to happen we are again this 1 is psi 2 x only S2 in the initial round, in the second round you are going to change this 1 and you are going to have sorry you are going to replace S1 this 1 with S1 comma S2 and again rerun the whole algorithm again to see what it returns.

(Refer Slide Time: 50:55)



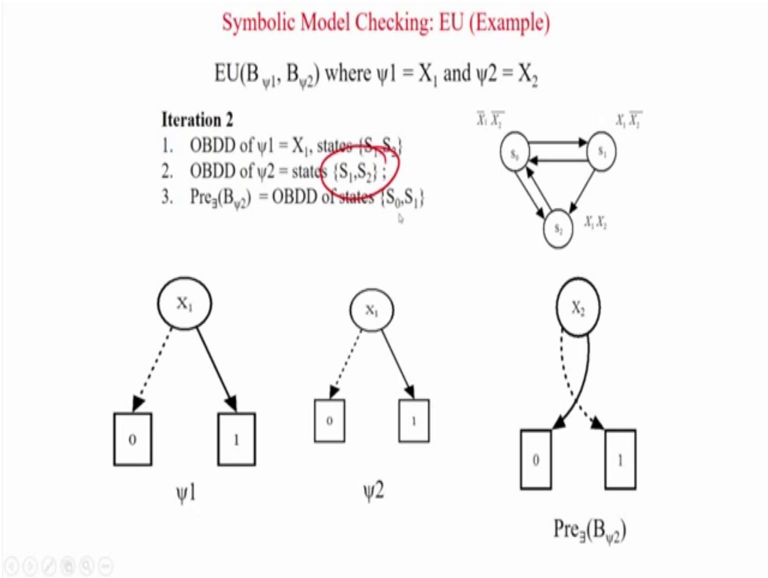
So, now your Bx is having the value of S1 S2 initially it was only S2, but now B of psi 2 has been changed it is now S1 and S2. So, Bx in the next iteration become X1 and X2 and we again redo the second iteration.

(Refer Slide Time: 51:08)



So, let us see what is the second iteration? So, this is iteration one is done; so this is iteration two.

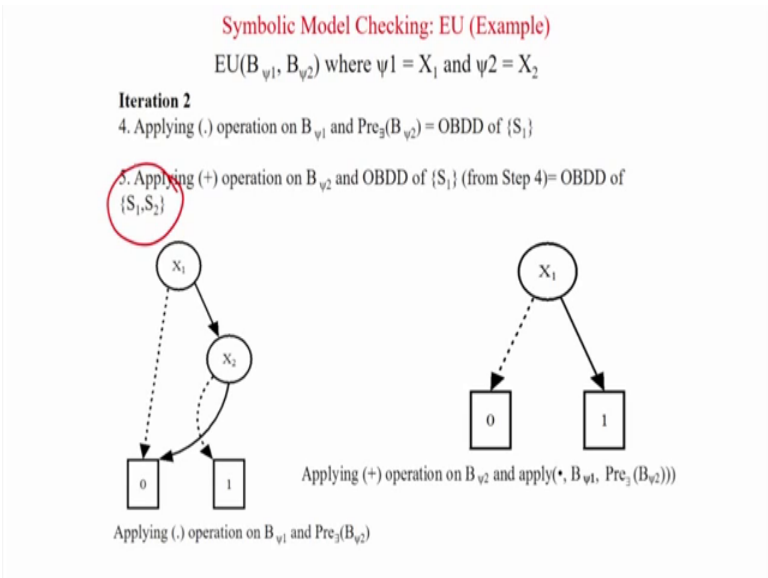
(Refer Slide Time: 51:54)



So, only change is now psi 2 has become S 1 and S 2 it was just similar. So, if you look at, so again in this case only the your psi 2 has changed because, now it is S 1 and S 2 in a initial round it was S 2, but X1 is anyway change.

So, you will first do the again the same step calculate the pre of psi 2. So, in this case it will be states S 0 and S 1 just you can do the computation.

(Refer Slide Time: 51:37)



Then again we will go for the application of dot operation on B of psi 1 and this one will get the value S 1, you can just do the check and then again apply operation of B of psi 2 psi 2 with these case states where default the formula is true and OBDD of S 1 which you have got in the last step and then finally we are going to get the answer S1 and S2. Just the same states which I have applied in the round one and I have shown you in details same thing you have to just apply with the change of psi 2 of S1 psi 2 is now becoming S1 and S2 same 4 steps 5 steps you have to apply.

Again finally, you are going to find out that I am going to have the answer of S1 and S2 again. So, now initial round we had psi 2 equal to S2 after doing it answer was psi 1 and psi 2 sorry X1 S1 and S2. So, you have to again iterate but in the second iteration you will find the answer is equal to S1 and S2. So, and this iteration you have to stop because there is no further addition and we have reached a fixed point, that is again that is again I am going to come to that algorithm.

So, now we started with S1 and S2 after completing this iteration again the answer is S1 and S2 there is no new more states has been added so you have to stop that iteration first. Now, we have B of x equal to only S1 and we added one more state that is S2 in that round and in the first iteration, so if they again go for second iteration to check if any more new state can come.

But we finally, concluded that no more new state has come. So, only S1 and S2 is going to satisfy the formula fixed point is this and you stop over here. So, intuitively also we had seen that only in the case of S1 and S2 there exist a path where in future X1 until X2 will satisfy, because in S naught this is the this is false because X1 is not holding even X2 is not holding. So, intuitively S2 will not be a part of the state set, but the formula will be satisfied.

So, similar things again I have got it with the help of this symbolic model checking I have again I have getting the answer S1 and S1 and S2 over here, just you can repeat the calculation and we are done. So, basically we have seen how the three important or the three basic formulas required to replace any CTL formulas that E X E U and AF can be handled symbolically.

Now, given any very complex formula you can just use these three specific formulas to represent them and you can do this model checking symbolically. So, what we have achieved, so we have seen that model checking is a very very powerful tool which can automatically tell you which of the states which are satisfying the formula, if some states are not satisfying the formula or if any of the states are not satisfying the formula, why it is not satisfying the formula some counter examples are also generated.

So, you can in mathematically you can prove whether their your system will going to is going to satisfy the requirements you want to decide, that is the beauty of model checking. But problem with explicit state space modeling is the exponential complexity. So, to handle very large scale systems or complex systems we have to go for symbolic model checking, that is you have to model them using BDDs without explicit states modeling and the labeling or model checking can be done as we have explained in the two lecture series of 4 and 5.

But again this as we are going by BDDs as already I have mentioned some several times that higher level decision diagrams, HLDDs mathematical decision diagram arithmetic decision diagrams still people are working to find out how they can be used for symbolic model checking.

BDDs are widely used for single symbolic model checking, but if your systems are if you want to model them in an abstract level by using our register transfer level models or arithmetic decision diagram levels, still model checking algorithm in those level of BDDs does not exist or under research. So, that is a very open and a open and interesting problem to research upon.

So, even if even you as you are telling in the course that we are handling NOCs and SOCs or symbolic model thing directly if you want to said I want to have use of very large NOC to handle it is not directly appropriate as of.

Now, to do it in that in that quote unquote fashion that I can use symbolic model changing to modular whole NOC that is quote unquote not as simple as to do because, BDD is also have a limitation to handle NOC and SOCs as you have already seen in testing part and several other parts that we can use a high level decision diagram to do that.

So, again if you want to have a very nice optimization answer you should have a symbolic model checking at the level of arithmetic decision diagrams or high level decision diagrams, which is not unfortunately which is still under research and not as matured as your symbolic model checking using BDDs. So, that is when we you want to conclude these two lecture series on symbolic model checking and next we will just going to see something called a bounded model checking, that is in future you do not want to extend the future to infinity because in a practical sets future is limited. So, maybe we have we have we will just export up to a few state iteration and try to conclude something and get some optimization out of the solution.

Symbolic model checking does not hamper any of the quality of solution this is just represent states as a symbolic built in manner and does the computation for you; but on the other hand if you are using a bounded model checking; that means, you are not allowing to full extent to check, future means maybe 10 iterations, 20 iterations, 100 iterations future cannot be infinite we put a bound on them.

So, in a very lamer language it will have slight impact on your situation or your verification answer, may be something would have been failing up to a ten thousand states, but you are checking only up to a certain limit. So, in the but you have very large scale system that is what is the optimization solution, so in the next lecture on verification we are going to discuss elaborately on your symbolic model checking.

Thank you.