Embedded Systems - Design Verification and Test

Dr. Santosh Biswas
Prof. Jatindra Kumar Deka
Dr. Arnab Sarkar
Department of Computer Science and Engineering
Indian Institute of Technology, Guwahati

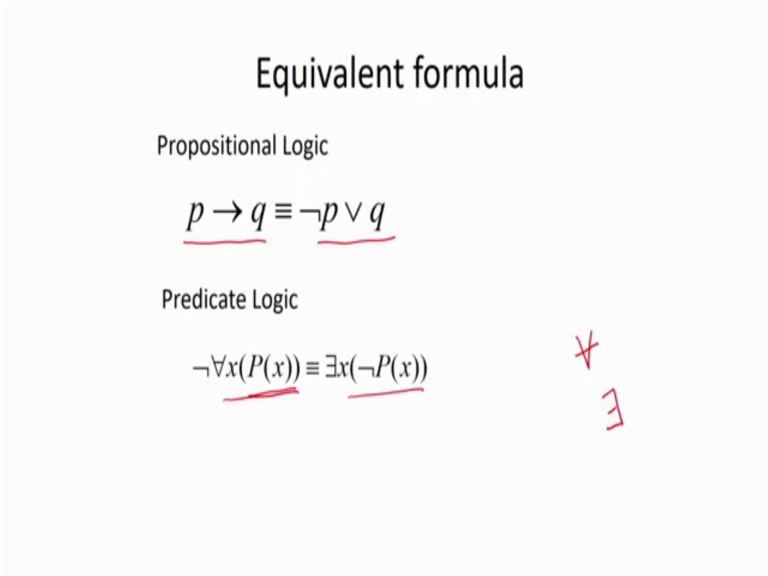
Lecture - 21
Equivalence between CTL formulas

Welcome back to the online course on Embedded System Design Verification and Test. So, in this part we are discussing about the temporal logic and in last class we have seen about a particular logic called CTL, Computational Tree Logic. We have discussed about the syntax and semantics of CTL. Now, we know or we have idea about the meaning of a CTL formula.

(Refer Slide Time: 00:59)

Now, today we are going to see about the equivalence between CTL formulas. So, when we said the two CTL formulas are equivalent. Now, what is the notion of equivalence in terms of CTL formulas, ok.

(Refer Slide Time: 01:12)

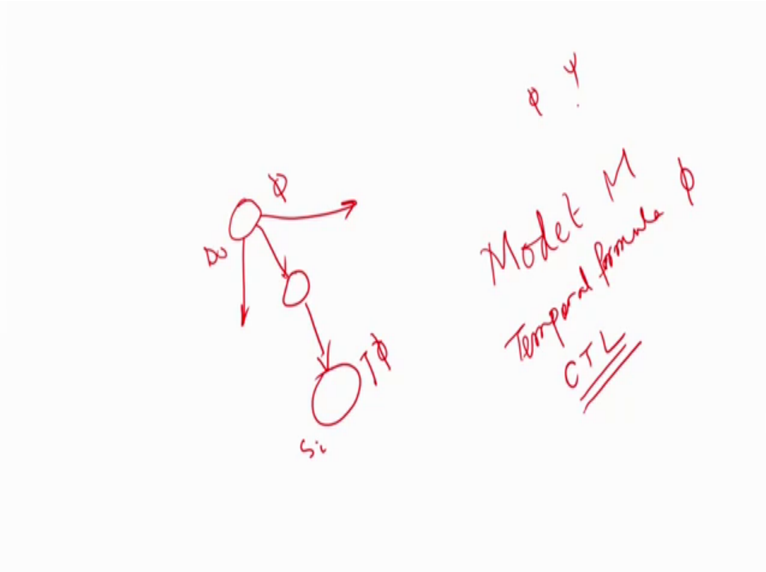


Now, we know about the equivalent formulas and if you look back into the propositional logic and predicate logic we know when we said formulas are equivalent. As a simple example I am giving here the p implies q or if p if p then q this is equivalent to not of p or not of q, not of p or q. So, all of you know this particular difference that means, what it says if the truth values of if p then q or p implies q is true then the truth values of not of p or q is also true. If the truth values of p implies q is false then the truth values of not of p or q is also false. So, that is why we said the truth values of same in all possible scenario, all possible truth values of their components p and q. So, that is why I said these are the equivalence formulas.

Now, in predicate logic what we have actually apart from those particular logical connected we are having two quantifier, one is known as your for all and there exist, ok. For all x that means, for all possible values of x the predicate P x is true, and there exist some x for which the given predicate P x is true. Now, in this particular case we are having an relationship between for all and there exist. So, this is what we say that if for all x P x and if the negation is true for this particular formula then what basically we can say that there exist at least one x, one value of x for which P x is false, ok.

So, for all values P x is true if I said negation that means, this is not true for all values of P x. What does it means? At least we are going to one values of x for which P x is not true. So, that is why there exist not of P x. So, these two formulas are equivalent. So, there are for all possible values of your equivalence for all possible values of x and the given predicate P x. So, this is called not of for x, for all x P x is equivalent to not there exist some x not of P x. Now, what is the notion of equivalence in case of temporal logic formula? How do you define the truth values of a temporal logic formula?

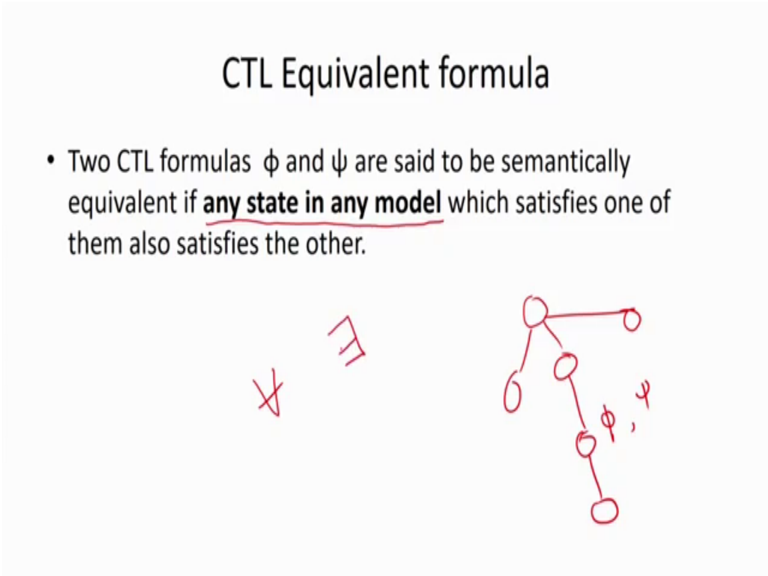
(Refer Slide Time: 03:54)



So, basically we are having a model M and we are having a temporal formula say phi or say in particular we can say that we are talking about the CTL formulas phi. Now, the truth values of the CTL formula defines over the model M and also truth values is defined in a particular state of this particular given model M, because phi or CTL is a state formula truth values is defined on a state of a given model. Now, if we look into it then what we can say that truth values of this formula may be true in some state say if formula phi is true in this particular state say s 0 they are having several possible transition and in other state that s i the formula is not true, ok.

So, this is the notion of truth values of a CTL formula or in general temporal logic formulas. So, in case of state formula the formula maybe true in some state and may be false in some other state of a given model. Now, when we say the truth formulas phi and psi are equivalent in case of CTL formula.

(Refer Slide Time: 05:23)

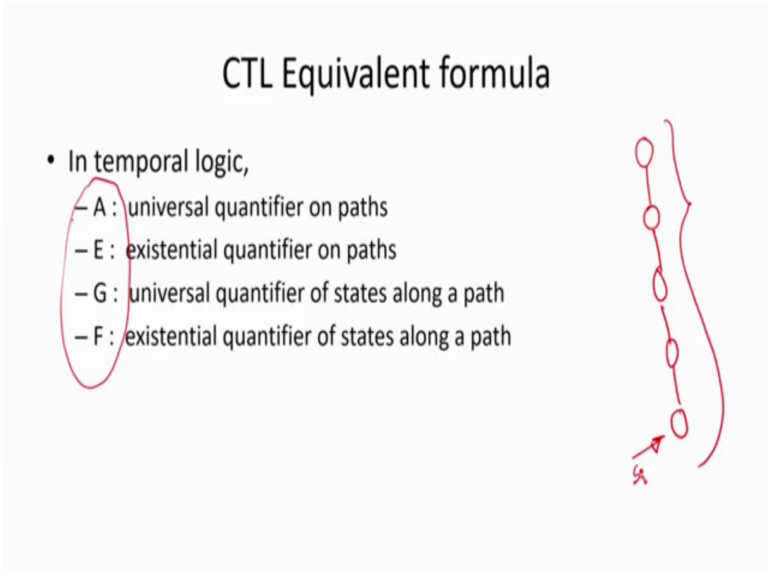


So, basic notion is we say that, two CTL formulas phi and psi are said to be semantically equivalent if any state in any model, this is the important things we just say that any state in any model which satisfies one of them also satisfies the other.

So, if I am considering any model M, ok, so in a particular state say phi is true and psi is also true. In that case what we are going to say that if truth values are same for any given model or any state if they are always true whatever model we are going to consider whatever state we are going to consider if phi and psi are always true then we are going to say that they are equivalent. On the other hand if phi is false then psi is also false then also we are going to say that they are equivalent. So, the truth values must be same in all any state of any model then we say that these two states are these two formulas are equivalent.

Now, how we are going to look for the equivalence? You just say that in case of (Refer Time: 06:44) predicate logic formulas we are getting a relationship between for all and there exist. That means, if for all it is not true and there exist something for which it is false we know about this notion if for all some predicate is not true, then at least we are going to some values of x for which P x is false. So, this is the way we are going to look into the equivalence in the predicate logics. So, in the similar notion we are going to see what are the quantifier that we have.

(Refer Slide Time: 07:14)

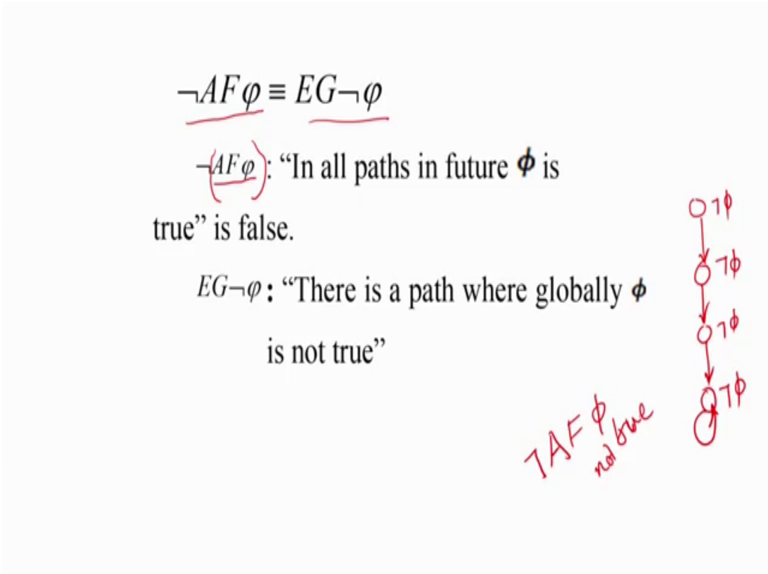


So, here we are having two path quantifier, one is your A, A is your universal path quantifier and E is your existential path quantifier. So, it says there exist a path of in all possible path. So, these are having there exist and for all which is very much similar to your predicate logic. So, we must have a relationship between A and E. So, if it is not true in all paths there exist some path where the given formula is false this is one. So, I can looking it.

Another tool we are having one is the temporal operator G which means globally and another one is at which is in future or eventually. So, if you look in to it x at notion then G can be treated as a universal quantifier of state along a path. So, we are going to talk for all possible state in a given path. So, that is why I am saying it is a universal quantifier of states along this particular path. Similarly F future or eventually it is going to talk about a particular path the particular state in this particular path. So, it is an existential quantifier of state along a path. So, again you just see that we are having the notion of for all and there exist in case of your paths. Similarly we are having a notion of all states or there exist a state along a particular path and this is captured with the temporal operator G and F.

Now, by considering these particular 4 parameters AE and G of F we are going to have some equivalence formulas. Now, we are going to see what are those equivalence basically we are going to have.

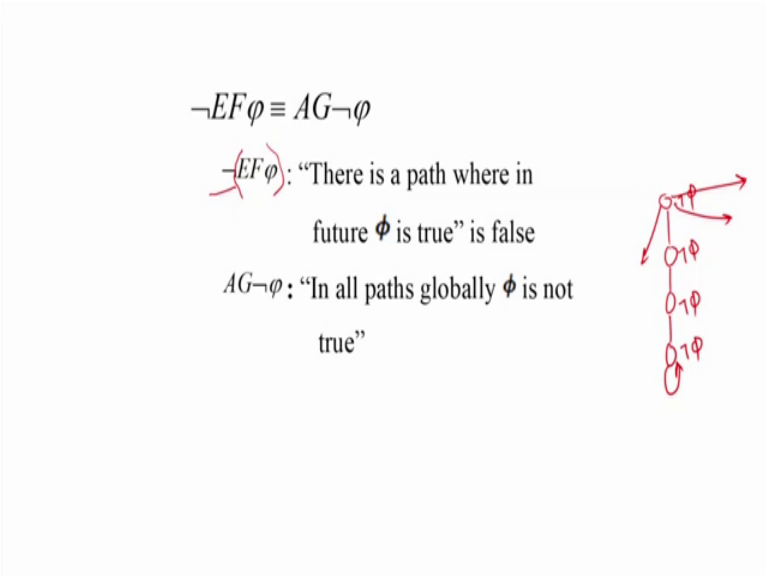
(Refer Slide Time: 09:20)



So, first equivalent formula we are writing it as not of AF phi is equivalent to EG not of phi, ok. What does it means? Not of AF phi it says that in all path in future phi is true is false. So, it says that in all path in future phi is true is basically false that means, in all path in future phi is not true. So, since it is going to be false in all paths what we can say that there exist a path where globally phi is not true. So, there exist a path where globally phi is not true that means, you can say that if in a particular path, if phi is not true then what will happen at least in this particular path phi is not true globally. So, at least in this particular path in future we are not going to get any state, where phi is going be true. So, that is why it says that in this particular path F of phi is not true basically, not true, and if it is going to happen in all the possible paths then we are going to say that in all path in future phi is true is false basically.

So, here this is equivalent now, you can say that at least we have a exist a path where globally not of phi is true. Now, if you consider any model and if you go to any state of a particular in any model then if the truth values of not of AF phi is true then the truth values of EG not phi is also true. Similarly if the truth values of not of AF phi is false then truth values of EG not of phi is also false in that particular state. So, if you consider any models, and you look for any particular state you will find that if the truth values of both the formulas will be same either both are true or both are false. So, that is why we are going to say that these two CTL formulas are equivalent. This is the first equivalent formula that we have observed. So, not of AF phi is equivalent to EG not of phi, ok.

(Refer Slide Time: 12:06)

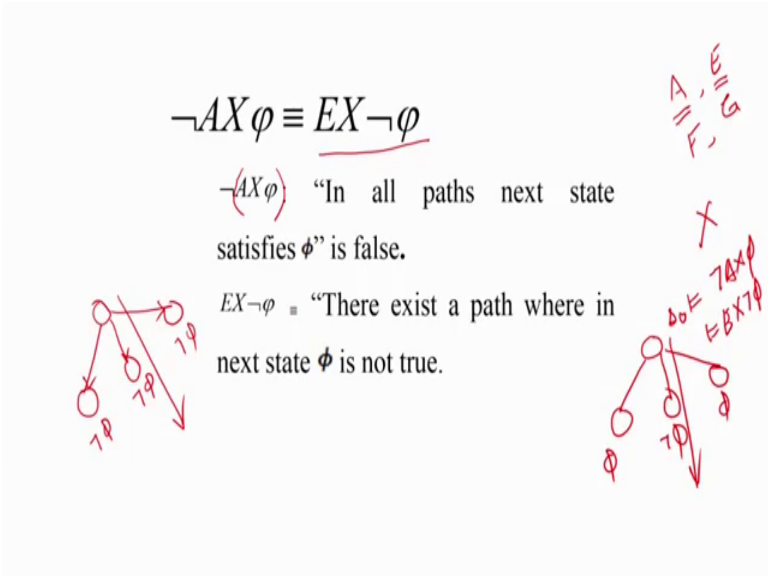


Another one, now it is the reverse you can. Now, correlate second equivalent formula is not of EF phi is equivalent to AG not of phi. So, not of EF phi what does it means there is a path where in future phi is true is false because this negation is there. So, we are talking about the formula EF phi and this is the negation. So, there is a path where in future phi is true is basically false, the exact meaning of this formula is telling like that so AG not of phi, ok.

What does it means? In all path globally phi is not true, ok. So, in all path globally phi is not true. So, again you just see consider some scenario if globally not of phi is true in a particular path again the similar formula model I can have, say this is your not of phi. So, in this case in this path globally not of phi is true. Now, if you same scenario exist in all possible path then we can say that in AG globally not of phi is true. So, if this is the scenario. Now, wherever you go you are not going to get any path where in future phi will be true because everywhere it is not of phi. So, this is also another equivalence that we have observed not of EF phi is equivalent to AG not of phi.

So, in any model in any state if one is true then other is also true. So, either both are true or both are false and that is why we say that these two formulas are also equivalent.

(Refer Slide Time: 14:05)



Now, we are talking about the relationship between path quantifier AE and the temporal operator F and G. Now, another temporal operator we are having X which is your next state, ok. Now, whether something is true in the next state basically we are going to design about it. Now, that next state is also an having an relationship with all path and there exist a path, ok. So, this is another equivalent that we having A not of AX phi is equivalent to EX not of phi.

So, not of AX phi basically what does it means or what is the exact interpretation it is we can say that in all paths next state satisfies phi is false. So, it what it says in all path next state satisfy phi is false, ok. So, this is the meaning so that means, if I talk about one particular state and if I have some scenario something like that and say everywhere is not of phi then what we are going to say that wherever you go whatever path you are going to consider; in next state that satisfy phi is a false that means, phi is not true in any of the next state.

So, this is basically you say that there exist a path in next state not of phi is true. So, if you consider any path you find that not of phi is true. And due to this reason only what will happen AX not phi is true. So, I can write another diagram. So, in this scenario also you see what is the status of the formula not of AX phi we will find that not of AX phi is true over here because in all path in next state phi is not true. So, at least one path we are having where not of phi is true. So, this not of AX phi is true in your this particular state s 0 not of AX phi and this is happening because of this particular path, so that means, I can say that again s 0 models there exist a path X not of phi.

So, again you see that we have got an relationship between the all path and there exist a path of this path quantifier along with the temporal operator next state, ok. So, not of AX phi will be equivalent to there exist the path in the next state not of phi is true. So, these two formulas also have that same truth values in any state of any model either both are true or both are false. So, that is why we say that these two formulas are also equivalent. So, this is the formula already I have explained. Now, here we can with the help of these 3 given equivalence we can get another 3 equivalence also and you know about these things a negation, negation property.

(Refer Slide Time: 17:24)

$$\neg AF\varphi \equiv EG\neg\varphi \quad \rightarrow \quad AF\varphi \equiv \neg EG\neg\varphi$$

$$\neg EF\varphi \equiv AG\neg\varphi \quad \rightarrow \quad EF\varphi \equiv \neg AG\neg\varphi$$

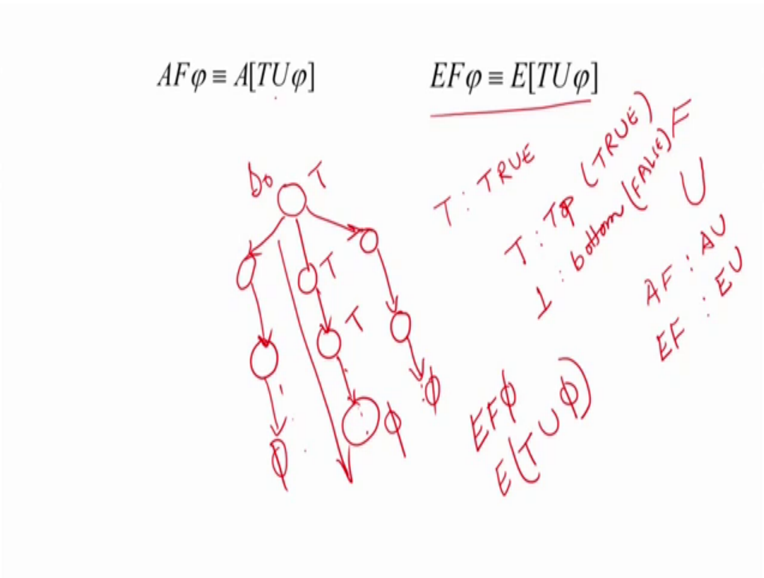
$$\neg AX\varphi \equiv EX\neg\varphi \quad \rightarrow \quad AX\varphi \equiv \neg EX\neg\varphi$$

So, already we have seen that not of AX phi not of AF phi is equivalent to EG not of phi. So, if I negate both the side then they will also remain equivalent. So, after negating both the sides what I am going to get AF phi is equivalent to not of EG not of phi, ok. This is one equivalent it is derived from this given equivalent such taking the negation in both the side.

So, similarly other two equivalence also come from these two equivalent not of EF phi is equivalent to AG, not of phi so from here we can derive this particular equivalent by taking the negation in both the sides of the equivalence. So, EF phi will be equivalent not of AG not of phi. And third one not of AX phi is equivalent to EX not of phi. So, given by taking the negation by negating in both the sides left hand side and right hand side we are going to get AX phi will be equivalent to not of EX not of phi. So, now, we are getting 6 basic equivalent relationship over here, ok.

Now, the way we have explained or we have seen how these two will be equivalent. You can draw some structure or model and just see how these two will be equivalent, in simple way we can do it. So, these are the some equivalent formula that we have in our temporal logic or in particular CTL computational 3 logic.

(Refer Slide Time: 19:14)



So, another equivalent or true equivalent here I was writing one is talk about the future. So, AF phi is equivalent to A T until phi second equivalent EF phi is equivalent to there exist A path T until phi that means, we are having temporal operator F, this temporal operator F is having some relationship with the until operator. This is the way you can say and whatever it may be if it is your AF or EF then AF will be related to your a until and EF will be related to E until. So, this is the things.

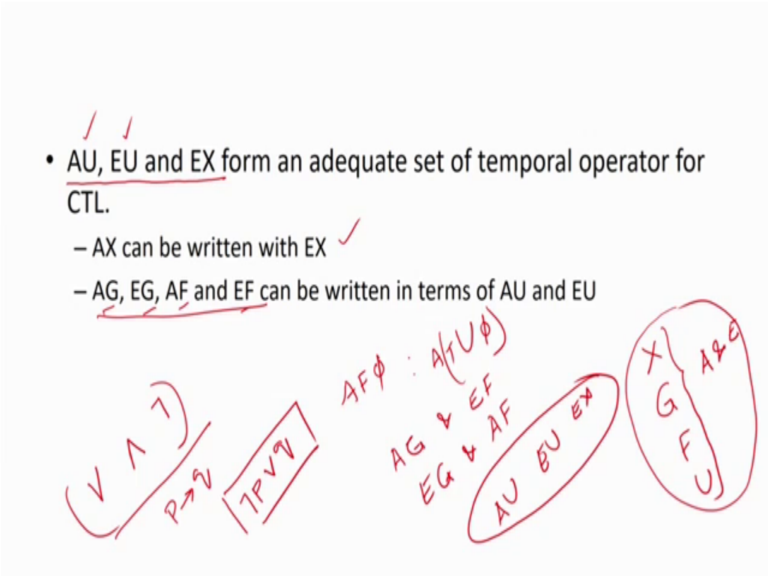
Now, what we are seeing? That AF phi is equivalent to in all path E until phi. Now, or EF phi is equivalent to E T until phi. Now, what is T? T is nothing but the truth value true here this coming as starting state but already we have introduce the notion of two symbols one called top and bottom, ok. While defining the semantics syntax of our CTL formula we have introduced these two notion top and bottom these two symbol and basically top indicates truth value true and bottom indicates truth value false, ok.

Now, while we are defining the semantics of CTL we have defined the semantics in a given model M. Now, what we have? We have observed or what semantic has been defined for this particular truth values it says that true is true in all state and false is false in all state, ok. That means, true top or the truth value true will be considered as true in all state and bottom will be considered as false in all state. So, this is the two constant we can say that two constant we are having in our CTL which represent the truth values true and false and the basic notion is like that true is true everywhere and false is false everywhere. So that means, if I look into the labelling of this thing that means, we can consider that everything is labelled with the truth value true.

So, now in this particular case what happens if I am having this scenario then what will happen here I can say that in this particular state s 0, EF phi is true at least there exist one path where in future phi is true. Now, by looking into the labelling you say that true is true everywhere that means, all the state this is the true will be considered. So, that is why I can say that it can be written as a E true until phi, true will remains to until phi becomes true. So, this is the relationship that we are having there exist a path in future phi which will be equivalent to there exist a path true until phi. And if some scenario prevails in all the path, that in all path in future phi is true then this can be again designed as A true until phi.

So, this is another relationship or equivalence relationship we have the relationship is between future and until operator, and path quantifiers have to be same if it is A then AF phi is equal to A true until phi and EF phi is equivalent to E true until phi.

(Refer Slide Time: 23:37)



So, by looking into it what we are going to say that, we are having total 8 CTL operator because what will happen we are having 4 temporal operator X, next state globally, future and until these 4 temporal operator will be preceded by path quantifier A and E, in all path and there exist. So, all together we are going to get 8 temporal operator in CTL.

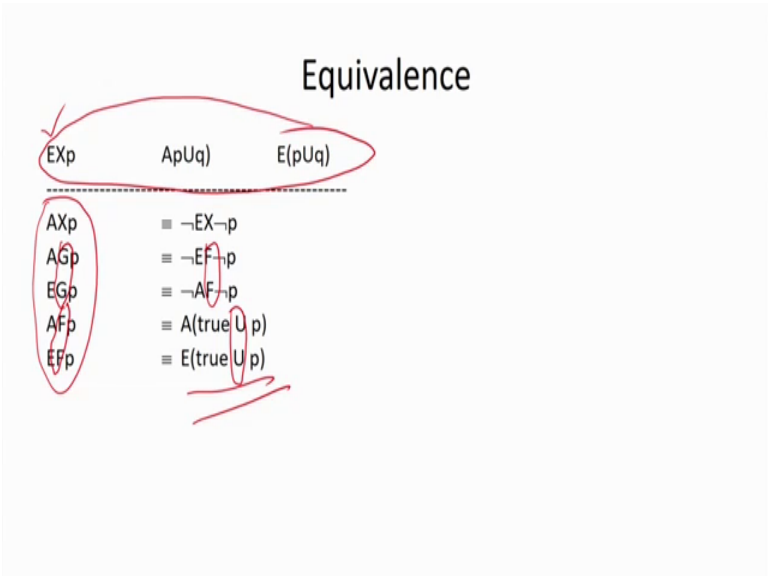
But out of this 8 temporal operator we can say that 3 temporals operators are adequate to express the other operators. So, here we are going to consider one particular case it says that AU, EU and EX from an adequate set of temporal operators for CTL, because we know that AX can be written in terms of EX already we have seen and AG, EG, AF and EF can be written in terms of AU and EU, ok. Because you just see already I have seen that AF we can express it as with until operator true until F. So, basically AF phi can be written as your A, true until phi. Again we have seen that relationship that AG and EF are is having a relationship, ok. So, AG and EF are having a relationship and similarly EG and AF is having another relationship we have seen those equivalence. So, these 4 operators AG, EG, AF and EF can be either expressed with AU or EU, ok.

So, if we are having know the methods how to check the truth values of this particular operator AU EU and EX then the truth values of other operators can also be find out. So, that is why we are saying these are adequate set of temporal operator this is one particular set of temporal operator we may have other adequate set of temporal operators.

Now, you can correlate with this say we are having 3 logic operator, one is your disjunction, conjunction and negation. You say that these state are also complete state of operator for proportional logics, ok, all logic families because any other operator can be expressed with the help of these 3 operator just today itself we have seen that p implies q, ok. This can be written as your not of p or q, ok. Just if you consider any operator that can be expressed with the help of or, and, and not. So, we say this is the complete set of operator in case of proportional logic.

So, similarly here also we have observed that we can have a adequate set of temporal operators in case of CTL, with the help of those adequate set of temporal operator other operator can be expressed. We are considering or we are talking about 8 different operators we are having in CTL, but out of that 3 are adequate one particular set of adequate set we are talking about AU, EU and EX, ok. But this is not only the adequate set of operators we can get some other adequate set of operators but for that we need one more equivalent formula.

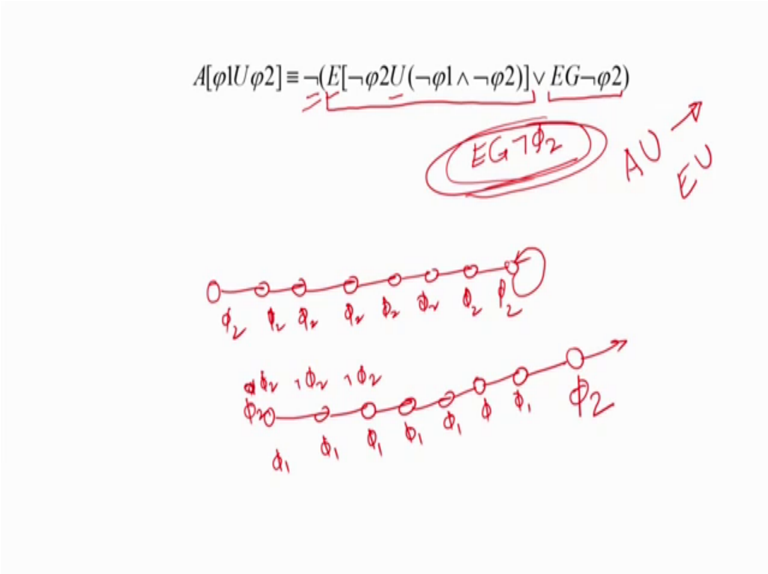
(Refer Slide Time: 27:33)



So, this is the way I can say already I have explained say if I am considering that this is the adequate state of operators. Then what will happen? EXp is equal to not of EX not of p that means, it is expressed with this operator EX, AGp is equal to not of EF not of p but it is not in that adequate set of operator, EG p also your not of AF not of p. So, the G is expressed with F, the operator G is expressed with F. Again the operator F is expressed with U, that means, finally, we are saying A until and E until.

So, if you know how to check the truth values of these 3 operators eventually we will be knowing that truth values of their corresponding operators also because there can be eventually converted to this particular until operators. So, that is why I say that this is an adequate state of temporal operators.

(Refer Slide Time: 28:42)



We are having one complicated equivalence over here. So, I will give the notion it says that A phi until phi 1 until phi 2 is equivalent to this is one big expression but which involve until E that means, we can say that A until can be expressed with E until, ok. So, if A phi 1 until phi 2 is equivalent to not of the whole expression. So, what will happen? In not I am having two expression EG not of phi and E of not of phi 2 until not of phi 1 and not of phi 2, not of phi 1 and not of phi 2, ok. So, this is the equivalence that we are having just I am going to give some notion how we are going to treat this things as equivalent.

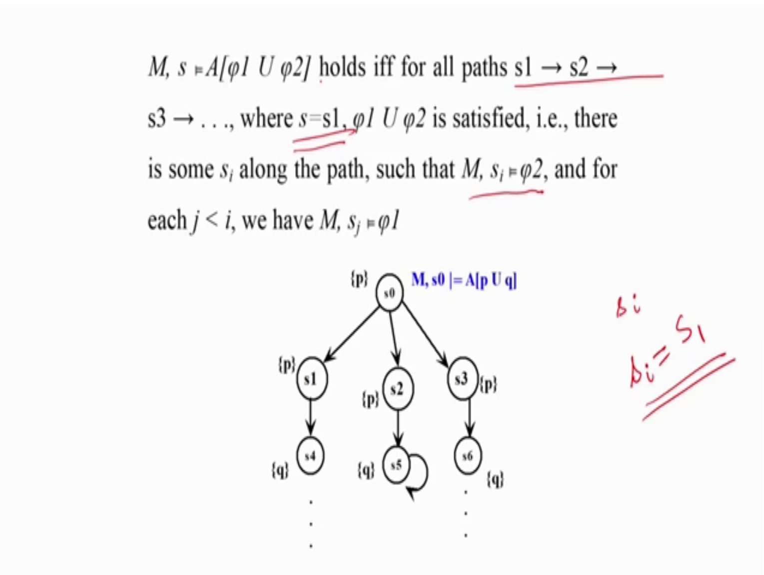
Now, you just see if I consider this component, EG not of phi 2. Now, just consider one path then I can extend it to any path, ok. So, EG not of phi 2, if all the states phi 2 is false then we say that EG not of phi 2. So, at least in this particular path we are not going to have a phi 1 until phi 2. Now, my phi 1 may be true in all the states but phi 2 is not true. So, this is the component it says. So, it happens like that EG not of phi 2 then A until phi A phi 1 until phi 2 is false, ok.

Now, when it will become true? If I take the negation, so that is why this negation sign is there inside this, ok. So, if such type of scenario is happening and if I take the negation, then what will happen? At least a phi 1 until phi 2 is going to be true it may be true provided it is going to have the second component, because now, here I am saying that phi 2 is false everywhere.

We consider another scenario what will happen, see somewhere phi 2 is true at that particular point, ok. If phi 2 is true at that particular point then what will happen? Then this property is not only over here, that means, in that particular case I cannot say that there will be equivalent. So, if such type of scenarios happens and if you say that in all the state phi 1 is true, then phi 1 until phi 2 is true at the particular path if it happens in all the this things all paths then a phi 1 until phi 2 is true.

So, in that case now, this is having a particular things now say, if all the (Refer Time: 32:27) now my phi 2 is say false, ok. But in some situation if phi 2 becomes true or say if phi 2 becomes true over here then what will happen a phi 1 until phi 2 is true because the semantics you just see what we are defining A phi 1 until phi 2 holds if for all path s 1, s 2, s 3 like that where s equal to s i then phi 1 until phi 2 satisfied, if there is some s i along the path such that M s i models phi 2 and for all j less than I we have M j equal to phi 1, ok.

(Refer Slide Time: 32:48)

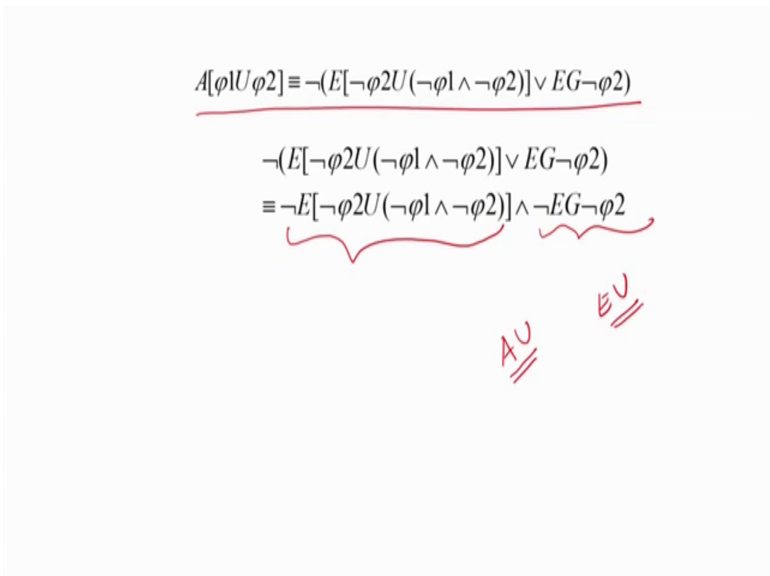


So, now in this particular case I am talking about some s i, and s i model phi 2. Now, what is the scenario if this s i equal to s 1, I am considering about this particular path and if this s i become s 1, then what will happen? So, if s i become s 1 in that particular path then we say the since phi 2 is true at that particular point I am going to say that a phi 1 until phi 2 is true at that particular point. So that means, if phi 2 is true over here itself, then I can say that phi 1 until phi 2 is true at that particular point.

So, what is the notion of times we are considering over here while defining the semantics? That means, future includes the present I think in the in some previous lecture I have mentioned about future includes present and future excludes presence. So, in this particular semantics the way we are defining the semantics what we are considering that future includes the present scenario also presence. So, if phi 2 is true over here then phi 1 until phi 2 is true at that particular point. So, that is why it says the not of phi 2 until not of phi 1 and not of phi 2 if there exist some sort of path and then phi 1 until phi 2 may becomes true, ok; A phi 1 until phi 2 may becomes true in that particular point that is why we are putting this particular negation.

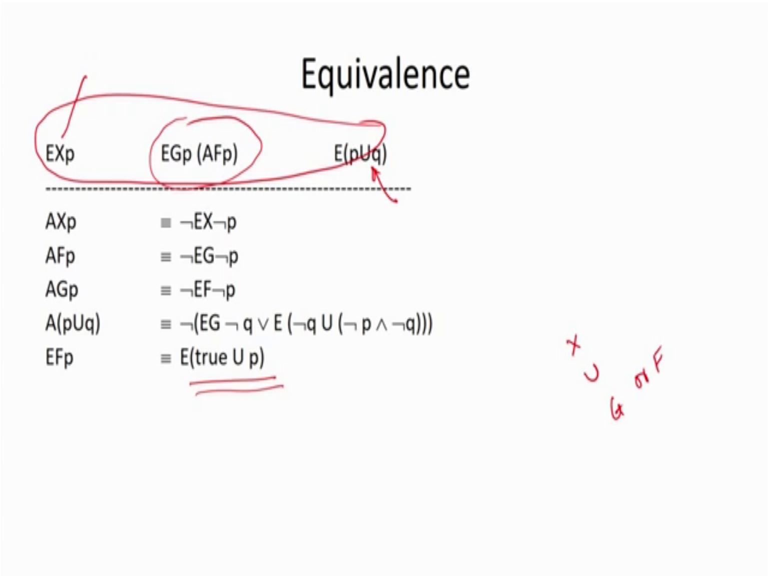
Now, if none of these components are true then we are saying that a phi until phi 2 will be true if the negation of this whole expression is true, ok. So, this is the scenario that we are having.

(Refer Slide Time: 35:11)



So, we are going to consider this is another equivalent that means, A until can be expressed with your E until, ok. So, this is the basic this things negation of these whole expression. Now, we can apply de Morgan's theorem that means, negation of this component and this component, negation of the other component. So, we are getting an equivalence relationship between A until and E until. Now, since until can be expressed with E until or E until can be expressed with A until. So, we are going to get different set of adequate set of operators in case of CTL. So, that is why we can have now one adequate set like that.

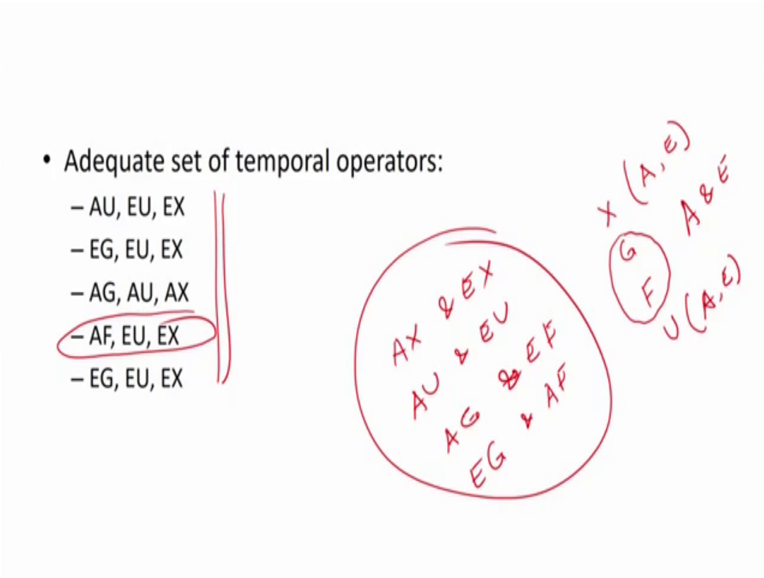
(Refer Slide Time: 35:56)



Here X we cannot avoid because what will happen AX can be expressed with EX then until either we can take both until already we have said or we are going to take one until say if you are taking E p until q, then one of the operators I have to take from either future or global. So, you just we need one next state of operator, we need one until operator and we need any one of global operator or F operator.

So, if I consider any of these 3 operators then it is going to give me an adequate set of operators because already I said that AX can be replace with your EX AF can be replace with your EG. So, either AF or EG I am going to take and AG is again I replaced with the help of EF and until A until can be expressed with the help of E until, and E p can be replace with your again until operator. So, this is the way you can form and you can now, construct different set which will be adequate to express all the CTL operator. So, these are some example of the adequate set of CTL operator.

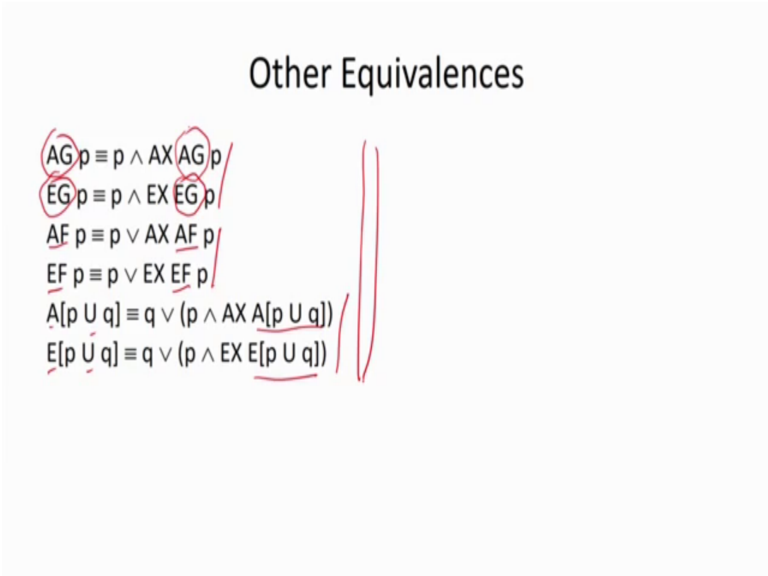
(Refer Slide Time: 37:25)



Now, if we know the method of finding the truth values of a CTL formulas say with the help of these 3 operators AF, EU and EX then other operators can be derived with the help of these 3 operators. So, that is why we say this is the adequate state of operator. So, what we have seen? We are saying that we are having operator X next state global F and until, ok. We have seen that global and F is having one relationship, global basically say in all states along that path and if future says there exist at least one state in future along that particular path. So, we are having a relationship between EG, G and F along the path quantifier A and E in all path and there exist a path.

Similarly, X is having a relationship for this all path and there exist a path and E is also having a relationship between A and that means, I am going to get an relationship between AX and EX, and AU and EU, ok. But AG is having a relationship with EF, and EG and AF is also having some ignorance relationship. So, with the help of these things we are going to find formula equivalence between this operator, and with the help of these things we can now, find the equivalence CTL formula.

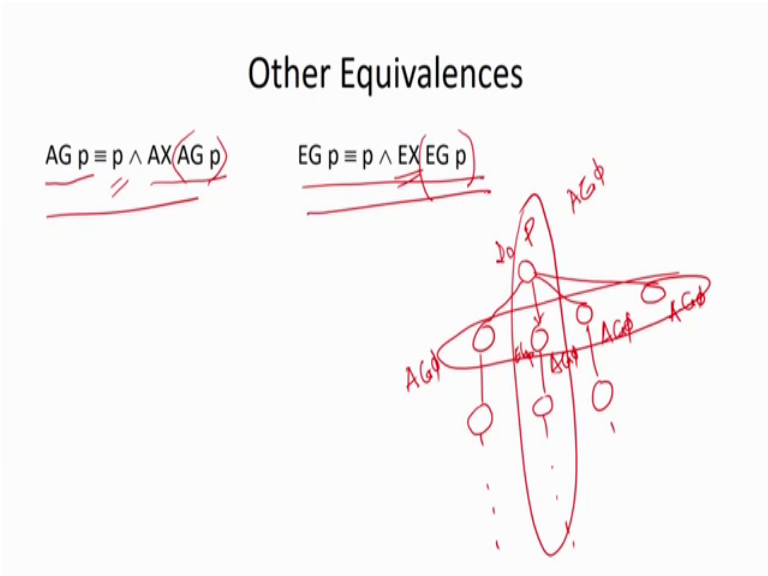
(Refer Slide Time: 39:35)



The notion of equivalence can be looked from the meaning of those particular CTL operator, ok. So, these are some other equivalence that we are having. So, what is this one? Basically these are some sort of expressing the operator with the help of the same operator to get some sort of equivalence. So, you just see AG is representing with the help of EG, AG only, EG is again representing with the help of EG similarly AF, EF, A until and E until. So, this is A until and E until.

So, now these equivalence basically give us the idea or a meaning of the given those particular CTL operator. Now, what it says? First say, so I cannot consider about this is one state, second set and third state.

(Refer Slide Time: 40:40)

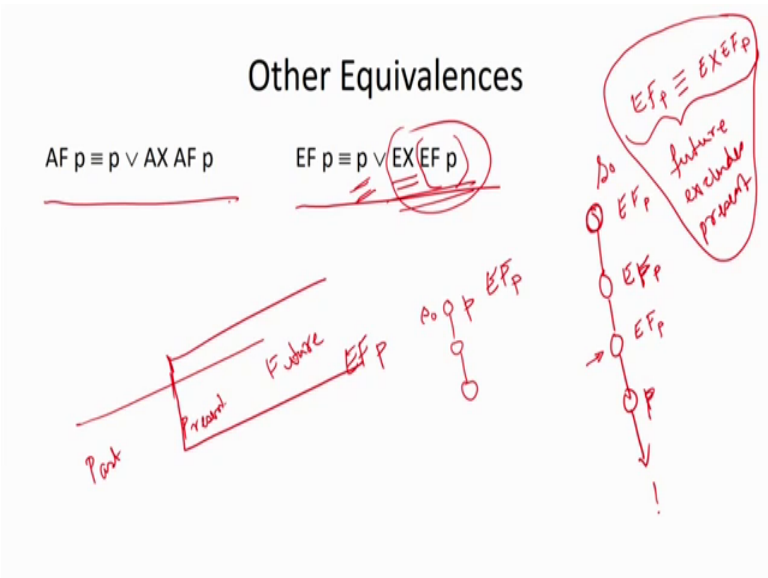


Now, what the first state says AG p is equivalent to p and AX, AG p or EG p is equivalent to p and EX, EG p; you just consider something like that say. So, it says the globally something has to be true, so that means, globally in every state of the model, wherever you go this p must be true that means, if I am going to consider in this particular state s 0, ok. It says that if p is true at that particular state then you can consider but this is not the only condition we are having second condition also what it says? In all path in next state whatever is the next state we are having state should also satisfy this particular AG phi property. This is this must also satisfy is AG phi property, ok.

Now, it is a big model say we are having lot of transition, ok. Now, what will happen we can talk about the truth values of this given formula either AG p or EG p, in a particular state by looking into its the next state behaviour. If next state if we find that AG phi is true, in all the next state then we can say that AG phi is true in this particular given state also. Similarly it holds for EG p also, in EG p we will concentrate look for any one of such type of path if p is true over here, and the next state also satisfies the EG p is true there exist a path in next state EG p is true then what we say EG p is true in that particular state.

Now, what basically it says? When I am going to talk about the global operator and I am looking for the truth values of this particular global operator in a particular state then by looking into its next state, we can find out the truth values of this particular global operator in a given state. So, that is why I just saying that, AG p is equivalent to p, p must be true in that particular state and in all path in the next state AG p must be true, similarly in all path next state EG p must be true if you are going to talk about EG p. So, that is why these are the two equivalence that we have in our CTL formula.

(Refer Slide Time: 43:54)



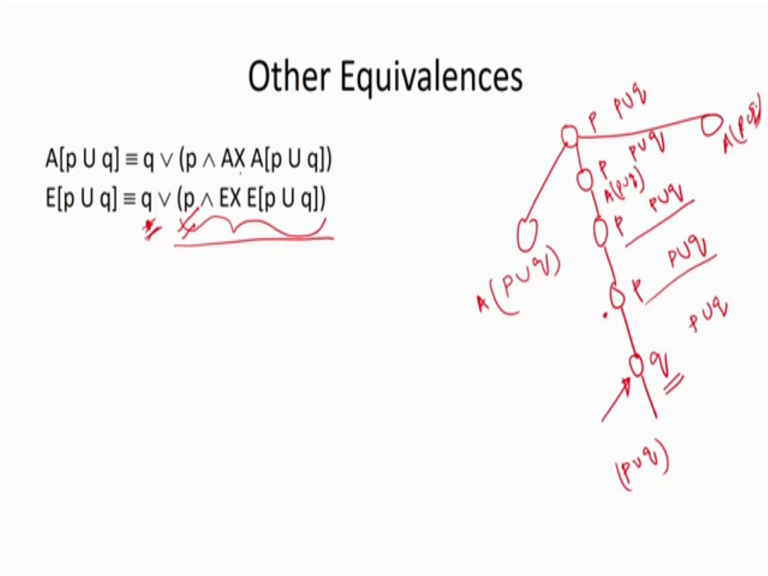
Now, consider a second state it says that AF p is equivalent to p or AX AF p and similarly EF p is equivalent to p or EX EF p. Now, what it says? If p is true then I can say that AF p is true over here because in future I am getting going to get it then EF p is true here and finally, I can say that EF p is true here. Now, what it says? So, if I am going to look into it then if there exist a path so that EF p is true then I can say that EF p is true over her, ok.

So, if I am looking into a particular state s 0 and if there exist a path in the next state EF p is true then you can say that EF p is true in this particular state. So, this is the component that we are having EF p is true in a particular state if there exist a path in next state AF p is true. So, if I can say that if p is true over here in this particular state EF p is also true, ok. So, like that EF p is equivalent to p or EX EF p. So, this component is clear to us.

Now, what about the other component? If it says that if p is true here itself then we are going to say that EF p is true. This is as for our different semantics. How we are defining our semantics? How we are considering the time? We are saying that future includes the present. So, we are having a timeline, if this is the present then here I am going to get past in this direction and here I am going to get future. So, when we are going to consider about the future behaviour it includes the present also. So, this is the way we are defining our semantics future includes the present, ok. So, that is why if p is true here itself in a particular state s 0 we say that EF p also true here itself.

Now, if we want to exclude present from the future behaviour, then what will happen? We have to slightly modify our semantics, we have to redefine our semantics, it is possible you try to look into it how we are going to define a semantics such that future exclude the present behaviour. But in our discussion the semantics is define in that way that future includes the present behaviour; so, due to that this is coming over here p or EX EF p, is equivalent to EF p. But if future exclude the present behaviour then what will happen, we can say that then in that particular case EF p will be equivalent to EX EF p. When we say these two are equivalence if in our semantics we say that if future exclude present then we may get such type of equivalence, but in our case in our semantics what we are considering future includes the presence. So, that is why we are having these two semantics.

(Refer Slide Time: 48:01)



And similarly these are the last pair of equivalence where it is defining until in terms of next state. So, first we consider about this particular component. So, when I say about that p until q, then p must remain true until q becomes true. So, if this is the scenario then what I am going to get, here I am going to get p until q is true because p after a q, p p remains true then after I am going to get p until q is true over here p until q is true here and p until q is true here particular point this is the scenario.

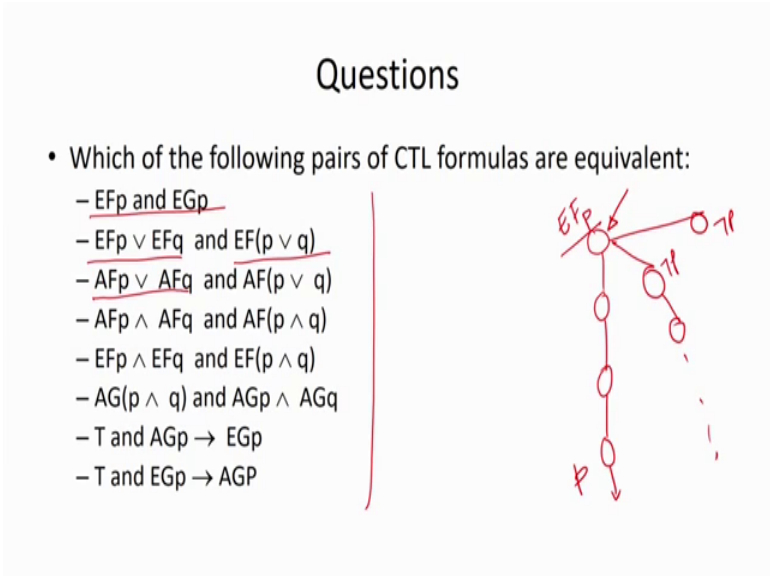
Now, what it says? If p is true, and there exist a path in the next state E p until q is true then we are going to consider E p until q will be true at that particular point, ok. Now, this is the scenario. Similarly if here I am going to say p and p until there exist a path in next state p until q is true then I am going to consider p until q is true at that particular point, ok. Now, what is this particular component q? If p is true and there exist a path in next state p until q is true, then we are going to say that p until q is true at that particular point, ok. So, similarly we are going to say like that.

Now, in next state where a p until q is true or not in this particular scenario what is the status of p until q. So, here, now as per this definition if q is true in a particular state we are going to say that p until q is true over here. Why we are considering these things? Again as per our defined semantics and in the defined semantics what is the notion of time presence include the future, present includes future. So, that is why if q is true here itself then I am going to say that p until q is true in this particular state. So, that is why this component is there.

So, in this particular equivalence what we are saying that E p until q will be equivalent to q, if q is true in that particular state itself or it must satisfy the other condition p must be true in that particular state and there exist a path in next state E p until q is true, ok. So, similarly for A also we can say that A p until q will be true in a particular state provided q is true in that particular state or if q is not true then we are going to look for the other combination other path p must be true in that particular state and in all path in next state A p until q must be true. That means now, if you consider all path if p is true over here and in all next state A p and q is true then we say that A p until q is true over here. So, p must be true and we must have a scenario like that in all path in the next state A p until q is true, ok.

So, these are the equivalence that we are having we can represent in temporal operator with the help of which next state behaviour. We are defining the temporal operator by itself which is next state behaviour. And this equivalence are going to help us to build a method to check whether a given CTL formula is true in a particular state or not. We are going to use these particular equivalence these are the equivalence that we will be using to find out the truth values of a particular formula in a given state.

(Refer Slide Time: 52:27)

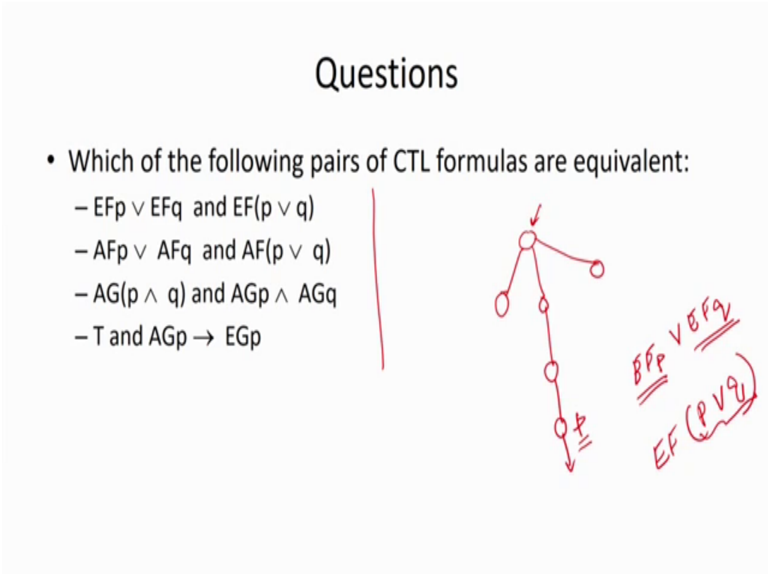


Now, some of the pair of equivalence pair of formulas I am just writing over here EF p and EG p, EF p or EF q and EF p and q. So, these are the pairs of formula. I am writing and we are just going to check whether these two formulas are equivalent or not whether these two formulas are equivalent or not.

So, what is the first one? EF p and EG p, there exist a path in future p holds and there exist a path globally p holds, ok. Now, you consider any model, this is very simple actually say p is true at the particular point, ok. So, there exist at least one path where EF p is true, but in this particular model in this particular state EG p is not true we are not going to get any path where globally p is true say if I say that this is a not of p not of p like that, but EF p is true. So, one is true other is false. So, they are not equivalent.

Now, similarly now can you check whether AF p or AF q and AF p or q you can check. So, basically what will happen? If they are equivalent then we have to argue with the meaning of these operators if they are not equivalent then try to give a counter model where we can say that in one state in a state one particular formula is true other formula is false, ok. Now, if I look into it now, I am just taking some subset, ok. So, if I look into those particular subset, then what we are going to get?

(Refer Slide Time: 54:34)

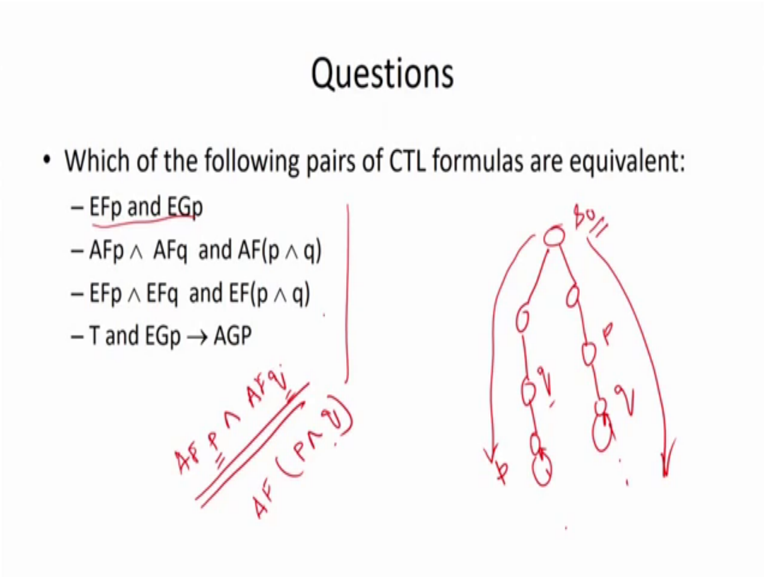


So, here I am writing some formulas but I am taking of some smaller state over here. Basically if you now, try to look into those particular formula you will find that these formulas are equivalent.

Now, what I can say for the first one? Say this is a formula either p is true over here then what will happen; there exist a path in future at least we are going to get EF p is true or EF q is true whether it is true or false (Refer Time: 55:25) it is (Refer Time: 55:26) false because one component is true, so I am going to get that EF p is true at that particular point. Similarly EF p or q, so since in this particular path somewhere p is true that means, in this state p or q is true. So, EF p or q will be true over here.

Now, by looking into the notion or meaning of this particular operator or an future we can find that these two formulas are equivalent where whatever model you are going to consider if one is true in a particular state then other will also be true, similarly we can look for other one.

(Refer Slide Time: 56:05)

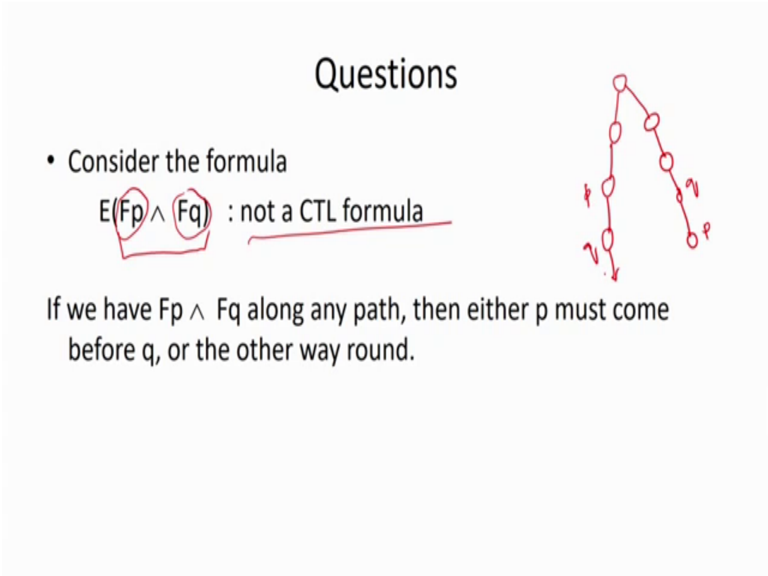


So, other state we are going to consider over here and here we will find that these are basically not equivalent. Already I have discussed or said that these two are not equivalent because in future p may be true but it may not be true in all the state of any path, ok. So, they are not equivalent. So, one may be true but other is false. But if EG p is true then EF p will be also true we do not have any problem about that, but if EF p is true it does not guarantee that EG p is true.

Now, what about the second formula? AF p and AF q, and AF p and q they are not equivalent and to establish that not equivalent what we can say we can give a counter example over here. Now, I can talk about one counter example consisting of say two path only I am not going to consider more path. If this is the scenario now, what is the status of this particular formula? Now, what is status of this formula? AF p and AF q so, if you consider this particular path somewhere in future p is true somewhere in future q is true.

So, AF p and AF q is true in this particular path. Similarly if you consider this particular path AF p is true and AF q is also true, ok. So, if this is the scenario then what will happen? I can say that AF p and AF q is true in this particular state s 0. But, what about other component? AF p and q, so here we do not have any state where both p and q are true, that means, AF p and q will be false. So, if I consider this particular state s 0 of this particular model then one is true second one is false. So, that is why we say that these 3 formulas are not equivalent. Similarly now we can check for the equivalence of the other formulas also. So, consider this particular formula EF p and EF q, ok.

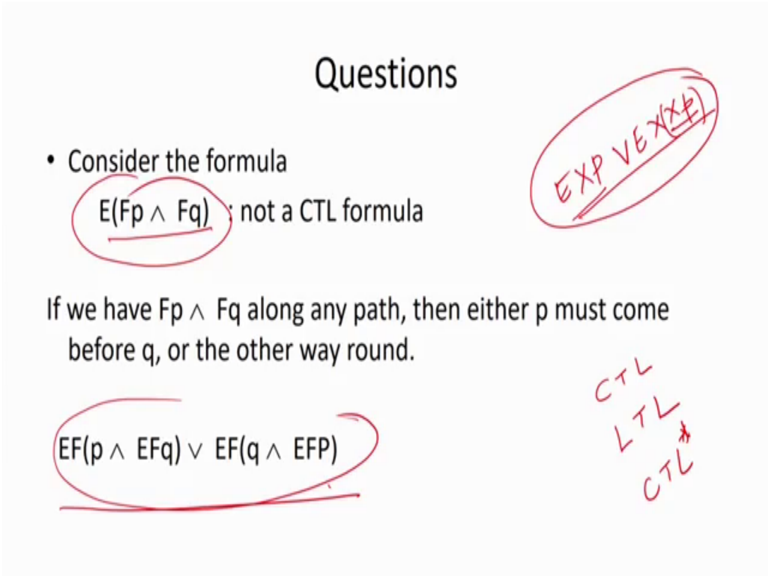
(Refer Slide Time: 58:33)



Now, if I look into the formula then what I will find that this is not a CTL formula. Why it is not a CTL formula? F q and F p they are not preceded by any path quantifier but their conjunction is preceded by a path quantifier E, ok. So, in that particular case this is not a CTL formula. But in this particular case what will happen if we have EF p and EF q it says that along any path then either p must come before q or other way around. So, either p may come first or q may.

So, basically in this particular case what will happen? I can say that p is true over here and q is true over here or it may happen that q is true over here p is true over here. So, there exist a path either F p is true and F q is true. So, here also at least in this particular path we can say that this is true or may be here also true. So, this formula now, we have seen that there exist a path either p may come first before q or q may come before p.

(Refer Slide Time: 59:56)



So, these things can be written as like that we are exist a path in future p and EF q, p and EF q or there exist a path in future q and EFP. So, if this is happens then what I can say that there exist a path in future p, F p and F q. So, no this is not an CTL formula but this formula is equivalent to this particular formula, and if you look into the notion you will find that this is a CTL formula, ok.

So, just I am showing one example but it is not always possible to express this particular any formula to a CTL formula. So, these basically talk about the expressive power of my logic or CTL. So, what will happen? We are discussing about a CTL computational tree logic but all cannot be expressed in CTL. So, it says about the expressive power of that particular logic.

So, here we shown that these particular formulas can be expressed with this particular formula, but all may not be possible. Like that if I say that, there exist a path in next state p holds are there exist a path next to next p holds. So, this is also formula but you just try but it will be difficult to express these things with the help of a CTL it is not a CTL formula because due to this part this particular operator temporal operator it is not preceded by any path quantifier, but if I put another path quantifier then the meaning will be different, ok.

So, it says along this particular path but it may happen in different path also but such type of formula cannot be expressed with our CTL. So, that is why expressive power basically says: what are the properties that we can specify in our logic. So, we have discussed one logic called CTL computational tree logic. Due to its expressive power now, researchers are looking for some other way to construct the formula temporal formula and for that we are getting different class of temporal logic formulas.

So, one here we are talking about CTL. So, like that people are considering about another way to do it which is known as your LTL, computational tree logic, Linear Time Temporal Logic. Like that we have come up with some formula or logic called CTL star also. But in this course we are not going to discuss about just saying that we are having a varieties of temporal logic families and the expressive powers are different for different families, ok.

So, with this I am going to end my lecture today. So, we will continue for other issues in the next class.

Thank you.