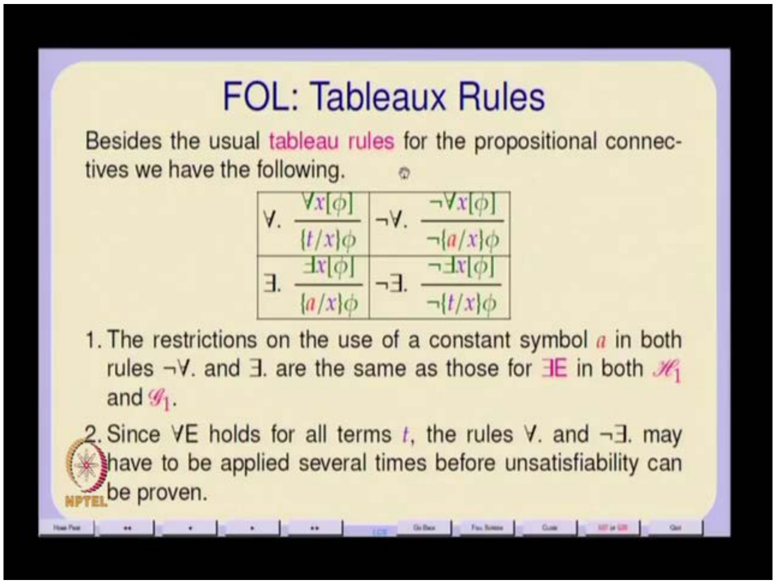
Logic for CS
Prof. Dr. S. Arun Kumar
Department of Computer Science
Indian Institute of Technology, Delhi

Lecture- 33
Completeness of Tableaux Method

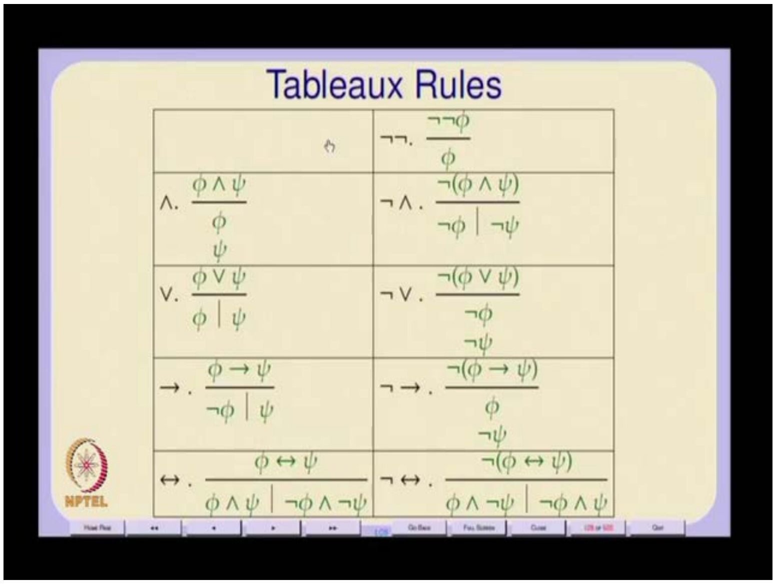
Today, we will complete the Completeness proof for tableau. And, next week I will do the completeness for Hilbert set of systems and then, if there is some time I will do some first order number theory. So, that we will get some you get some more holistic picture of the application of first order theories and I mean so. But, I do not know whether I have all that much amount of time to do full first order number theory ideally. I would have like to do the hurdles in completeness theorem also but, that is obviously not going to be possible. So, in a certain sense all these completeness theorems for first order logic actually come from they had derived this essentially from the hurdles original proof of completeness of for the Hilbert style proof system. But, anyway we will some of these completeness theorems sometimes now can be expressed in the independently of hurdles original proof so we will proceed with this.

But, before we proceed with this completeness. Let us just look at an example.

(Refer Slide Time: 01:47)



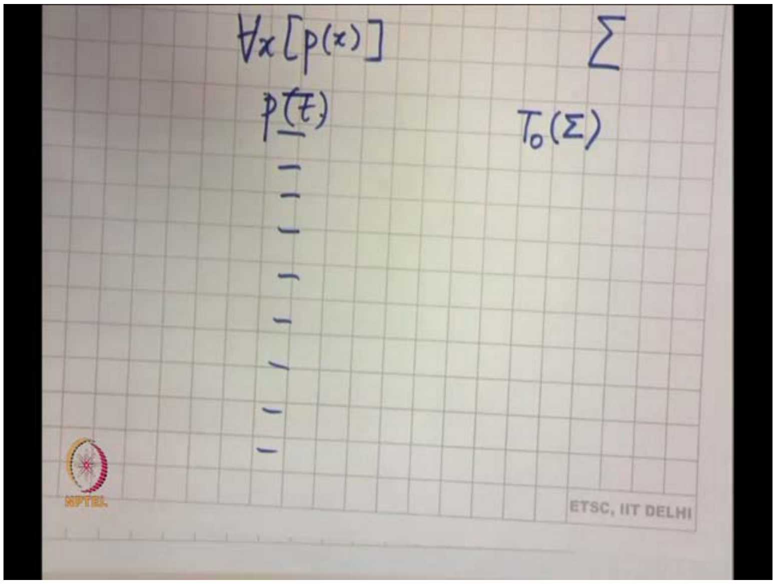
(Refer Slide Time: 01:51)



So, these were the tableau rules so you are the usual tableau rules for propositional statements. And then, once you have tableau rules for the proposition connectors these are the tableau rules for the quantifiers and their negations. And of course, as I said there is a restriction on the use of these constant symbols a. In the original work on tableaus created by a smullyan we actually, call them parameters but anyway we are treating it as constant symbols.

But, they should be new I mean that is the property that should be there and so those restrictions are exactly the same as in there exists elimination in both the Hilbert style proof system and in the Kinston style natural deduction system. The other thing is of course, is really this that theoretically any satisfiable formula so if, you take some if I just take some unary predicate p atomic predicate.

(Refer Slide Time: 03:08)



And if, I were to take px then for all x px and I consider some signature sigma. Then, theoretically what this rule says actually is that you have to take the entire set of all ground terms and of course we are implicitly assuming that there is at least one constant symbol in the signature otherwise this set would be empty.

So, you are implicitly assuming that actually this entire infinite in some order all the elements in t naught sigma replace x to give you some p of x. So, if you look at so theoretically what we are saying is that this is the tableau, this infinite path is a sort of tableau for this predicate right. And however, of course from an implementation point of view it we are not interested in those infinite paths basically. And, as that is one of the reason why what we want to do is we wanted to use tableau proofs in a way we the same way we use resolution.

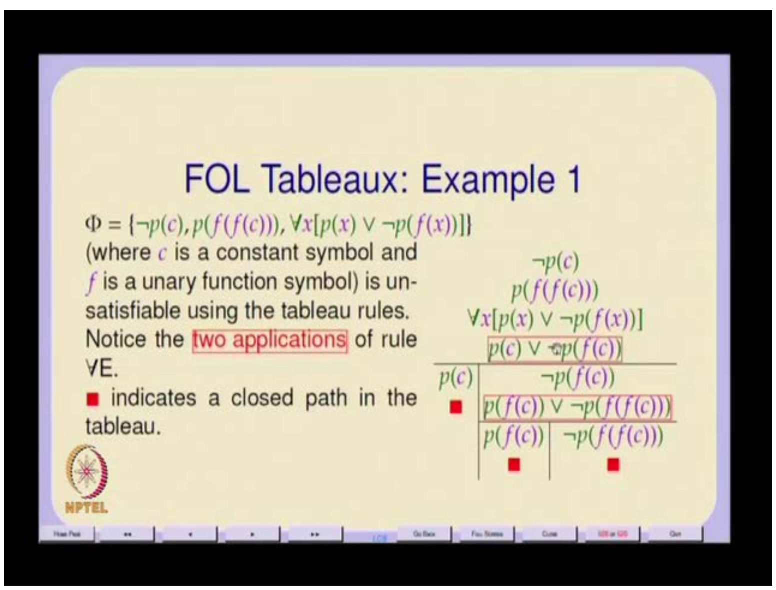
So, you try to prove contradictions and when you try to prove contradictions basically what you are trying to do is to prove unsatisifiability. And, what we know from compactness theorems and so on is that thence that those tableaus will then be finite. And all then there should be closed tableaus and all the paths would be closed and they would be finite tableaus and that is what you are using.

But, actually I do not have the time to do decidability issues but this problem is related to the fact that the validity in first order logic is actually un-decidable. Which, means that there is no algorithm which for any formula in first order logic which given an arbitrary formula in first order logic will tell you whether it is logically valid or not.

So, the net effect of that is that potentially so it is actually semi-decidable so if it is valid then it will find you can find a proof for it. But, if it is not valid you do not know if it does not it can possibly not terminate and you will not know whether it is going to become valid later. So, as a result what it means is that theoretically that issue is related to this issue of an infinite open path in a tableau.

So, theoretically speaking you cannot you cannot guarantee algorithm weakly that you will always be able to prove that your tableaus are finite. And, there are always closing right for all unsatisfiable formulas but, so that is why we use unsatisfiable we use these constants essentially to direct the search on an demand bases. So, we instead of directly extending the tableau by an infinite amount we are doing that is something that is acting to what is lazy evaluation in programming languages.

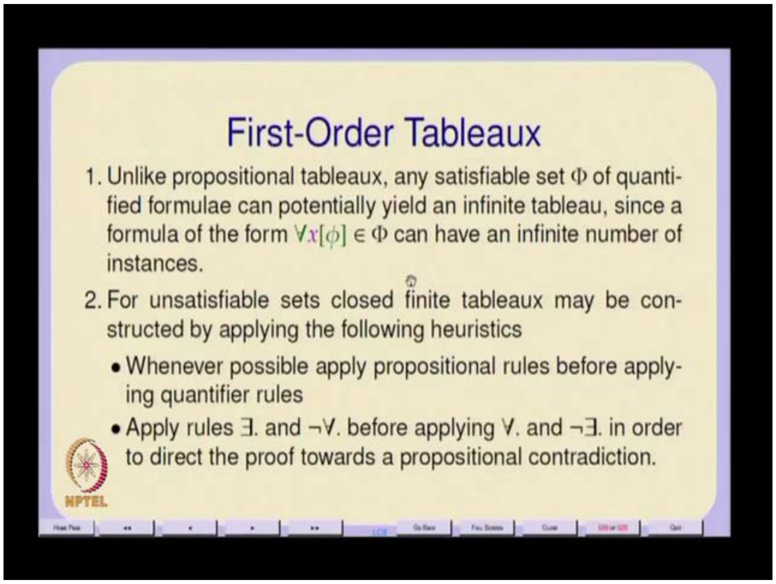
(Refer Slide Time: 07:09)



We wait till a demand is raised in order to get an instantiation that is important and that is actually illustrated by this example. Where by using this constant at this point a demand gets raised for an instantiation of the universal quantifier. And at this point another demand gets raised so in every open place a demand gets raised and so you will sort of work on it lazily on an in demand basis.

It very similar to the implementation of lazy functional languages where you do a call by need as different from you know any other call by any other mechanisms. So, you do a call by need and this is what you use as a heuristic as an important heuristic to ensure that you get finite tableaus.

(Refer Slide Time: 07:59)

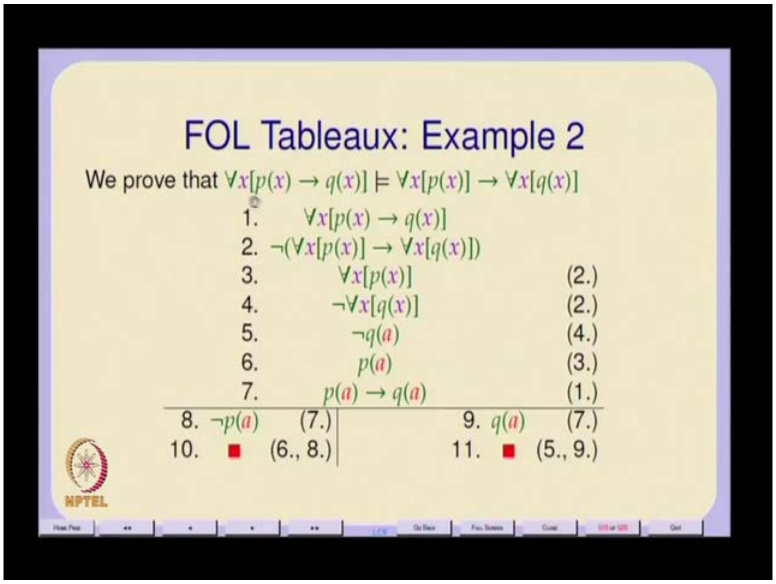


So, the other thing is that in the case of propositional tableaus as I said to your formulas are all can be one they broke once they are broken up the original formula can be thrown away. Whereas, because of this particular case and of course whatever I say for the universal quantifier also holds for the negation of the existential quantifier and so the same arguments of.

So, unlike the proposition case therefore all these universal universally quantified statements and negations of existential quantifiers which are the same as the universal quantifiers are actually need to be reused several times. Basically, each time you will come up with a different instance from this set T naught sigma. And, in order to get ensure closure of the tableau for example especially for unsatisfiable sets. So, for unsatisfiable sets are closed finite tableaus maybe constructed by basically following these heuristics this ensures that you are tableaus are so one thing is that you. Whenever, possible you apply propositional rules so that can happen only if the proposition if there is a formula whose root formula is root operator is a proposition connector.

So, basically what we are saying is break up all the propositional break up all the formulae's which have propositional connectives in the root and then get on to the quantified formulae. And in order to ensure that you get a directed proof and needs are raised demands are raised for particular instantiations it is better to first apply the rules for the existential quantifier then negation of universal quantifier. So that, will automatically raise demand for an appropriate instantiation of universal quantifier. And, so this way you hope that you will direct the proof towards a propositional contradiction.

(Refer Slide Time: 10:14)

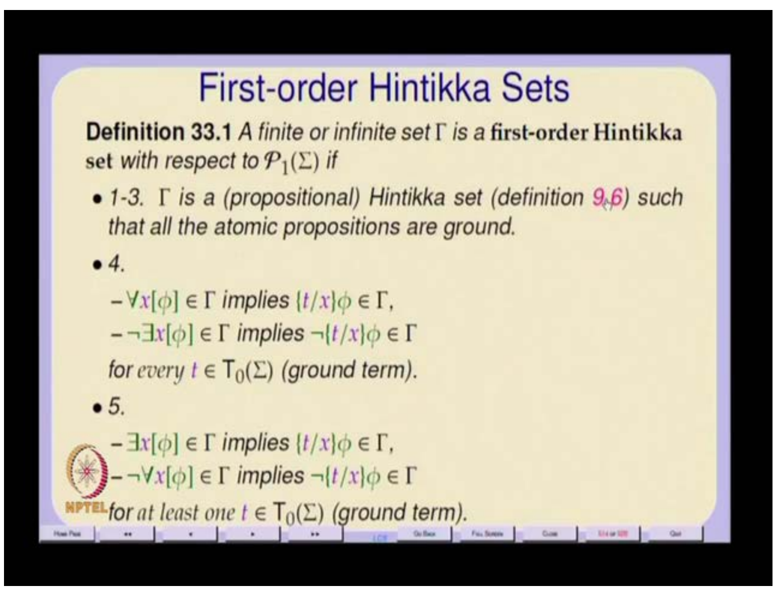


So, here is one example so here I take this. This says that the universal quantifier distributes over arrow basically right. And so well only one way I mean the converse is not true so if you take this so a typical closed tableau will essentially I rewrite this formula right here. I take the negation on this formula clearly now actually there is nothing to do except.

I have a choice between these two applying a universal instantiation and breaking up this negation somehow this is actually naught of arrow right I means you can think of naught of arrow is a single arrow operator. And of course, I choose to break this up by according to my heuristic. I also choose not to do anything with this because, I still do not know what they demand and for a term would be. And so when I break this up I get for all xpx and I get naught of for all xqx and then. And now, it is clear this is an naught of for all so I have basically so this step two being propositional can essentially be disposed off.

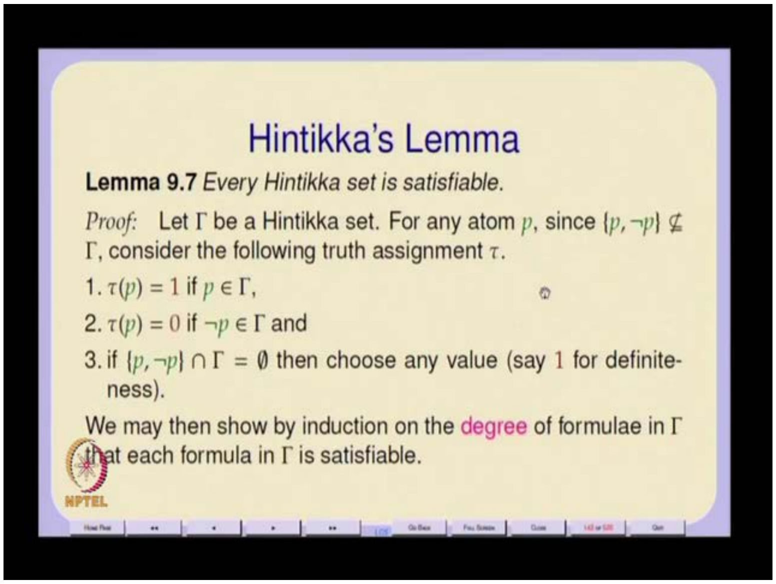
So, actually what I have this universal quantifier and essentially this existential quantifier which by my heuristic I get naught qa. And that raises the demand for this exist for universal instantiation so which for example, use me the p of a right. And given this p of a then it also gives me this raises the demand to initialize this to p of a arrow q of a. Which, is a branching operation is a branching connective so that gives me naught p of a and of course there is a contradiction here. And then, it gives me q of a there is another contradiction here and so there the tableau is closed. And so this way you sort of ensure that you get in the case of propositional logic we were interested in slim tableaus. Here, you are interested in well the tableau actually being finite I mean that is the first criteria in and slimness can come later that is a second priority. So, that is how these tableaus methods work and of course we will follow what we did for propositional logic in order to trying to prove completeness.

(Refer Slide Time: 13:12)



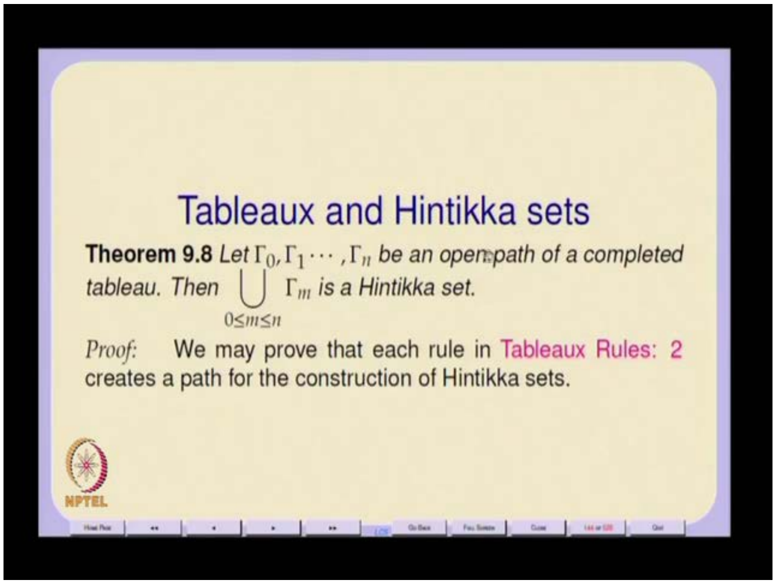
And one of the things we did was we constructed let us just briefly go through it. We constructed this motion of this Hintikka Sets. And essentially the propositional Hintikka set is closed over tableau inference. So, if you look at these inclusions in the Hintikka sets every subsequent application of a tableau rule essentially gives you to another member gives you the closure for the Hintikka set.

(Refer Slide Time: 13:51)



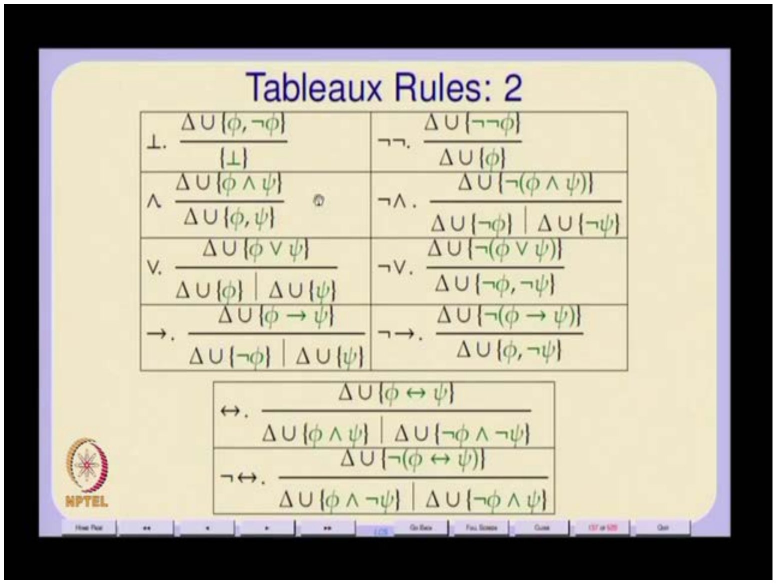
So, and in fact so one possible one thing is of course is that we had this Hintikka's Lemma which said that every Hintikka set is satisfiable.

(Refer Slide Time: 14:02)



But, the more important thing is really that you take any open path in a tableau. Take all just collect all formulae in that open path and that will be a Hintikka set and the intuition is clearly that the construction of Hintikka sets essentially follows tableau rules.

(Refer Slide Time: 14:23)



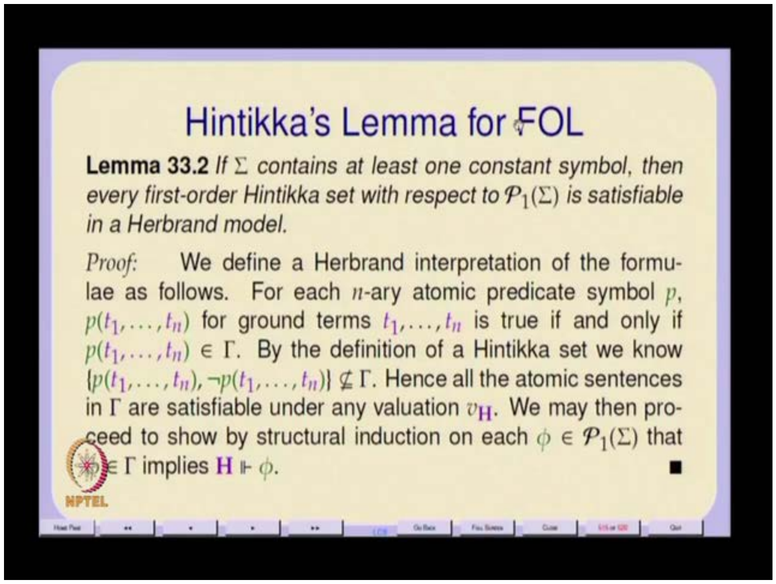
Which, is like so what you do is. So, for example if so this is like a take this tableau rule if you have the formula phi and psi then you include the sub formulae phi and the sub formula psi also in the set so you close it. So, it is all if all the closures work from the hypothesis of a tableau rule to the conclusion so it is natural that the construction of Hintikka set therefore is essentially of preserving open paths. So, if you take the set of all formulae in an open path of a tableau which and an open path is one that was there is no contradiction. And therefore, it is satisfiable and then you will get in if you take the complete open path you will get a Hintikka set.

Now, exactly the same thing holds for Hintikka sets of first order logic except that it is going to be infinite. So, any Hintikka set which contains a universally quantified formula is going to be an infinite set because the entire the set of all ground terms is also going to instantiations of the universal formula for all ground terms is also going to be included. So, while it was possible in propositional logic to have finite Hintikka sets in predicate logic the moment you have a universal formula the negation of a of an existential formula you are going to have only infinite Hintikka sets. So, it in the case of propositional logic your tableaus could have closed paths and open paths and everything would be finite. Here, what you are going to get are tableaus which in general if they are not propositional in general are going to be infinite. And, your open paths are going to be essentially infinite paths except in the most trivial cases.

So, what will do is so the notion of the Hintikka set is extended to the notion of first order Hintikka set with respect to this language. So, of course this 1-3 is all those propositional Hintikka set definitions right. So, you take all the take the propositional so every Hintikka set has to be a propositional Hintikka set in the sense that it will be closed according to those rules of this definition and such that all atomic propositions are ground.

Then, you have essentially the quantifier rules which say that now this says that for any universal quantifier or negation of existential quantifier every ground term instantiation that is possible should be included in this set right. In the case of the existential quantifier and negation of the universal quantifier or you are asking of course that there should be at least one term which is ground which should be included in it in the set for the set to be a Hintikka set. So, they of course what happens in practice is that this may not actually be since because of the use of constant this term would be some formation based on those constant so, it is not always going to be a constant as this thing shows right this example shows. So, for example some may have to have things like ffc for example the second instantiation so it bridge so there would be a some ground term that is what first order Hintikka set has.

(Refer Slide Time: 18:37)



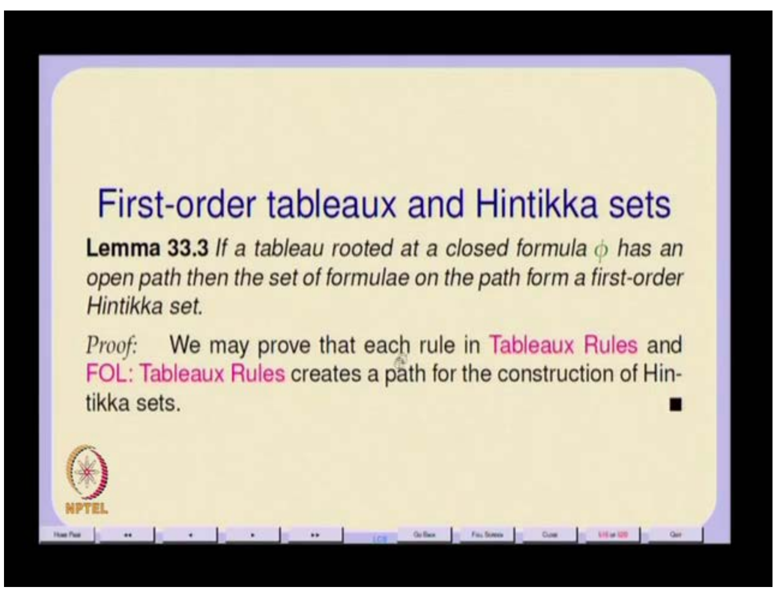
So, then for Hintikka's lemma for first order logic now of course in the case of propositional logic we just had truth valuation. But, in the case of predicate logic you have this complicated notion of models which are all brown color in my slides. And, you do not know what kind of models can be created it is all too complicated the only concrete things that are available therefore are Herbrand's theorem. And, what Herbrand's theorem gives you are essentially these concrete things like these ground terms. So, Hintikka's lemma for first order logic essentially says that if sigma contains at least one constant symbol then, every first order Hintikka set with respect to this is satisfiable in a Herbrand model that is all you need to show what we know from Herbrand's theorem is that. If, there is a Herbrand model then there is a model and are other models. And, what Herbrand's model also shows is that in order to show the existence of a Herbrand model you need to consider only is a ground clause ground instances right. So, it is that is that really all that we need to do so in fact what we are going to do is we are going so we will only look at Hintikka sets. Which, create a ground Herbrand model and you can include the word ground here or you do not need to include it by Herbrand's theorem they both are equivalent.

So, what is it? It is very simple so all that we need to do is define a Herbrand interpretation. Where value and valuation is of course in the Herbrand interpretations are substitutions of just free variables. But, since we are going only to consider ground terms so it is only going to be ground substitutions. And, we do not need to do anything really about the terms all that we say is so supposing you have a first order Hintikka set gamma. Then, if it is a Hintikka set and if it was not just purely propositional then it would have some ground predicates.

And if, it has some ground predicates then it would have ground atomic predicates also that by definition it would have to be closed under all those things right. Now, look at all the ground atomic predicates and interprets your valuation in such a way that exactly those atomic ground predicates are true. That is it once you have done that all other predicates in the Hintikka set will be true. And that by definition of firstly by the definition of the Hintikka set and secondly by structural induction on the motion of truth.

So, Hintikka's lemma of first order logic essentially shows that if I were to take all the ground atoms given a Hintikka set gamma. If, I take all the ground atoms and make them true and all ground atoms which are not there in the Hintikka set make them false. Then, I can guarantee basically the structural induction that all the formulae all the ground formulae in gamma are true that is your interpretation that is it. It is a trivial interpretation. So, then basically what you can do is you can proceed to show by structural induction on every formula. That if, this formula does belong to gamma then the particular Herbrand then this formula is satisfied by a Herbrand interpretation so that is what Hintikka's lemmas for FOL says.

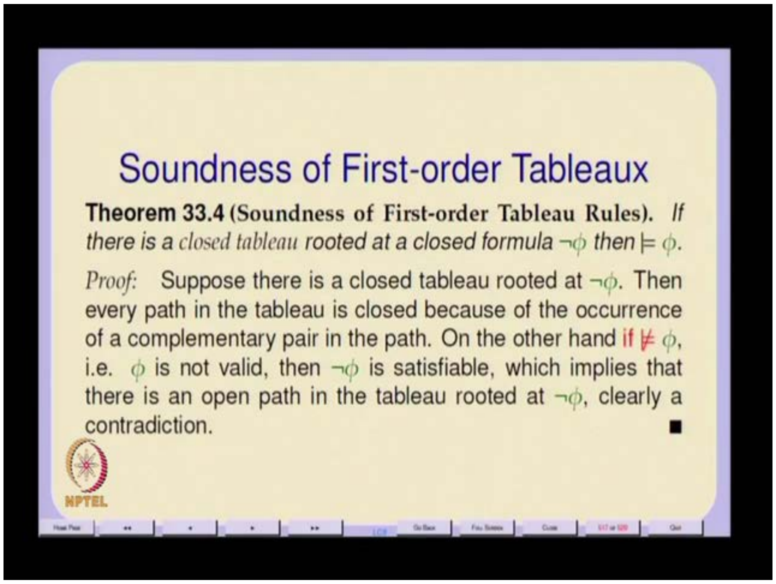
(Refer Slide Time: 23:02)



And then, you have a corresponding first Hintikka theorem for tableau first order tableau and which is just that if let us just go back were you remember we have this. So, this is the lemma we have just done is corresponds to this one every Hintikka set is satisfiable. And, then there is this theorem which says take any open path of a completed tableau and collect all the formulae in this open path and what you get is a Hintikka set. Of course, I proved this in terms of having sets of formulae on the path. But, that was because I was using these tableau rules the second form where we considered sets we could have used the first form which was like this in which case there would be one formula at each node of the tableau and you would just collect the entire lot either way this theorem this could have been proven where is it.

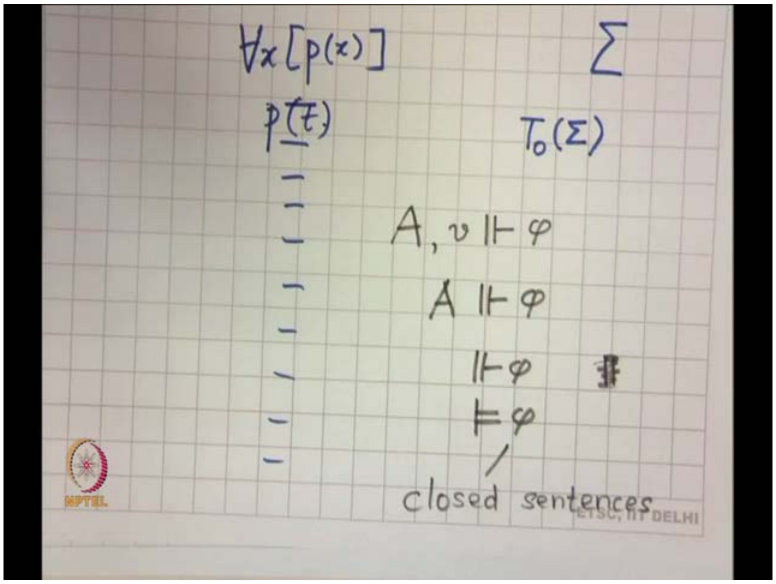
Now, either way this could have been proven and it would have essentially every open path would be a would have an Hintikka set. But, coming back to the corresponding theorem for first order logic you can just prove that each rule in the each propositional rule and each first order logic rule. Basically, these rules and this rule these two these four rules essentially create a path for the construction of a Hintikka set. And if you have universal quantifier basically that open path could be an infinite path if you have a universal formula.

(Refer Slide Time: 25:25)



The Soundness of the tableau method were it has basically what remember that we are not looking at so we here we are looking at logical validity remember that. Because, we have various notions of validity if you recall.

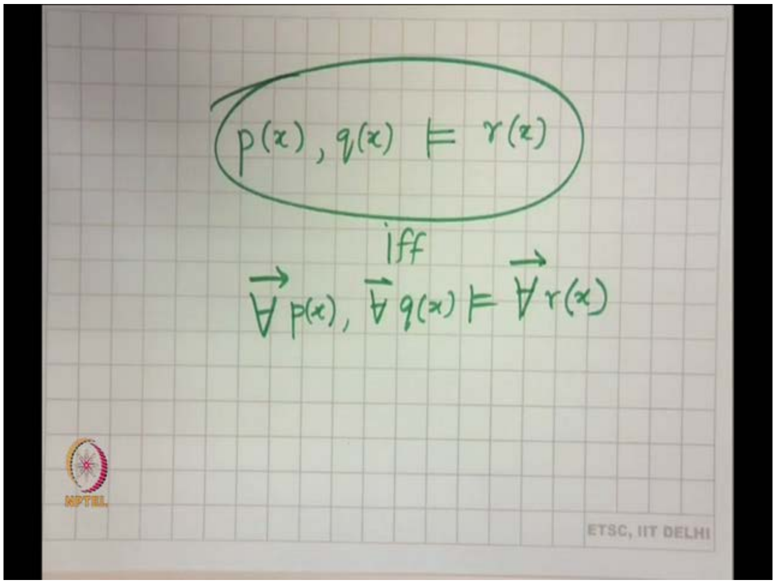
(Refer Slide Time: 25:42)



One was you have this structure and a valuation and you are asking whether the formula is satisfied with the valuation. Then, next one was you have this structure and you are asking whether this formula is valid for all valuations and the third one was that regardless of the structure whether the formula is valid. And, what I showed you what I told you at some point was that this is what this the same as this logical validity. And, what we also know is that here so we looking at logical validity and if has to be independent of all these valuations and so on so forth. Then, unless it is trivial you are essentially looking at closed sentences.

So, you can think of it as essentially sentences without free. So, that includes pure propositional sentences without it includes all the ground things, ground purpose ground predicates. It includes propositional combinations of ground predicates it includes quantified predicates which might have some which are variables in them but they should be closed. So, there are no free variables in them so you look at all those sentences and actually are all are notions of soundness and completeness are essentially of that kind. We are not taking an arbitrary formulas with free variables and looking at their validity.

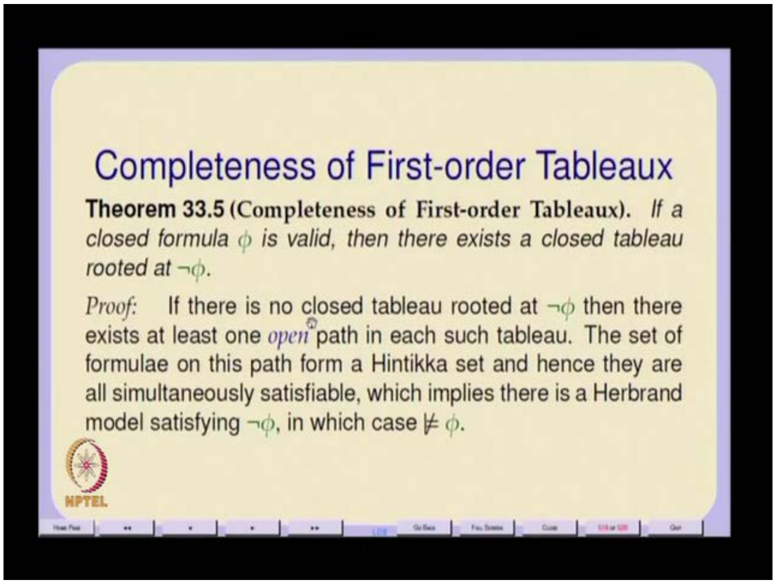
(Refer Slide Time: 27:48)



Because, one thing you have already seen somewhere is that you take you take some formulas with free variables. Let us say and you want to prove that some other formula this is valid is a logical consequence what we showed somewhere was that this is valid. If and only if the universal closures of this formulas are valid therefore, for the purpose of the validity it is sufficient to consider only the closed formula.

So, we will assume that phi is a closed formula and if there is a and suppose there is a closed tableau rooted so a tableau proof of course tries to prove a contradiction. So, which means you take the negation of phi and you take the closed tableau rooted at phi rooted at naught phi. Then, every path in the tableau is closed because of the occurrence of a complimentary pair in the path and that complimentary pair is of ground terms so that is what going to happen. So, on the other hand if phi is not logically valid then naught phi would be satisfiable. Which, means the tableau would not be closed so it would have a open path and so which means if there is a closed tableau then phi is satisfied so this is the way of proving soundness which is vector it looks intuitively obvious which is sort of indirect.

(Refer Slide Time: 29:56)



And then, we have a really trivial way of proving Completeness once you have Hintikka sets if a formula is a we are looking at in a closed formula if they are valid then what we are saying is theirs guaranteed to exists a closed tableau. So, if there is no closed tableau rooted at naught phi let us say phi is suppose to be valid. Then, there exists so you take consider any tableau rooted at naught phi there must be at least one open path in it. And, you consider all the formulae along this open path they form a Hintikka set and we know that always Hintikka sets are satisfiable. Which, means this naught phi is satisfiable which means phi is not valid.

So, Hintikka sets actually give you a complete short give allows you to shorten proofs of these things like completeness especially for tableau methods. Our proof of the completeness of the Hilbert style system may not use Hintikka sets and is actually what we will do is we will go through something very similar to Gerdals original proof. Because, gerdal did not have Hintikka sets when he proved it. So, that is something we will look at later next week so at this point so basically that is all there is a for this lecture. We proved the completeness of resolution, we have proved the completeness of tableau we have to prove the completeness of Hilbert style of system.

So, what then maybe I should do is firstly what now the important thing is that we should not take a fixed signature sigma the sigma is a parameter right of your language. So, the question is can you implement how do you implement of first order tableau where sigma itself is a parameter of your implementation right that is the next assignment. I think I have to put some deadline submission of that assignment. So, you implement first order tableau but do not take any fixed sigma the sigma has to be variable because I should be able to use it as a general purpose engine for any sigma provided by a user.

And, so it is so the crucial thing in that will be the generation of the generation of this the ground terms because, you do not know what this sigma is this sigma is variable right. So, you have to somehow deal with it at the level of an arbitrary string parameter. That comes along with all the kinds of a the along with we will follow standard discipline. In the sense that unlike C language a function symbol can should have a unique arty its arity cannot change from place to place and that arity should be specified as part of. So, there is there has to be a syntax of the string of a string of functions. Which, allows you to take an ordered pair string a, function name and an arity and an integer arity. Which, is consistently used so there has to be a check throughout the program that is it is not being inconsistently used anywhere. So, that is one thing so the generation of an arbitrary term algebra where the signature itself is a parameter of the algebra that is one challenge.

The second challenge is of course is addition to the heuristics which we have specified I mean this heuristic should be these heuristic rules of course will be a part of your engine. But, in addition to that, there is a possibility that if give a formula which is satisfiable. And, then you will go off and do an infinite loop. But, what you need to do is not go off and do an infinite loop and instead the moment use is to take certain judgment on whether I can construct a Herbrand model for that, for that open path for example and that is that is a challenge.

But, in the worst case it may not be always possible and you might actually have to consider. So, as I said you might have to consider several instantiations of a universal quantifier. And, there is no a priory bound on the number of instantiations. So, it might actually go off onto an infinite loop. But, what kind of heuristics can you actually create for example to ensure that some point you decide that. Any further instantiations will actually lead to will actually lead to an infinite path I mean it no use going for that. So, what kinds of so that requires an ordering on the ground terms is actually. And, which by which you can take decision but, in general that there are problems like occurs check and so on which take you off an infinitely. So, this I think we should just make this as next assignment.

So, you extend your tableau of propositional logic with a lot more code now. Which it is going to be a lot more as just the fact you have a term algebra. And, sigma as a parameter user defined parameter is going to make. And, may you a re going to introduce substitutions there code is going to be much more that it was in the propositional case. And, let us see what kind of an engine you can produce.