# Logic for CS

**Prof. Dr. S. Arun Kumar**
**Department of Computer Science**
**Indian Institute of Technology, Delhi**

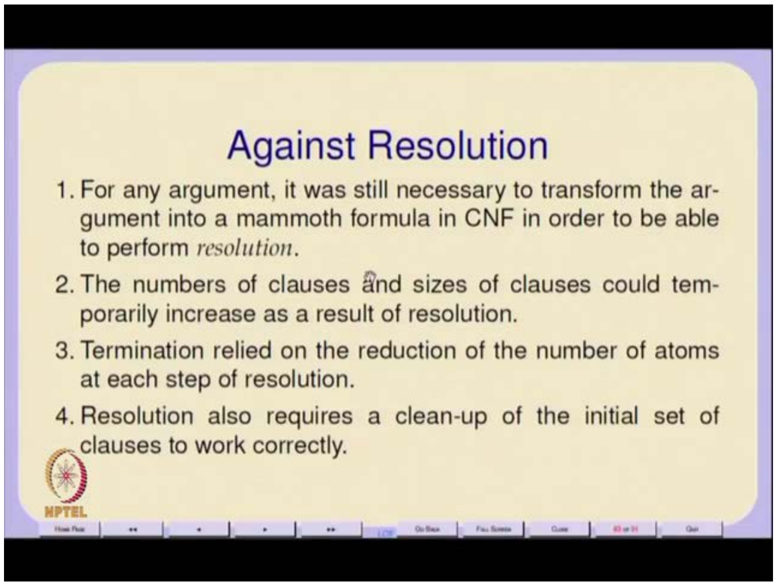
## Lecture - 8
## Analytic Tableaux

Today, we will start something called analytic tableau. And actually the interesting thing about the analytic tableau besides various other technical things is that. So, besides various technical things analytic tableau is interesting. Because it is actually invented by a logician called RAYMOND SMULLYAN. Who is also a magician a puzzlist and he is written some fairly interesting books on solving of puzzles on logical problems on self reference. And RAYMOND SMULLYAN is also a I mean he is written some books on incompleteness on self reference and so on. One of his book is called what is the name of this book.

That is self referential is another one on. Self Reference is also closely linked to self application in the lambda calculus. So, there is one on combinators called to mock or mocking bird I mean you know what a mocking bird is a mocking bird is a actually an australian bird called the kookaburra right. It is called a mocking bird but it sort of mocks itself you know and then it sounds.

So, mock or mocking bird is essentially like is essentially doing a self referential calculations in combinatory logic or the lambda calculus. So, in addition to the RAYMOND SMULLYAN also has some interesting puzzle books. And he has a way of developing logical theories just through puzzles. So you have a sequence of problems in and the solutions to those problems is eventually lead to what is how to what to talk about, might either be about self reference or might be about, (Refer Time: 02:44) incompleteness. or it might be about, a the lambda calculus whatever I mean and So, he is has some fairly a interesting looking interesting ideas as books and book writing is concerned. And also some interesting puzzles that he has invented and he is also magician who actually for many years in the 60s and 70s. Actually he had held a magic shows and with all the getup and so on. And so for the New York so, it an very interesting character. But, besides that he is also a logician and he came up with this analytic tableau method in the 60s. So it will look trivial especially from the top from the point of view of proposition logic. But, it is important to do it because practically all theorem proving methods nowadays use the analytical, use some form of analytical tableau derived from Smullyan tableau.

(Refer Slide Time: 03:43)



## Against Resolution

1. For any argument, it was still necessary to transform the argument into a mammoth formula in CNF in order to be able to perform *resolution*.
2. The numbers of clauses and sizes of clauses could temporarily increase as a result of resolution.
3. Termination relied on the reduction of the number of atoms at each step of resolution.
4. Resolution also requires a clean-up of the initial set of clauses to work correctly.

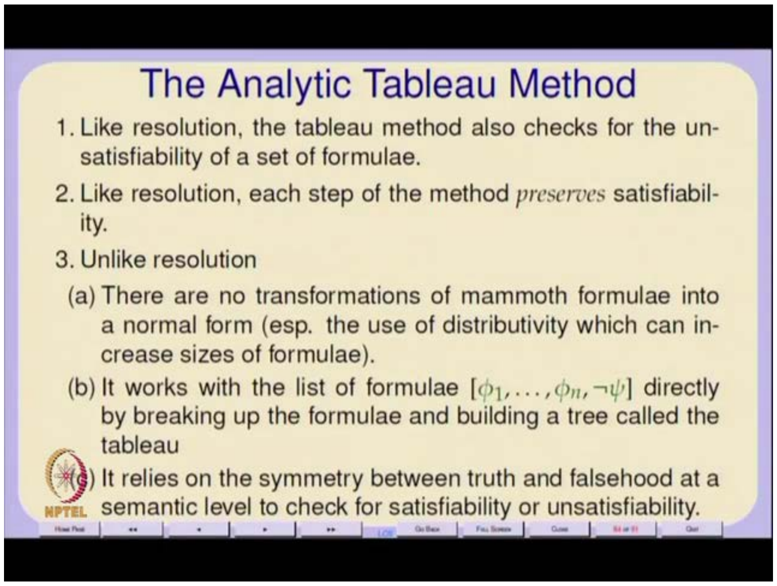
So that is interesting so it is quite fashionable to use tableau's for theorem proving So, as far as resolution is concerned so one of the things about resolution was that you still had this mammoth formula. Which a I mean an argument converted into a mammoth formula each component of I mean you had to convert it into CNF. Which sometimes can involve using distributivity. Which means you can it might expand the formula actually that you going to perform resolution on. And so, this CNF is something that so even if resolution was better than the tautology checker in the sense. That you could do this in a divided fashion it is you still have to do you still have to create a CNF before you can apply resolution.

So, the other thing about resolution is that the numbers of clauses and sizes of clauses could temporarily increase. So, in terms of the amount of space required for storage it can they can actually increase. Of course termination of resolution relied on the reduction of number of atoms at each step of the resolution. And other thing is for resolution to work realy correctly you need to clean-up your initial set of formulas. So, that you do not have debri lying around duplicates and so on and so forth.

Otherwise what can happen is you can be you can get into a false sense of saying. That you have not been able to resolve even though you might be a valid argument. Because of the fact of some of those formulas still lying around which should not have been there.

So, it might prevent your deriving the empty clause.

(Refer Slide Time: 05:49)



## The Analytic Tableau Method

1. Like resolution, the tableau method also checks for the unsatisfiability of a set of formulae.
2. Like resolution, each step of the method *preserves* satisfiability.
3. Unlike resolution
   (a) There are no transformations of mammoth formulae into a normal form (esp. the use of distributivity which can increase sizes of formulae).
   (b) It works with the list of formulae $[\phi_1, \ldots, \phi_n, \neg\psi]$ directly by breaking up the formulae and building a tree called the tableau
   (c) It relies on the symmetry between truth and falsehood at a semantic level to check for satisfiability or unsatisfiability.

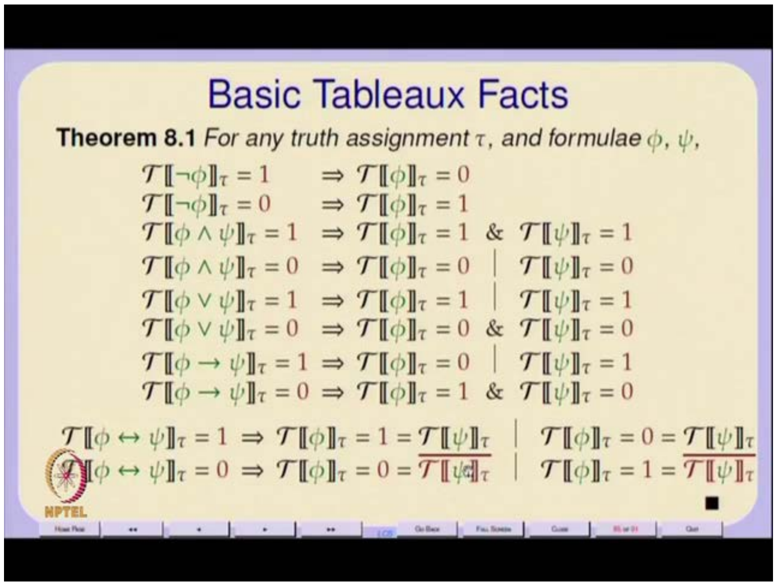
See the tableau method also does exactly like I mean like resolution the tableau method checks for essentially for un-satisfiability. So, the resolution method preserver satisfiability, so you start with an initial set of formulas delta. And you do a resolution you perform a resolution on some atom you get a set of formulas delta prime. And if delta is satisfiable then delta prime is also satisfiable and so and that is how the method of resolution goes on.

What we will do later is we will prove all these facts that I am stating now. But, essentially it preserves satisfiability. And if you derive the empty clause then essentially it means that it is unsatisfiable. So, if delta prime is that is if delta is satisfiable then delta prime is also satisfiable. Therefore if delta prime is unsatisfiable then delta is also unsatisfiable and that is how it works. So, essentially each step of the method the satisfiability is preserved. So, the tableau method also exactly does that I mean it preserves satisfiability. It does not preserve logically that is an important thing. So, unlike resolution one of the things that you save in the tableau method is that, you do not have to deal with transformations or rewrite rules on rewriting. Mammoth formulae into some normal forms so, a necessity of the normal form is not there. And

So, that tableau method works directly on this, formulae. So, you want to prove that psi is a logical consequence of phi one to phi a. Then you just negate psi and do not do any transformation take this list and your tableau method works directly on this list. And what the tableau method does is it actually does a breaking up of compound formulae. So, one thing is it does the breaking up of compound formulae essentially this for each formula that there is only a finite number of times you can up break it up into sub formulae. So, it uses a sub formula property which we will define later. But, which you intuitively understand it is like a sub term and you cannot do this forever. So, you are already guaranteed that for these n plus one formula once you broken them all up the tableau will end. I mean it is some finite tree so, there is no possibility of it going on forever that is one thing so termination is guaranteed in the tableau method.

And the other thing the tableau method does is that it relies on the symmetry between truth and false. So, after your modular truth had just this you could just think of them as just these two values belonging to a complete lattice there is no reason to believe that one is there is no reason to have any symmetry between them. So, the tableau method relies on a certain symmetry between truth and false. So, at a semantic level and it so certain versions of the analytic tableau are also called semantic tableau because of this reason. But, there are some difference between a semantic tableau and an analytical tableau we will come to that much later on the course. So, essentially at a semantic level you rely on the symmetry betweens truth and false. So, to check for satisifiability or unsatisfiability.

(Refer Slide Time: 09:42)



## Basic Tableaux Facts

**Theorem 8.1** *For any truth assignment $\tau$, and formulae $\phi$, $\psi$,*

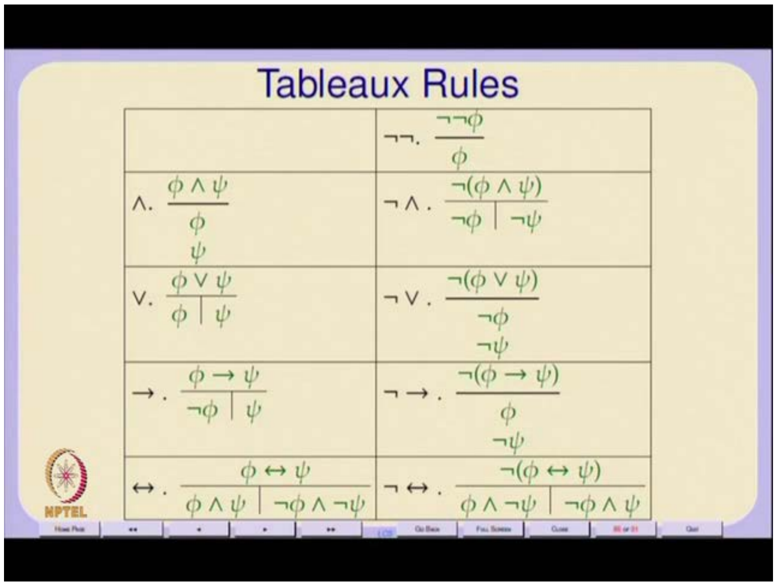
$$\mathcal{T}[\![\neg\phi]\!]_\tau = 1 \Rightarrow \mathcal{T}[\![\phi]\!]_\tau = 0$$
$$\mathcal{T}[\![\neg\phi]\!]_\tau = 0 \Rightarrow \mathcal{T}[\![\phi]\!]_\tau = 1$$
$$\mathcal{T}[\![\phi \wedge \psi]\!]_\tau = 1 \Rightarrow \mathcal{T}[\![\phi]\!]_\tau = 1 \ \& \ \mathcal{T}[\![\psi]\!]_\tau = 1$$
$$\mathcal{T}[\![\phi \wedge \psi]\!]_\tau = 0 \Rightarrow \mathcal{T}[\![\phi]\!]_\tau = 0 \mid \mathcal{T}[\![\psi]\!]_\tau = 0$$
$$\mathcal{T}[\![\phi \vee \psi]\!]_\tau = 1 \Rightarrow \mathcal{T}[\![\phi]\!]_\tau = 1 \mid \mathcal{T}[\![\psi]\!]_\tau = 1$$
$$\mathcal{T}[\![\phi \vee \psi]\!]_\tau = 0 \Rightarrow \mathcal{T}[\![\phi]\!]_\tau = 0 \ \& \ \mathcal{T}[\![\psi]\!]_\tau = 0$$
$$\mathcal{T}[\![\phi \rightarrow \psi]\!]_\tau = 1 \Rightarrow \mathcal{T}[\![\phi]\!]_\tau = 0 \mid \mathcal{T}[\![\psi]\!]_\tau = 1$$
$$\mathcal{T}[\![\phi \rightarrow \psi]\!]_\tau = 0 \Rightarrow \mathcal{T}[\![\phi]\!]_\tau = 1 \ \& \ \mathcal{T}[\![\psi]\!]_\tau = 0$$
$$\mathcal{T}[\![\phi \leftrightarrow \psi]\!]_\tau = 1 \Rightarrow \mathcal{T}[\![\phi]\!]_\tau = 1 = \mathcal{T}[\![\psi]\!]_\tau \mid \mathcal{T}[\![\phi]\!]_\tau = 0 = \mathcal{T}[\![\psi]\!]_\tau$$
$$\mathcal{T}[\![\phi \leftrightarrow \psi]\!]_\tau = 0 \Rightarrow \mathcal{T}[\![\phi]\!]_\tau = 0 = \overline{\mathcal{T}[\![\psi]\!]_\tau} \mid \mathcal{T}[\![\phi]\!]_\tau = 1 = \overline{\mathcal{T}[\![\psi]\!]_\tau}$$

So, what does it do so the basic tableaux facts are for a prepositional logic there are absolutely trivial So, you take this semantic for any truth assignment tau and any formula psi it is these facts are obvious. Basically So, the symmetry between truth and false is that I consider both cases. So, we say so naught phi is true under tau implies that phi is false under tau and that is as simple as that. So, similarly naught phi is falls under tau implies that phi is true under tau 1.

You take term from phi and psi this if this is true then so I am going to use so now, the colors encode language and meta language right. So, black is meta language ok so I use this and this bar for and or in the meta language right. So that is why there are black in color so where as this and this or are the part of the language of proposition logic. So, in each case you consider both possibilities of truth and false under tau. And essentially for a compound formula see how to break it up in terms of the truth and false should of its components. So, essentially the truth table method you sort of read it in this way right. But, this is only the semantics there is of course what happens is the result because of the SMULLYAN of course was only up to these four operators we of course got this fifth operator.

So, and the fifth operator you can see is a little combursive but the bi-conditional is sort of a it breaks the symmetry of the rest of the thing. And actually negation also breaks the symmetry of the rest of the thing. So, these are basic and obvious tableau facts for proposition logic. But, what you actually do remember that any kind of logical deduct or logical reasoning is formal in the sense that it is purely symbolic. And it is not it is not really got to do with its semantics but its only justified by its semantics. So, what you have are what are known as tableau rules

(Refer Slide Time: 12:29)



## Tableaux Rules

| | |
|---|---|
| | $\neg\neg.\ \frac{\neg\neg\phi}{\phi}$ |
| $\wedge.\ \frac{\phi\wedge\psi}{\begin{array}{c}\phi\\ \psi\end{array}}$ | $\neg\wedge.\ \frac{\neg(\phi\wedge\psi)}{\neg\phi \mid \neg\psi}$ |
| $\vee.\ \frac{\phi\vee\psi}{\phi \mid \psi}$ | $\neg\vee.\ \frac{\neg(\phi\vee\psi)}{\begin{array}{c}\neg\phi\\ \neg\psi\end{array}}$ |
| $\rightarrow.\ \frac{\phi\rightarrow\psi}{\neg\phi \mid \psi}$ | $\neg\rightarrow.\ \frac{\neg(\phi\rightarrow\psi)}{\begin{array}{c}\phi\\ \neg\psi\end{array}}$ |
| $\leftrightarrow.\ \frac{\phi\leftrightarrow\psi}{\phi\wedge\psi \mid \neg\phi\wedge\neg\psi}$ | $\neg\leftrightarrow.\ \frac{\neg(\phi\leftrightarrow\psi)}{\phi\wedge\neg\psi \mid \neg\phi\wedge\psi}$ |

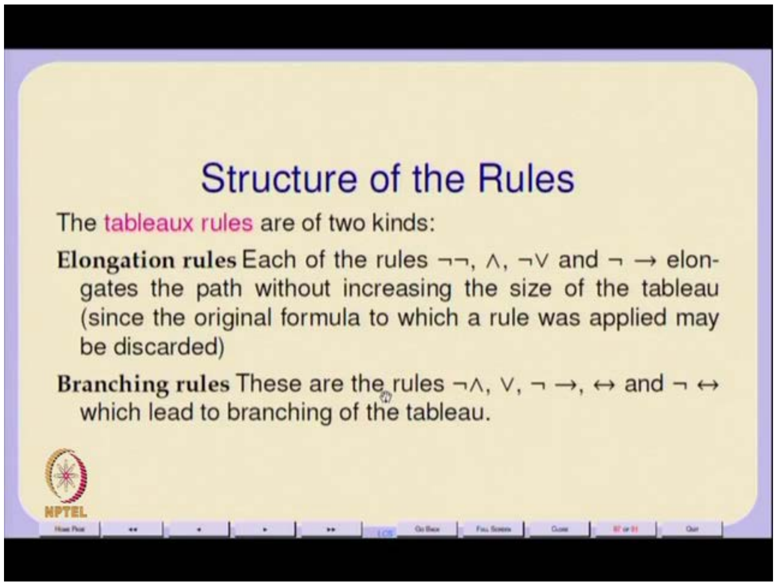
So, initially of course SMULLYAN  use this essentially to mark each formula. So, as true or false you know and mark the conditions. But, all occurrences of false can be replaced by negations and actually you can come up with this set of rules.

So, here is a set of nine rules which these are just these are also forms of rewrite rules these rules essentially preserves satisfiability in what way. So in case of double negation it is obvious. So, in the case of this so if I have a formula of the form phi. And psi then phi must be and if this is satisfiable then phi must be satisfiable and psi must be satisfiable. So, there is an here when I write psi below phi I am actually creating a path of conjunctions. So, there is an implicit and between them, So, if you look at naught. So, now because of the symmetry between both truth and false should and false should essentially works out to negation. So, you have rules for each of these operators and their negations.

So, the rule of not and essentially says that this formula naught of phi and psi is satisfiable. Provided naught phi is satisfiable or naught phi is satisfiable. So, this bar is the or the meta or now what we have. So, similarly so  or here, translates this to this meta or and similarly in this case this naught of or actually translates to an and naught phi and naught psi the case of implication phi or psi is satisfiable if either naught phi is satisfiable.

So, there is a so these rules so, we can classify all these rules before that lets look at this last bi-conditional the bi-conditional both the bi-conditional. And it is negation are branching rules the negation of the bi-conditional is just the exclusive or right. So, both of them are branching.

(Refer Slide Time: 15:33)



## Structure of the Rules

The tableaux rules are of two kinds:

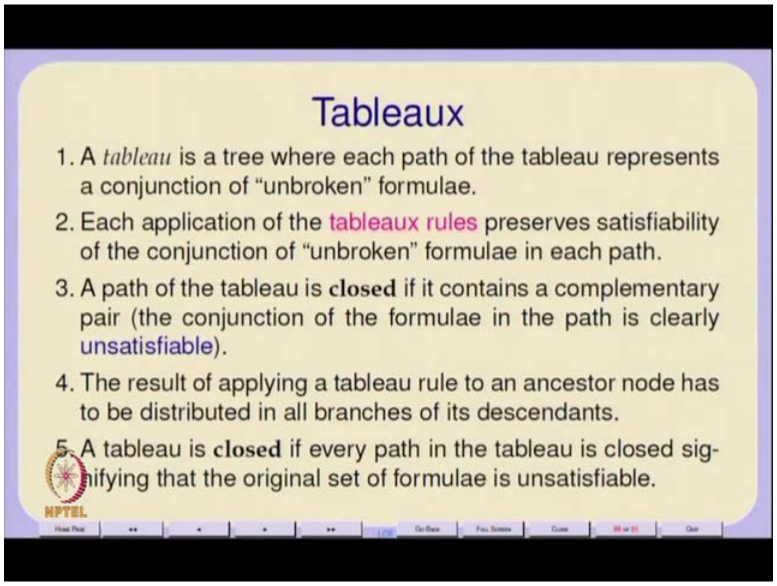
**Elongation rules** Each of the rules $\neg\neg$, $\wedge$, $\neg\vee$ and $\neg\rightarrow$ elongates the path without increasing the size of the tableau (since the original formula to which a rule was applied may be discarded)

**Branching rules** These are the rules $\neg\wedge$, $\vee$, $\neg\rightarrow$, $\leftrightarrow$ and $\neg\leftrightarrow$ which lead to branching of the tableau.

So, essentially the rules the structure of the rules is such that. You have elongation rules like there are the and rules the and rules also includes naught of or and you have branching rules. So, each of these so then double negation also can be regarded as an elongation rule because value it does not branch so that is. So, basically what actually the bi-conditional rule actually expands the size of formulas we considered. But, all other rules do not expand the size. So, if you think of in all the rules you think of the top as a numerator and bottom as the denominator. Then essentially the sizes of rules are non increasing I mean the sizes of denominators are non increasing except in the case of this bi-condition.

So, which is good reason, Why smullyan did not consider it because, it had sort of destroyed the beauty of tableau. But, we will since we are living with it we will going to live continue to live with it. So, what we are going to do so what you are essentially going to do is you apply a rule. So, you think of it as plain blank symbol pushing and you apply a rule.

(Refer Slide Time: 17:17)



## Tableaux

1. A *tableau* is a tree where each path of the tableau represents a conjunction of "unbroken" formulae.
2. Each application of the tableaux rules preserves satisfiability of the conjunction of "unbroken" formulae in each path.
3. A path of the tableau is **closed** if it contains a complementary pair (the conjunction of the formulae in the path is clearly unsatisfiable).
4. The result of applying a tableau rule to an ancestor node has to be distributed in all branches of its descendants.
5. A tableau is **closed** if every path in the tableau is closed signifying that the original set of formulae is unsatisfiable.

So, what you get is elongation and branching along. And maybe for some elongation some branching so on so.  So, what you get is a tree so a tableau is a tree I mean actually I myself did not know it, was till I learnt of smullyan tableau. So, a tableau is a tree and you take each path in this tree of what might be called unbroken formulae.

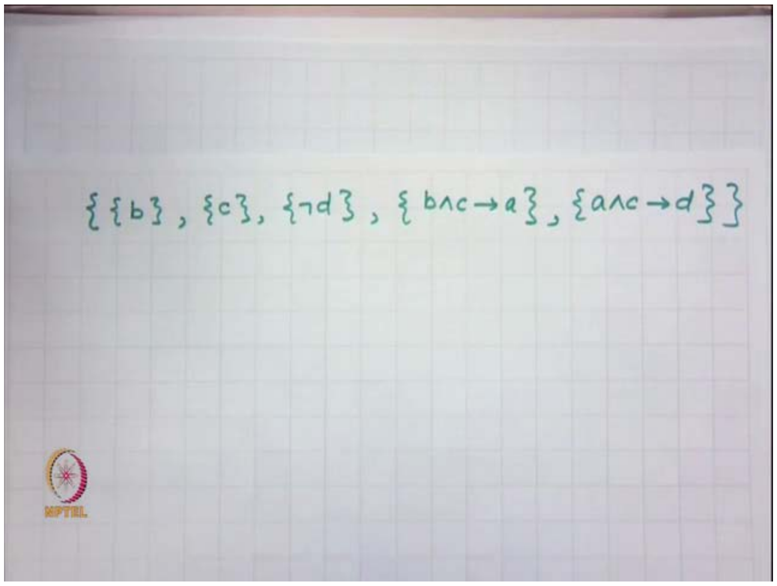
So, there is another concept also so there is an implicit concept of a formula being broken up which is what the original thing was doing. And when you breakup a formula you can essentially discard the original formula because you can you just need to keep the denominator and you do discard the numerator is flexing your muscles. So, the point is that see you have you are going to break up the formulas in some way. And when you break up one formula by applying a rule. Then you can discard the original. So, a part so since it is a tree I can consider the various parts of the tree. So, there is this the effect of applying formula is to consume an old formula in the creation of new formulae.

So, a path of the tableau is closed if it contains a complimentary pair. Essentially each path in the tableau represents a conjunction of formulae of unbroken formulae. Because I have seen of that said of I mean the original formula is satisfiable provided some path is satisfied. So, some conjunction of some unbroken formulae is satisfied. If it is but, if it contains the path contains a complimentary pair then that path gets closed of course that path might be a branch of something. So, only the path gets closed the branch does not get closed. Because they might be all so the ancestors which are common to several branches remain they still remain part of the unbroken set of formulae different path.

So, you look at each path and you close the path if it contains a complimentary pair and essentially what you saying is if I take all the unbroken formulae in this path. And take their conjunction then that is unsatisfied so, contradiction. So, in the case of branching what happens is supposing I take some ancestor node. And I have already branched that ancestor is a common ancestor of two branches and if I am going to breakup that ancestor then I have to distribute the denominator on or I have to replicate the denominator on all the branches. So, this the result of applying a tableau rule to an ancestral node has to be distributed in all branches of its descendents.

A tableau is closed if every path in the tableau is closed. Essentially signifying that paths are unsatisfiable all possible conjunctions that you taught of are unsatisfiable. So, this is what happens so let us just take a the example we did the time and let us look at it so, what I had yesterday was.

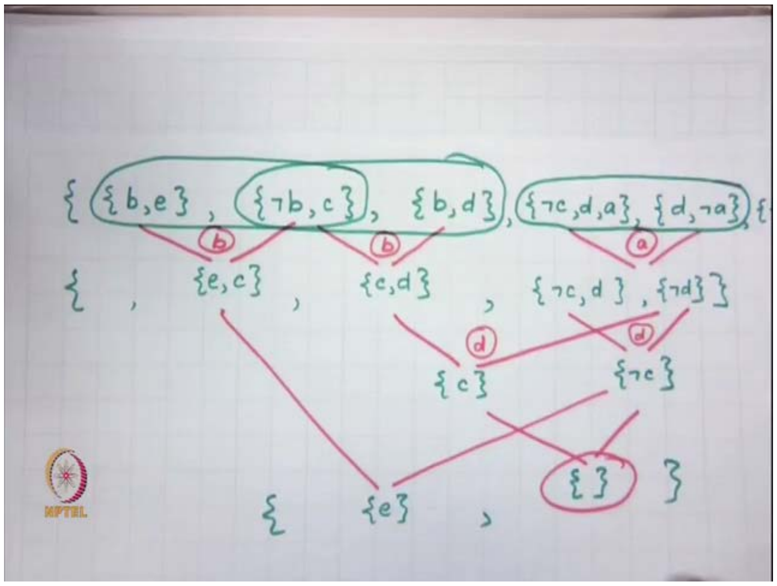
(Refer Slide Time: 21:04)



$\{\{b\}, \{c\}, \{\neg d\}, \{b \wedge c \to a\}, \{a \wedge c \to d\}\}$

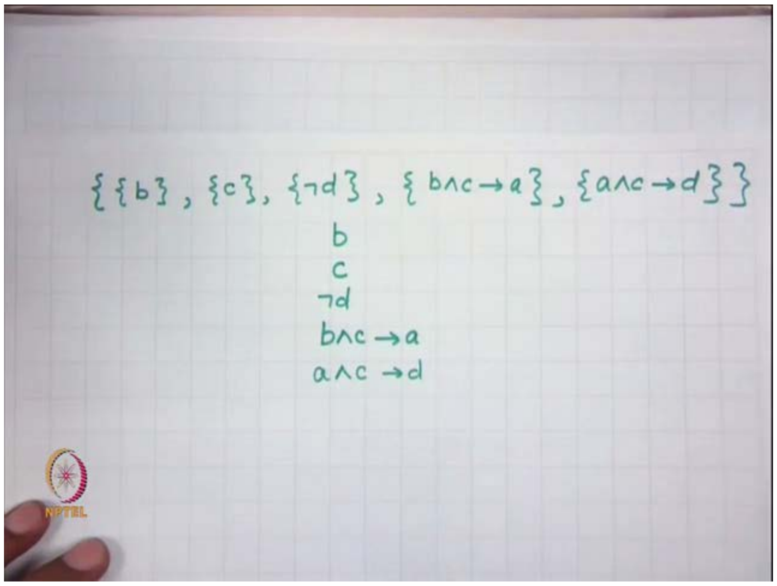
So, I had this so basically I had this lets say this b c naught d b and c arrow a, a and c arrow d. So, what I did yesterday was I actually took the conjunctive normal form of this result this set. And I did the resolution.

(Refer Slide Time: 21:57)


{ {b,e} , {¬b,c} , {b,d} , {¬c,d,a} , {d,¬a} , {
b
b
a
{ , {e,c} , {c,d} , {¬c,d} , {¬d} }
d
d
{c}
{¬c}
{ {e} , { } }

Which so, last time what we had was this right. Which is essentially the same thing how we did this resolution last time. So, now we take this essentially for this so it is a different one it is not the same one. So, what we will do is as of this what we are going to do this is we are going to create a tableau.

(Refer Slide Time: 22:25)



$\{\{b\}, \{c\}, \{\neg d\}, \{b \wedge c \rightarrow a\}, \{a \wedge c \rightarrow d\}\}$

$b$

$c$

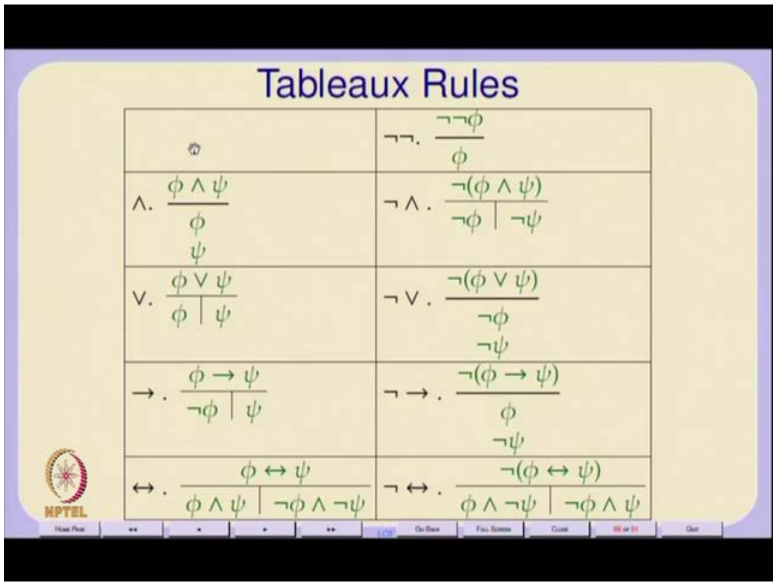
$\neg d$

$b \wedge c \rightarrow a$

$a \wedge c \rightarrow d$

So, in the creation of the tableau you can think of it this way. I start with let us say a b c naught d b and c arrow a, a and c arrow d. So, basically it us a conjunction of all these and based and what I want to know is I want to check whether it is inconsistence whether it is consistent whether it is satisfiable or not.

(Refer Slide Time: 23:00)

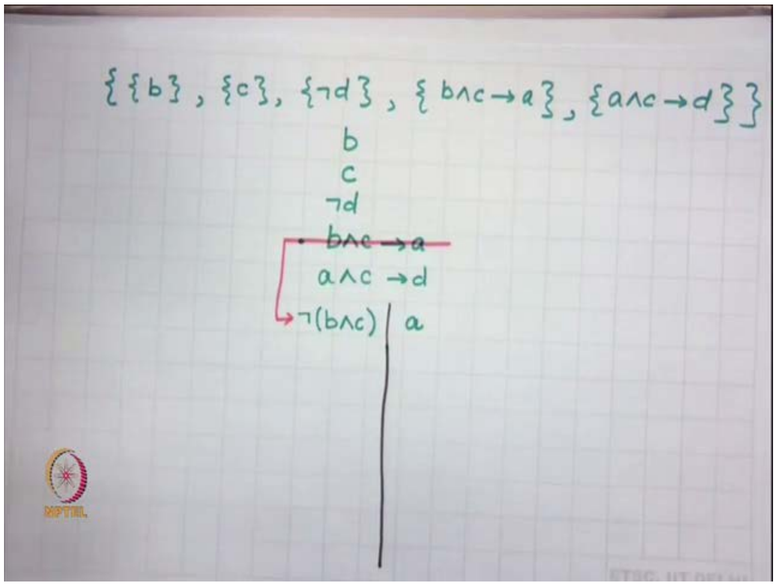


## Tableaux Rules

| | |
|---|---|
| | $\neg\neg.\ \dfrac{\neg\neg\phi}{\phi}$ |
| $\wedge.\ \dfrac{\phi \wedge \psi}{\phi \quad \psi}$ | $\neg\wedge.\ \dfrac{\neg(\phi \wedge \psi)}{\neg\phi \mid \neg\psi}$ |
| $\vee.\ \dfrac{\phi \vee \psi}{\phi \mid \psi}$ | $\neg\vee.\ \dfrac{\neg(\phi \vee \psi)}{\neg\phi \quad \neg\psi}$ |
| $\rightarrow.\ \dfrac{\phi \rightarrow \psi}{\neg\phi \mid \psi}$ | $\neg\rightarrow.\ \dfrac{\neg(\phi \rightarrow \psi)}{\phi \quad \neg\psi}$ |
| $\leftrightarrow.\ \dfrac{\phi \leftrightarrow \psi}{\phi \wedge \psi \mid \neg\phi \wedge \neg\psi}$ | $\neg\leftrightarrow.\ \dfrac{\neg(\phi \leftrightarrow \psi)}{\phi \wedge \neg\psi \mid \neg\phi \wedge \psi}$ |

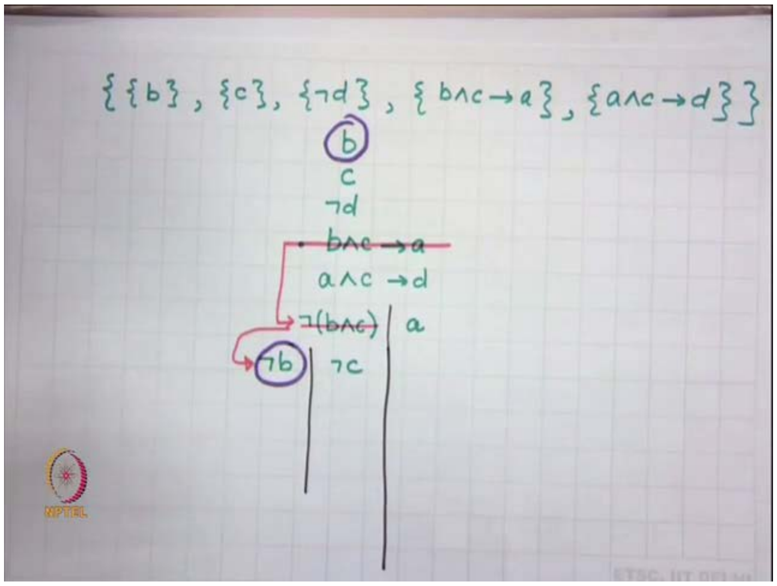
So, we apply tableau rules let us so the top three of course there are literals in their basic forms there is no breaking up ever possible. So, if you apply a tableau rule you have to apply to one or the other rule.

(Refer Slide Time: 23:28)



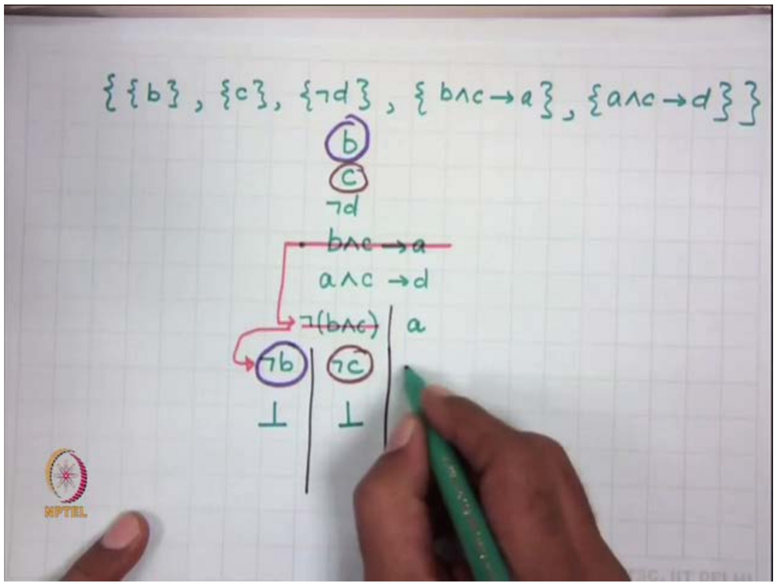
So, this rule so if I apply the tableau rule to this one then I essentially get naught b and c. And I have this bar all right and I immediately essentially have a branch. Now, actually throughout the proof this branch is like the Berlin wall I mean you cannot cross it. And what you have is you have a in this, on this side right but, you got this by applying the arrow rule on this formulas. So, this formula has been consumed.

(Refer Slide Time: 25:05)



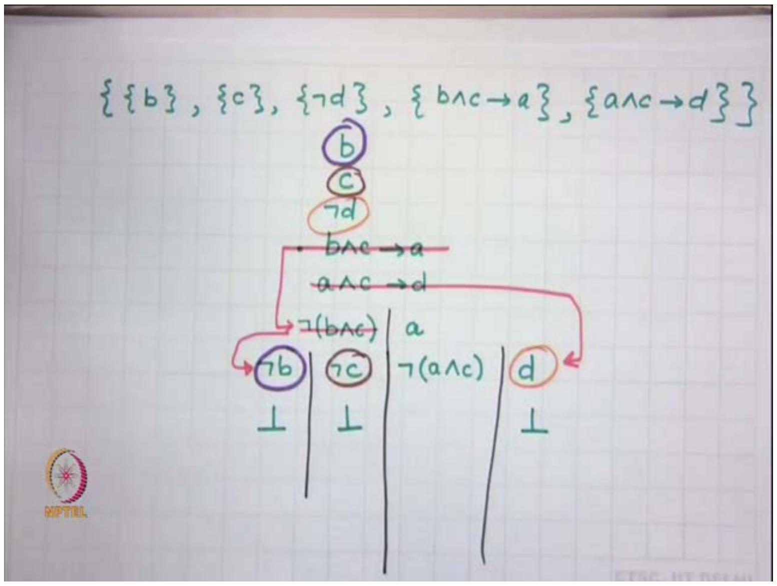
So, just think of it as this and now my tableau is essentially this formula b and c arrow a can be discarded. And now, my tableau consists of all these other rules right. So, now what happens is I could actually do a I could have this is a compound rule I could actually apply the and naught of and to the this rule to this formula. And what I get is I get naught b naught c. And especially I have another branch and I got this from this gets discarded. So, now I have only these of course but now, what has happened is that this path has a complimentary pair. And so essentially the path is closed and I signify the pressure of the path by using putting a bottom that is it.

(Refer Slide Time: 26:07)



So, now and actually then what happens is this path is also closed right and so these two paths are closed and that leaves only this open path. And therefore it allows me to breakup this formula up and when I break this formula up I get a.

(Refer Slide Time: 26:27)

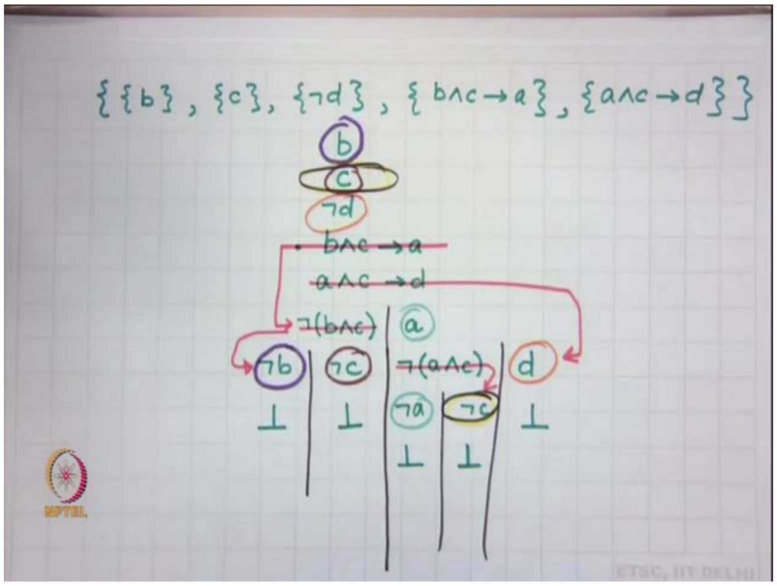


So, now I have a tableau with just one path on in it but now the breaking up of this formula will give me naught of a and c and then or of d. And I have a fresh branching here so this d so fine.

So, this d is has also gone and therefore I have I can close this path and that leaves only this I should also mention that this is being consumed right. So, this can be discarded so what happens to this well I get notice this the circled formulae are not consumed they are still unbroken.

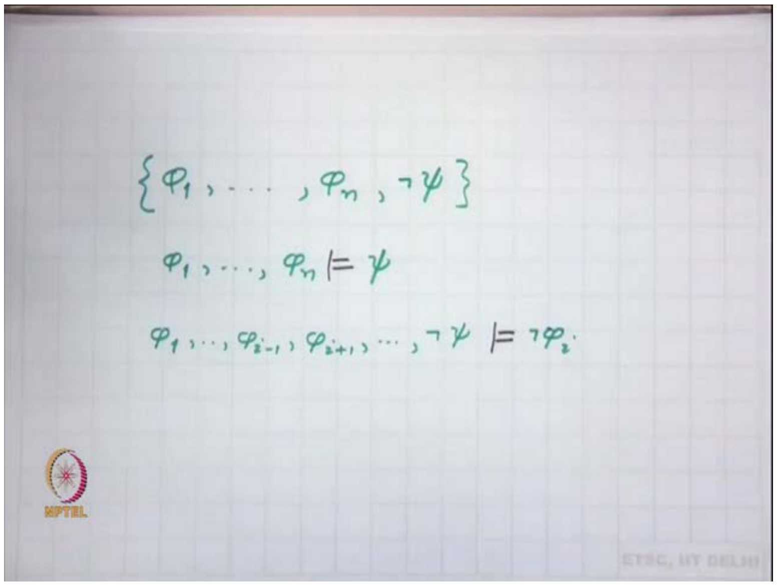
Student: For instance if (Refer Time: 27:30) or any other path then why (Refer Time: 27:34) a a and c would be a boken up along the path. Yes you would have to be distributed along all the paths. Otherwise you are have always you would have got into trouble.

(Refer Slide Time: 27:59)



So, this one would be naught of and so this one would give me naught a naught c and of course it is a branching. Here and now this one and firstly this one has got consumed and secondly this one has got closed. And then this one also has got closed because of lack of suitable colors. This is got closed along with this c so, this gets closed. Therefore essentially what we are saying is that all paths are closed so this tableau is closed and this tableau is closed essentially means. So, therefore that you remember that this is something in the case of both resolution and tableau what have you actually done. You have taken a set of formulae and proved that there is inconsistent let us say.

(Refer Slide Time: 29:29)



$$\{\varphi_1, \ldots, \varphi_n, \neg\psi\}$$

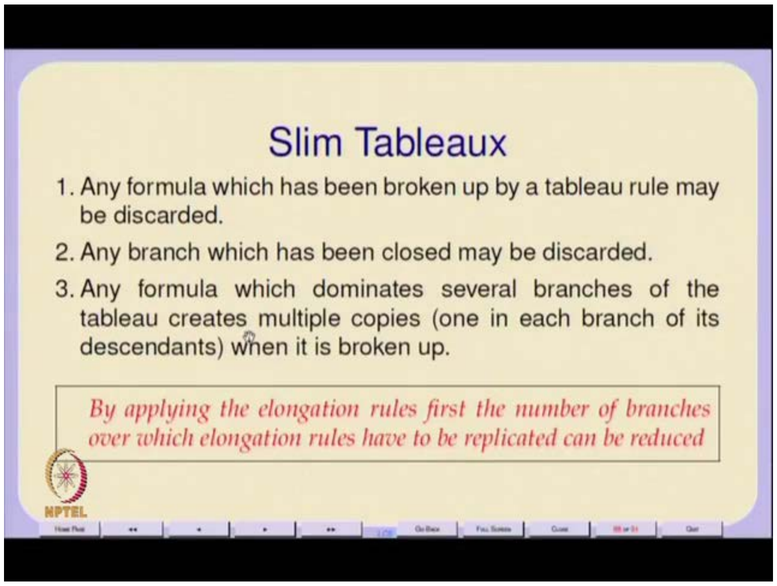
$$\varphi_1, \ldots, \varphi_n \models \psi$$

$$\varphi_1, \ldots, \varphi_{i-1}, \varphi_{i+1}, \ldots, \neg\psi \models \neg\varphi_i$$

So, this set of formulae was let us say phi one to phi n and a conclusion psi right. And when you look at it as a set of formulae there is absolutely no reason why you should rely regarded this way I mean there is absolutely no reason why you should think that this has to be only this. It could just as well I could choose some phi i for example and I could claim  that phi i minus 1 phi i plus 1 naught psi naught phi i right. I mean both these arguments would have yielded the same set of formulae right. You are only proving a set of formulae is inconsistent and there could be many ways of setting up the argument. So, this could just as well have been another argument.

Supposing in fact what you can show from the semantics is that if this is this is a valid argument if and only if this is valid argument. At the levels of sets I can pervert things in anyway. But I can also pull them out on either side of the what is this the models symbol anyway I like right. So, it does not matter whether you are looking at this argument or this argument and for any argument phi in fact. So, they all I can move things on either side of this doubleton style. And all those arguments would give me the same set of formulae. And if this set of formulae is unsatisfiable it means all those arguments are valid. So, you take any argument right I can negate the conclusion and make that a hypothesis take one of the hypothesis negate it and take it as a conclusion right. And either case the proofs are going to be the same. And therefore the arguments are equally valid or equally invalid. I mean so, that is why this is that is one thing that this symmetry gives us.

(Refer Slide Time: 32:41)



## Slim Tableaux

1. Any formula which has been broken up by a tableau rule may be discarded.
2. Any branch which has been closed may be discarded.
3. Any formula which dominates several branches of the tableau creates multiple copies (one in each branch of its descendants) when it is broken up.

*By applying the elongation rules first the number of branches over which elongation rules have to be replicated can be reduced*

The other thing of course is as he pointed out the other thing is of course you can create really slim tableau by ensuring that you use only elongation rules as far as possible. So, you start with this you can with essentially a path of you the starting point of your tableau is a path of formulae. So, apply only those choose only those formulae which elongate and postpone the formulae which branch to as late as possible. And that is a heuristic which will reduce the size of the tree that you created. So, as he said I mean if you had if you had apply the branching rule at the first it said then you are already branched. And then you apply an elongation rule to one of the ancestors of that branching point. Then you have to distribute the denominators into each of the branches and that increase the size of the tree right. And therefore the size of the tableau that you have created construction that you have constructed. And if you follow this heuristic of first applying only elongation rules to the tableau to the formulae in the tableau. And only after you have exhausted all possibilities of elongation do you apply the branching rules and you can have essentially a fairly slim tableau.

And of course it does not matter in either way you are anyway bounded by the size of the individual formulas. So, you your tableau anyways has gone to have only a certain fixed size that in terms of in terms of the lengths of the paths. What your what this heuristic is essentially saying is I can do a factoring of the paths of the common elements of the paths by delaying the use of the branching rules and not starting. And applying only elongation rules wherever possible.

So, of course so what happens is so the thing is the tableau so by this is actually the only heuristic you require. The other interesting thing about the tableau is that it is syntactic it is based on a form on something like structural induction. So, if you look at if you look at so it is not just it is like somewhat the SOS rules that you must have done in programming languages they were all based on structural induction on the on the operator. I mean you so this is also if you look upon naught and naught or and so on as single operators which is fair enough. Because if you are looking at this symmetry of 0 and 1 then naught and is also a binary operator naught and is also a binary operators. So, you are essentially chosen these operators. And what you have done is based on the root operator in the abstract syntax tree you are applying a rule and that is deterministic is unique.

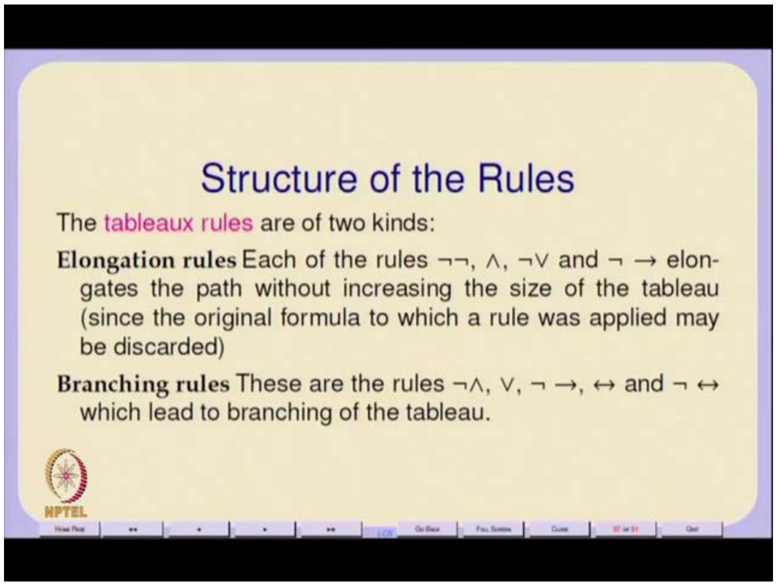
So, there is a unique deterministic way of actually applying these rules the only way the determinism gets destroyed. Is because you are starting with some you could you could be permuting you could permuting this starting list of formulas in any way. That is the only place where there is no determination. And the only other place where there is some non-determinism is actually the choice of formula you take for applying a rule. But, once you have chosen a formula the results a completely determinist.

These are important these are important consideration why they affect the complexity of both the space and complexity of the tableau method. The other thing you see is that this is what makes the tableau rules available to deterministic proof theoretic procedures take the notion of the proof. In any typical situation in any branch of mathematics given a certain given a certain theorem statement I have. If I have to prove it there are (refer Time: 37:40) number of possibilities of how you can go I mean in many proves there is this notion is if you do not get this crucial step when you are stuck. You are never going to get the proof but that is because, of the extreme amount of non-determinism that is available in the proof procedures in a normal reasoning.If you can restrict the reasoning to certain deterministic procedures. And here the deterministic procedures are entirely defined by the operators at the root. Then there is a much better possibility of actually running in proof procedures. So we will do formal proof theory from a logical view point and that time you will actually appreciate.

Sometimes what happens is one of the reasons is you keep circling without getting the main results is because of the fact there is a bewildering variety of possibilities. Iif you do not go in a syntactic way structured fashion. Whereas if you have unique rules we will see another proof system which has unique rules and from which actually the tableau was inspired. If you have unique rules and the number of degree of determinism the degree of non-determinism greatly reduces. And you have deterministic and gold directed proof procedures. Which therefore the moment the amount of branching reduces your complexity also reduces remember that.

If the number of possibilities is large initially then, your both your space and time complexity get multiplied by that factor of by the branching factor. Whereas if you have deterministic procedures then, your space and time complexity get narrowed down only to that determinism. So, here that degree of non-determinism is just the in the possibility of what which formula are you going to chose for the purpose of elongation for example.

(Refer Slide Time: 32:41)



## Structure of the Rules

The tableaux rules are of two kinds:

**Elongation rules** Each of the rules ¬¬, ∧, ¬∨ and ¬ → elongates the path without increasing the size of the tableau (since the original formula to which a rule was applied may be discarded)

**Branching rules** These are the rules ¬∧, ∨, ¬ →, ↔ and ¬ ↔ which lead to branching of the tableau.

So, there are certain other formal things that we need to do which now, that we have done two or three methods of proof from a purely implementation point of view. We have to formalize them and we have to do them also. We have to prove what is known as soundness and completeness of these rules. That will set the setting of it will complete the setting of all the concepts that we require before we get on to the first tautology. So we have to so I will do that next week the soundness and completeness of both resolution and tableau rules I will not do it for the tautology checker. But for these two at least I will and then we will come up with a notion of a calculus and the soundness and the completeness of the calculus. And how what are the kinds of proof methods that we require in order to prove some soundness and completeness.