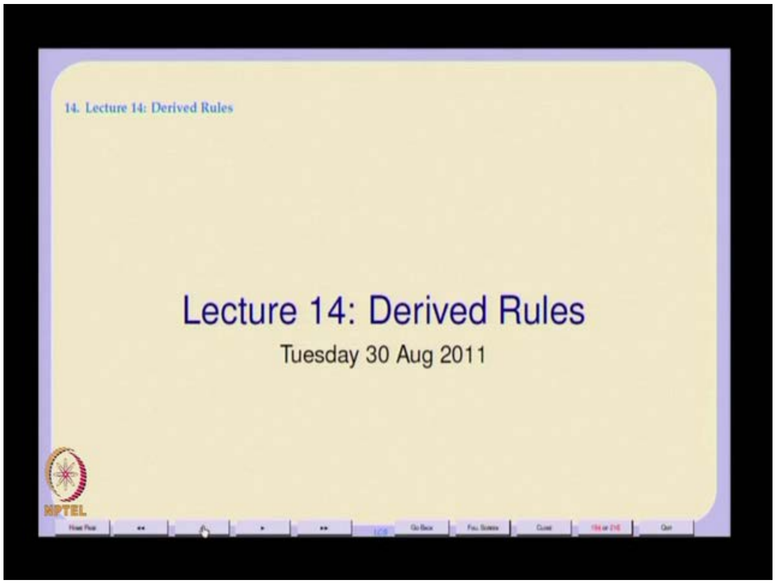
Logic for CS

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

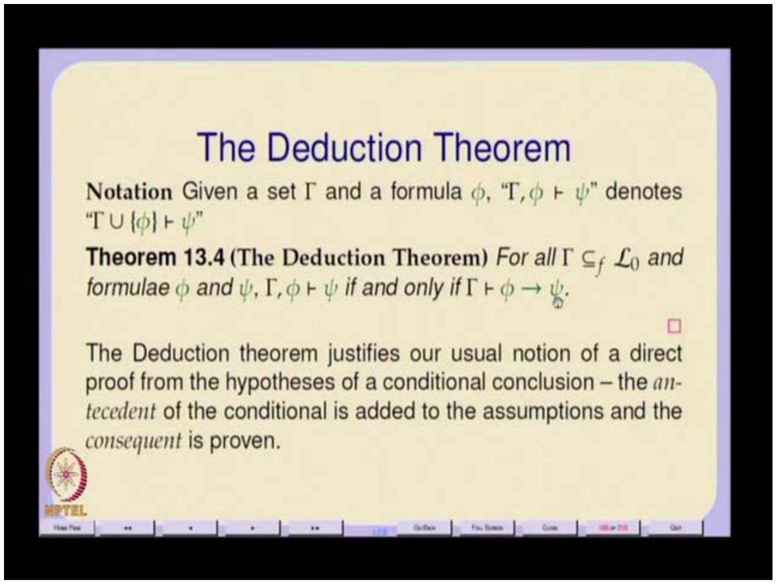
Logic of Computer Science
Lecture - 14
Derived Rules

(Refer Slide Time: 00:24)



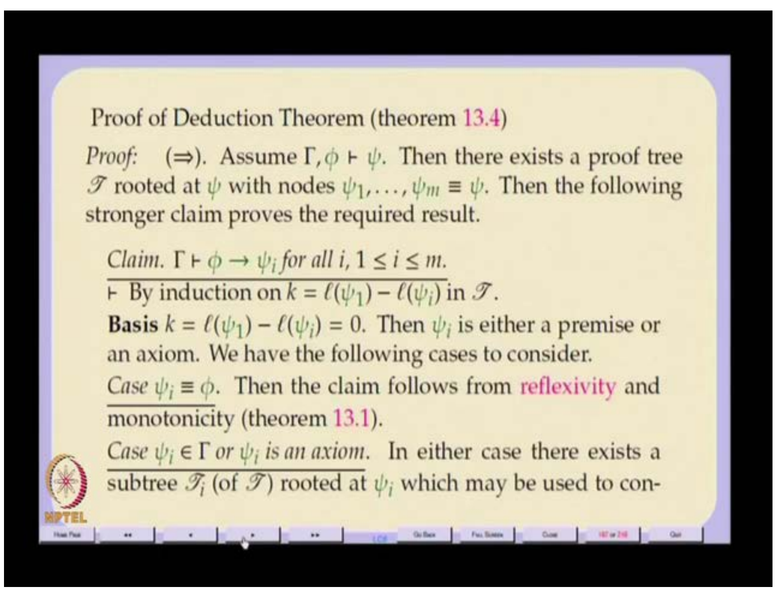
So, we were looking at Hilbert type proof system. And we in fact saw some proof and one important theorem that we saw was the deduction theorem. Which essentially allows you to move some formulae from the hand side of the provability symbol to the left hand side as an assumption. And prove what is essentially so in a formula of the form phi arrow psi phi is called the antecedent and psi is called the consequent. So, it allows you to move the antecedent as an assumption and prove only the consequent.

(Refer Slide Time: 01:11)

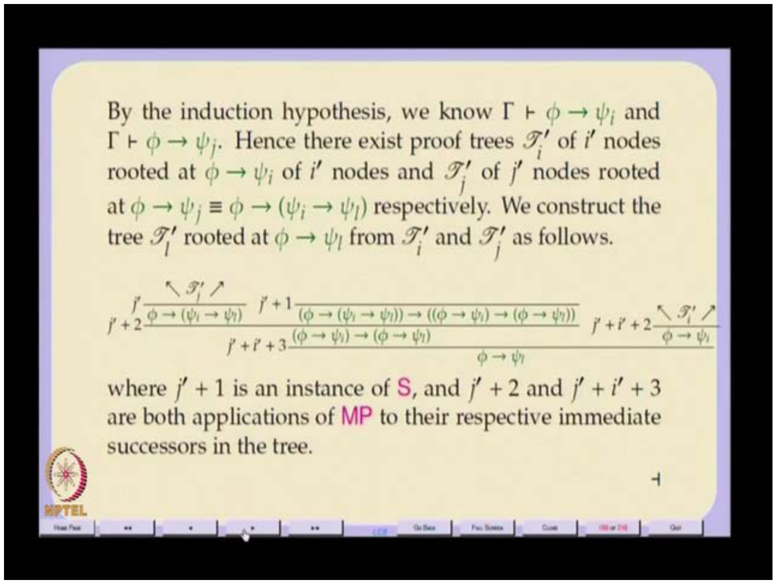


So, and of course this deduction theorem the way that we have proven it is in a if and is a characterization. So, it means that you can also move assumptions onto your prove onto the provability side and therefore you have two way characterization.

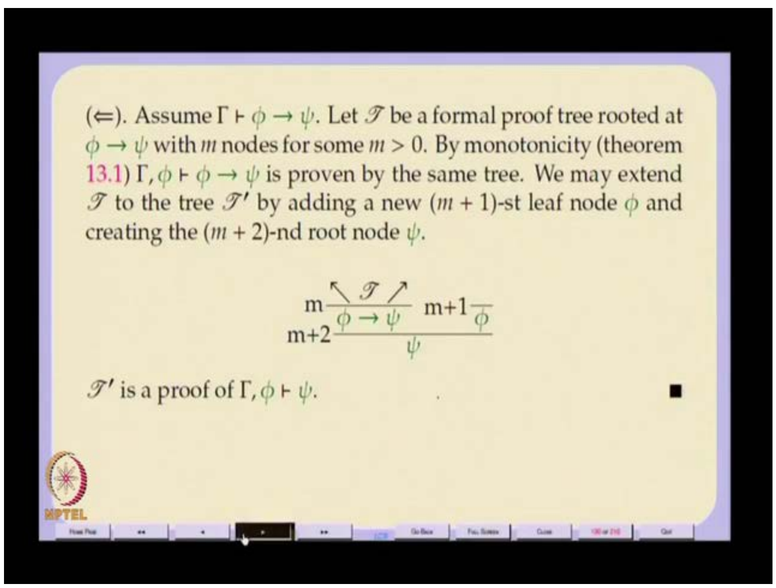
(Refer Slide Time: 01:39)



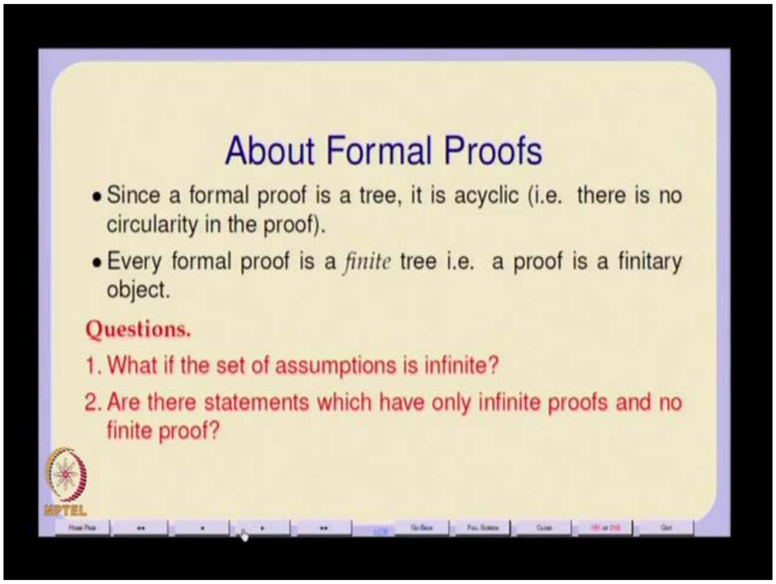
(Refer Slide Time: 01:42)



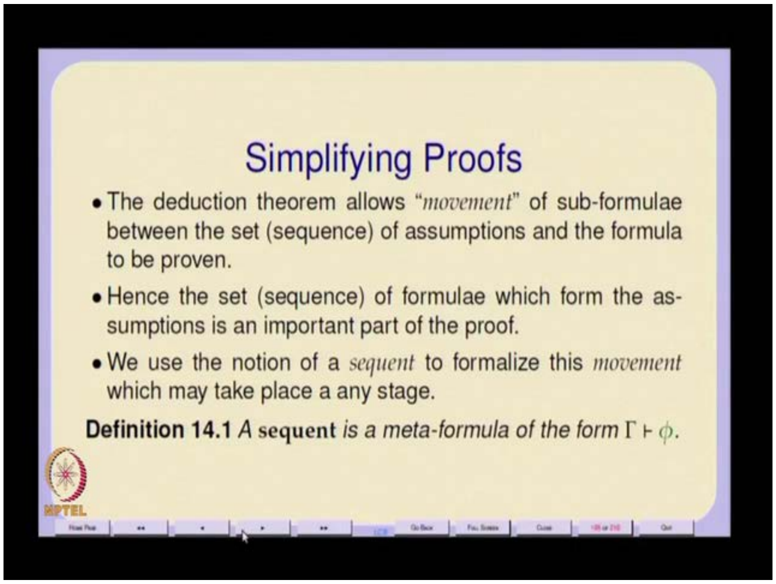
(Refer Slide Time: 01:43)



(Refer Slide Time: 01:44)



(Refer Slide Time: 01:44)



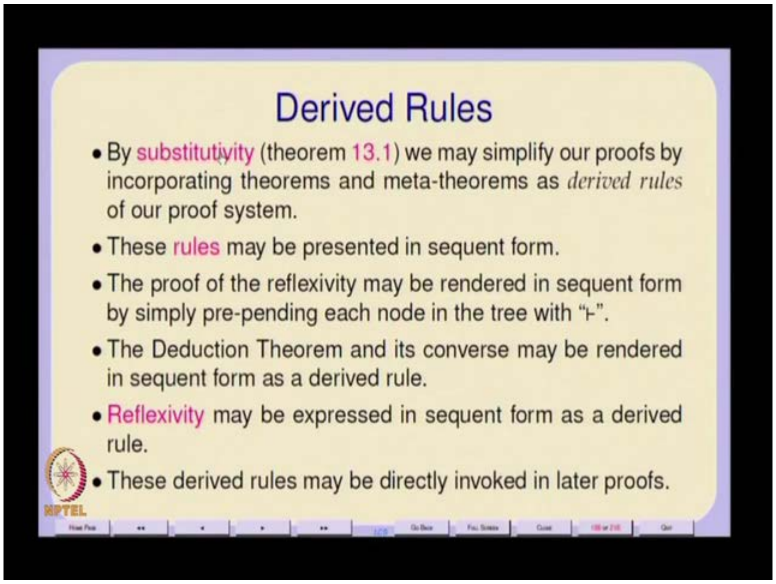
And then, we did this proof we just fairly complicated at least for the forward kind of rule. So, what happens is this moment that the deduction theorem allows you to do allows to greatly simplify the proofs. Because you are allowed to make essentially you can make more some extra assumptions and you have to prove shorter formulae that is one thing.

So, which means that essential so what you take as assumption is actually an important part of the proof. So, far the way we have looked at are proofs we just the assumptions were somewhere in the background. And we sort of understood what we were taking as an assumption. And we and we actually applied the rules as we have given. So, one possible thing to do is actually because of this movement that is possible due to the deduction theorem one thing is that one all have to make the assumptions explicit at each stage. Because, essentially this movement implies that at every stage in a proof one might have a different set of assumptions. And they have to be somehow consistent and they should all form of.

So, we will use a notion of a sequence a sequent is also a meta language a meta-linguistic formulation. And essentially this sequent essentially will allow us to clearly specify at each stage, What are the assumptions? And what is on the right side of the provability?

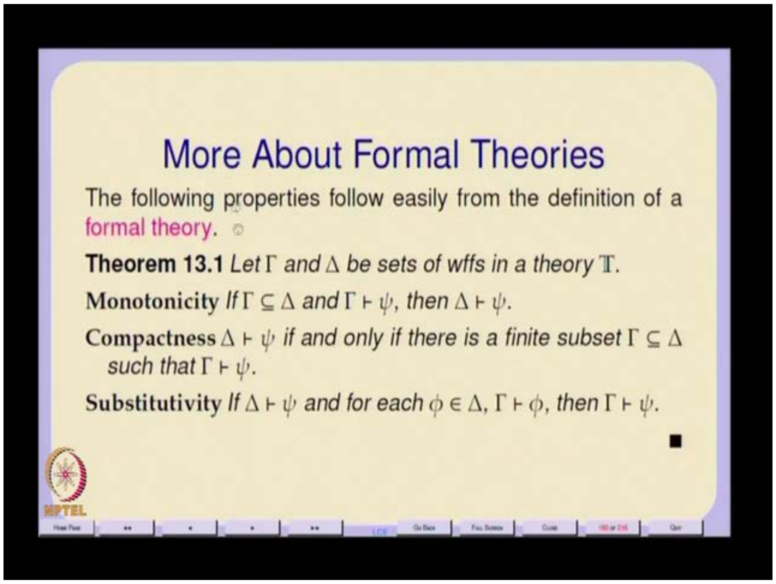
So, what is on the left side of the provability symbol? And what is on the right side of the provability symbol? So, a sequent is a meta formula essentially of this kind. So we will put we will incorporate the tensile and the assumptions directly into our notion of proof.

(Refer Slide Time: 03:47)



So, one of the things that the substitutivity allows you to do is to split up theorems into Lemmas and sub Lemma propositions maybe.

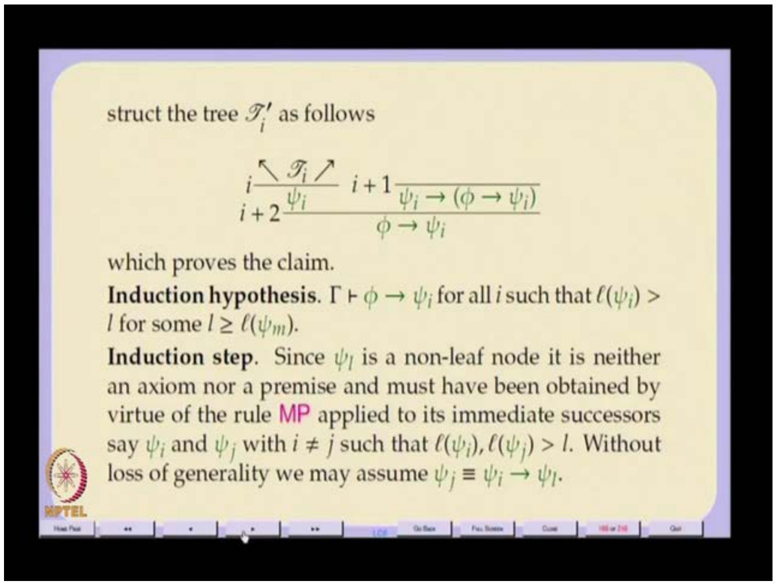
(Refer Slide Time: 03:48)



And also it allows us to incorporate those Lemma that we have proven actually as to be directly incorporated into the theorems. So, formally speaking if you incorporate a Le Lemma essentially what you are saying is I take the entire proof tree of that Le Lemma. And graft it, onto the current proof a simpler thing would be to just invoke that Lemma convert that Lemma into a derived what is known as a derived rule. And use that derived rule instead so essentially what we are saying therefore is that will have a in addition to the basic rules will have a growing number of derived rules. Rules that have been derived there like the Lemmas that we have already proven. And usually in our proof instead of taking the entire proof of the entire Lemma or reproving that Lemma of the current stock of symbols that have been used in this proofs. It is a good idea to take that Lemma convert it into a derived proof rule. And add this proof rule to the existing body of knowledge essentially of that theory So that is

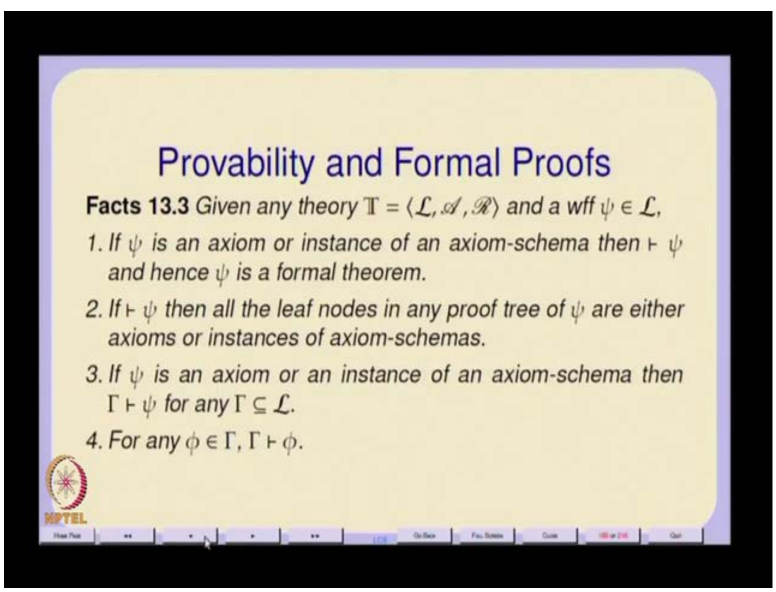
So, those are how we will proceed so, we can have derived rules. And what we can do is of course in all in the case of all these Lemmas. And theorems that we are going to use you they have there, own set of assumptions. So, a sequent form for these derived rules is actually the most suitable thing. So, then you can clearly understand whether that Lemma is actually applicable in the current scenario in a proof that you are doing or not. That is that will become clear from the assumptions. So, we could actually take this, Lemmas and this meta theorems and somehow reform a somehow express them as derived rules and present them in a sequent form. So, that the assumptions are also clear.

(Refer Slide Time: 06:28)



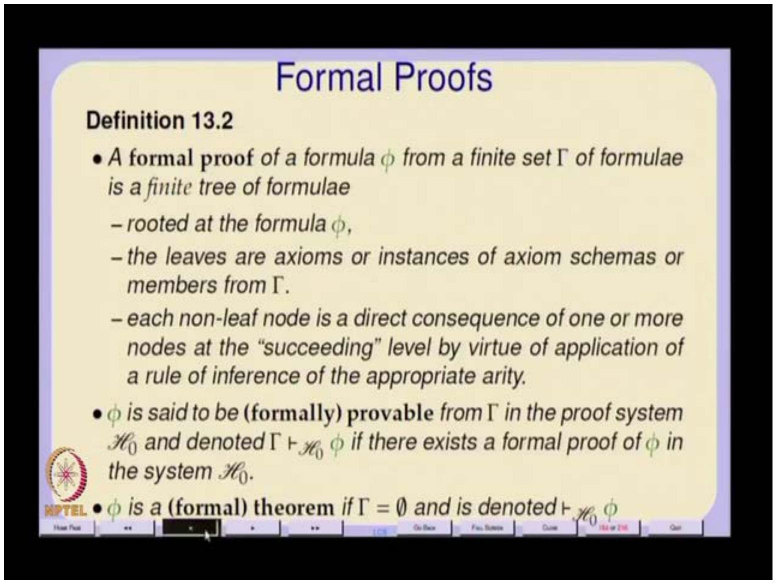
So, of course the proof that, we have already done of reflexivity for example. Is just is something that we can already incorporate so this is what we did.

(Refer Slide Time: 06:29)



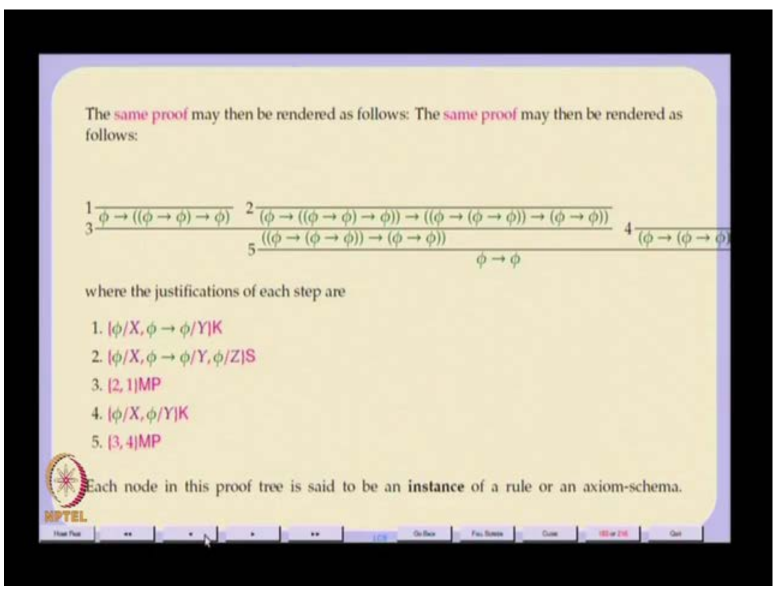
So, we can all make it a sequent by just putting a Lemma in front Lemma at a turn style in front.

(Refer Slide Time: 06:29)



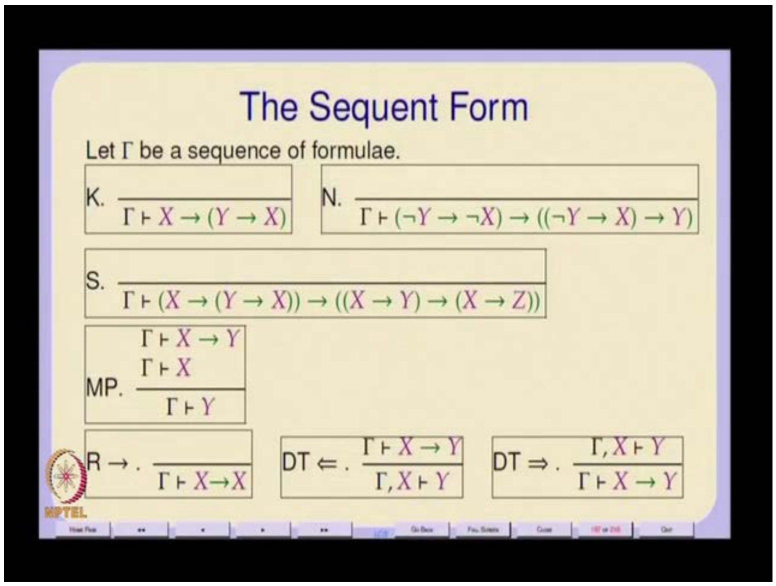
And it becomes is a sequent form of the proof I mean the basically since this since this theorem was proved without any assumptions and what that theorem of monotony says that any subset of assumption is also the same proof is valid.

(Refer Slide Time: 06:31)



So, therefore you can just prep end all these steps by a Lemma if you like for any Lemma Lemma could be empty it could be anything it does not matter. So, that is what we could do with this reflexivity but, what will. So, the other thing of course is that since we already proven reflexivity as a general theorem. Which holds for all formulae phi we may also express reflect reflexivity as a derived rule. And so, and then we can directly invoke this reflexivity that way our proofs will look shorter. And it will also ensure that our proofs remain non-circular we do not assume something we have already proved.

(Refer Slide Time: 07:47)



So, now the current state of the art so to speak of the Hilbert style proof system is in this in sequent form. So, what I have done is I have put in this extra set of assumptions. So, this gamma could be any set of formulae actually from the point of view of provability we will usually assume that gamma is a finite set of formulae. Now, gamma could be a we will essentially assume gamma to be a set there are people who become very nit piky and think of gamma as a sequence. Now, the difference between a set and a sequence is that a set does not allow duplicate elements. But, a sequence allows duplicate elements a sequence allows things to be moved across the turn style only in a certain order. In a sort of a queue or a stack fashion whereas, set we will just be more formal about it we can take it whenever, we want and if we want to have.

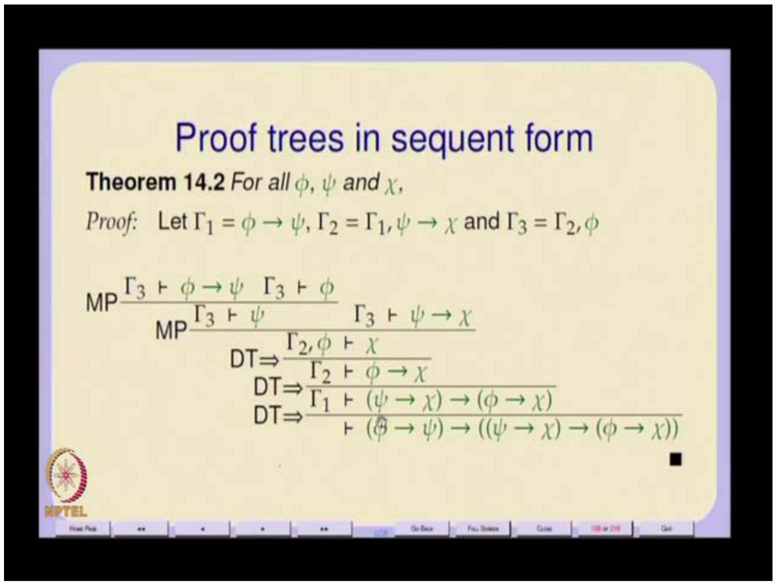
In the case of sequence sequences of formulae many people are actually nit piky about, also the order in which you also the number of duplicates that you exactly want to keep. Whereas, in a in the case sets will just assume that we can use the formula as many times as we want without worrying about, it is usage. But, there are interpretations of there are good reasons to believe in using those strict and rather pedantic methods which have to deal which actually come from a purely computer science perspective. Which is that you can think of these assumptions essentially as resources that you are being that are being used in your proof. And your proof is your program which essentially uses these resources. So, when you think in terms of resources and usage of resources it does matter whether you have duplicate resources or you do not have duplicate resources. Also it does matter how the resources are organized. And therefore, in what order you can move things to the left and to the of the turn set.

But, we will take a more in formal view we will take sort of logical view in which will just assume gamma to be a set or a sequence as and when as an how convenient to us. If, we assume it to be a set we will also assume that you can utilize the assumption in gamma as many times as you want in your proof without any hindrance to that. So, the 3 axiom schema K S and N now, look like this with gamma appearing in the turn style appearing there. So, this is let us say this is what we call the sequent form there is a much more rigorous and formal way of specifying the notion of the sequent. But, I am not going to go into it this is going to be my notion of sequence.

The modus (Refer Time: 10:53) rule actually now becomes much has a formulization which makes it clear that in all the three clauses you should have the same set or the same sequence of assumptions of gamma. You cannot have different gammas different sets of formulae floating around for each of these proofs. You have to get them somehow all the same set of assumptions. The reflexivity is actually is just this since, we proved it for all possible formulae. You can make it a you may make it a derived rule and a schemer with a with a variable like X. So, essentially X arrow X under any assumption gamma is perfectly fine.

The deduction theorem itself can be expressed as two new derived rules. So, this simply this actually specifies the movement if, from gamma I have proved some formula of the shape X arrow Y. Then I can claim to have proved from the assumption gamma union X. The formula of the shape Y and similarly if, I have from gamma and X if, I have proved Y. Then, I can claimed to have proved from gamma X arrow Y. So, it is very convenient to incorporate this otherwise just think of it the proof of the reduction theorem was quite long and complicated. And if you have to go through that process for every proof that you are going to do its going to become very tedious. And time consuming and very lengthy in simpler therefore to incorporate these things as derived rules you know. And just use them in their in a substitutable form. So, these are the derived rules so far we have got.

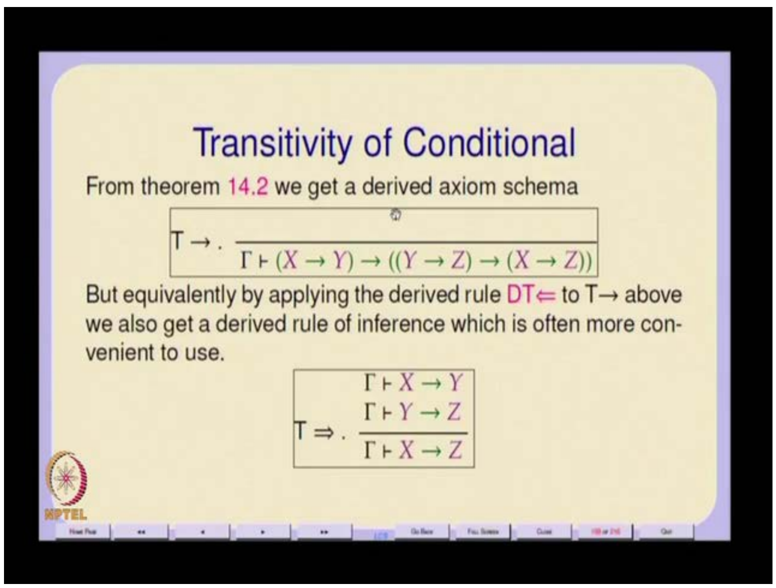
(Refer Slide Time: 12:56)



And then the next thing is of course, How are our proofs going to look like? Our proof in sequent form going to look like exactly I mean very much like they look before. Except the non assumptions are going to be specified explicitly. So, here is a proof tree for the transitivity of arrow at just look at this way just look at this carefully. So, if you remember we initially came we initially had assumed because we are going to use the deduction theorem. So, we took these three assumptions phi arrow of psi, psi arrow kai and phi itself from these three assumptions. So, gamma three consists of these three assumptions and so from which one of the things as I said was that any assumption is provable from that from itself.

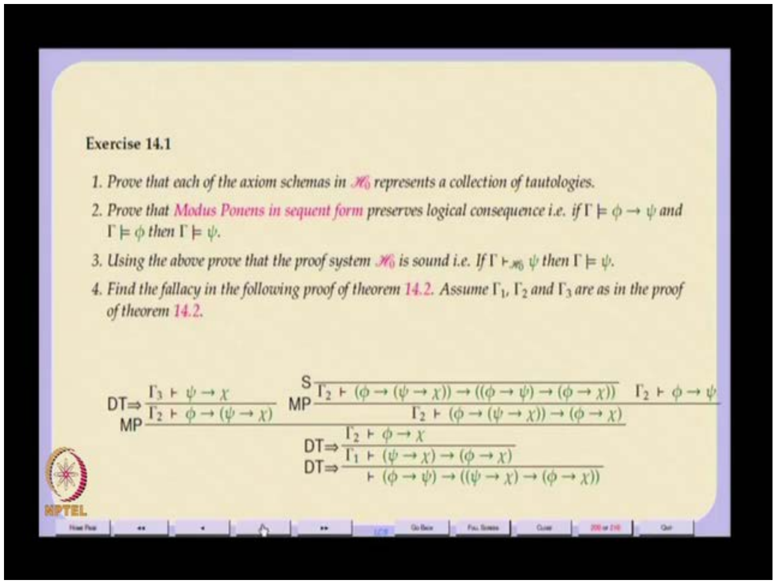
So, phi r of psi is essential to provable from gamma 3 and this reflects our set interpretation of gamma rather than a sequence interpretation for gamma not worrying about the order. The other is of course phi is provable from gamma 3 and by modus Refer Time: 14:22) essentially psi is provable. Which of from these 2 and of course one of our assumptions was psi arrow Kay. And since, psi is provable then what we have is a this is gamma 3 gamma 2 comma phi is gamma 3. So, from gamma 3 I can essentially prove Kay. Now, I have specified this as gamma 2 comma phi instead of gamma 3 because I am going to use the deduction theorem to move phi to the right. So, then the assumptions get depleted when phi moves to the right and gamma from gamma 2 you essentially proved phi arrow Kay. And now, the assumption get depleted more and more as I move each of the assumptions to the side of the turnstile So, our proofs in sequent form are going to essentially look like this. So, each node of the proof tree is going to be a sequent. So, it is not just going to be a formula it is going to be a sequent. Where, when we will be informal enough of course that when if the assumptions are known or there are not changing. Then, we will just write it as if each node has a each node is a formula rather than a sequent. So, this is the proof and of course this theorem holds for all phi psi and Kay. So which means that, we can actually render it immediately as a derived rule of endurance.

(Refer Slide Time: 16:11)



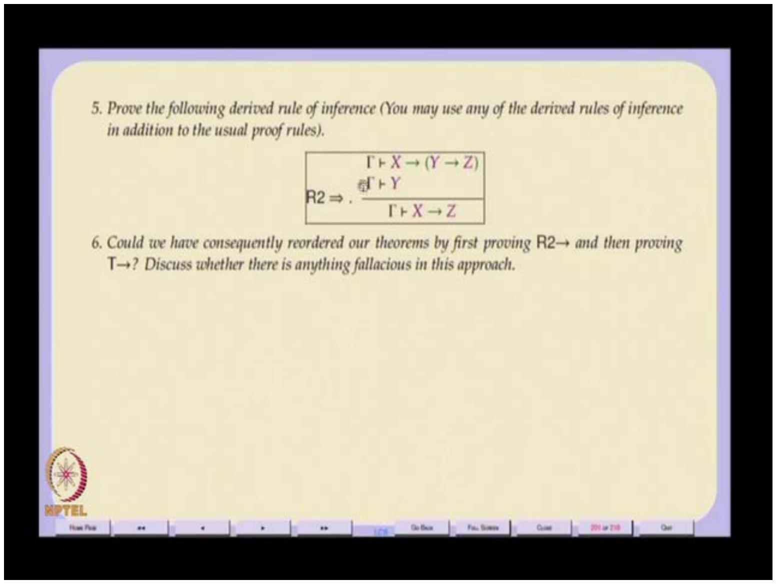
So, the derived rule of influence is this, so one thing is either we can think of it as a derived rule of influence. Either as an axiom schema in which we have proved this which corresponds to the last step of this proof. or we can we can or actually shorten the proof to this portion. And essentially think of it as a proof rule which gives us the transitivity of the logical implication if you like. So, essentially by applying the deduction theorem we could backwards we could actually move all these assumptions behind. And therefore you could you could actually give something like this So, this is a so very often you will find this form of the rule more convenient to use than this axiom schema. Because it I s a this axiom schema is the huge I mean this the form of the rule splits it up into smaller formulae. And therefore, it is easier to use than in the form of these axiom schema. But, in both cases both proofs should be equivalent because in either case you will essentially be moving formulae to the left and of the turnstile by invoking the deduction theorem. So, these two the axiom schema and the rule of influence are both essentially equivalent ways of doing things. But, we will use but I think we can use whichever one is more convenient to us at any stage so, that is why I have given them two different names.

(Refer Slide Time: 17:58)



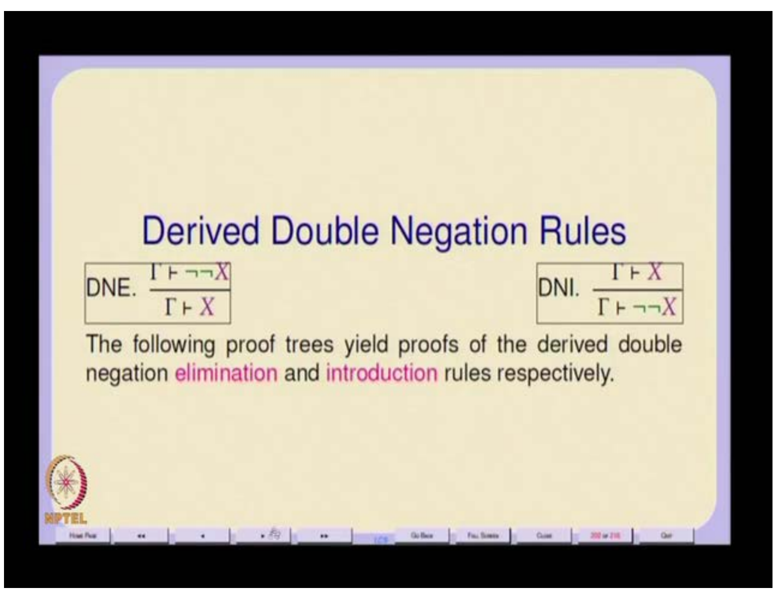
So, that is transitivity of conditional I have also got lot of exercises for you can do this there is a fallacious proof.

(Refer Slide Time: 18:11)



Which you can look at there are also some extra rules this here is an interesting rule this essentially says if, X arrow Y arrow Z and Y holds then X arrow Z holds. This is like this is something very strange I mean it is not you can take out you can pull out the between formula essentially if it holds. So, and sometimes it is convenient to use this as a rule of inference. So, we can we will use this in later proofs. Assuming of course that you have already proved this here.

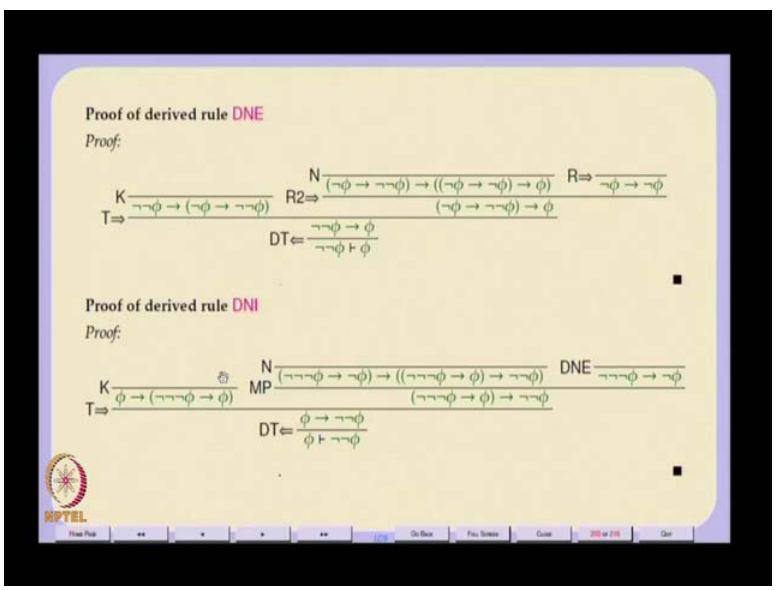
(Refer Slide Time: 18:11)



Now, there are some something's our formal theory should somehow be consistent with the known facts about propositional logic and provability. So, one basic thing one of the things we have so far not used is the negation rule remember the n the n axiom. So, here is so here to here are two derived rules. Which means that they have to be proven first and so now I am jumping the gun I am expressing theorems as derived rules. And then you have to come out with a proof. There is a specific reason why it is in this order. And that is you will see that that is because my proof at least I am not saying that is a unique proof.

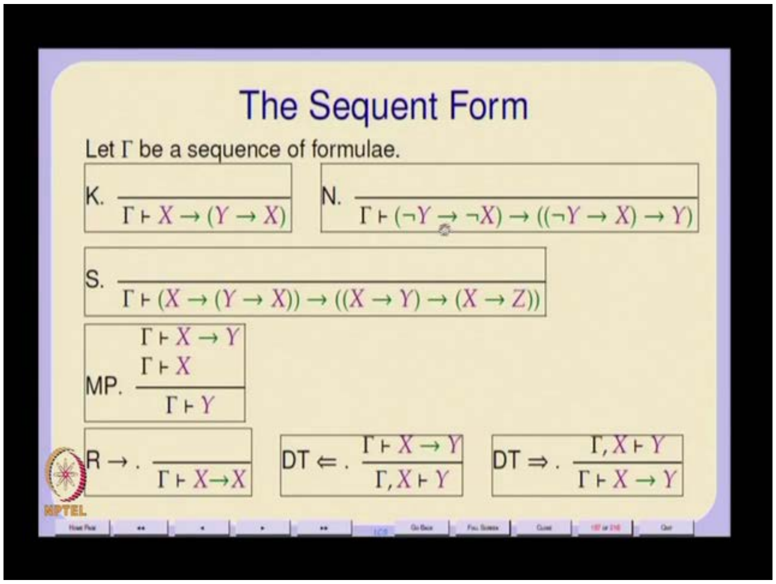
But, these proofs are quite hard so there are likely to be unique. The fewer the number of rules that rules and axioms that you have the more unique the proofs are likely to be. Once you add more and more rules and axioms, derived rules and axioms. Then you can have actually you can have a combinatorial explosion of a number of different proofs for the same kind of theorems. So, in this particular case actually this so this DNE stands for Double Negation Elimination and d NI stands for double Negation Introduction. So, these are some two basic things that we should prove.

(Refer Slide Time: 20:23)



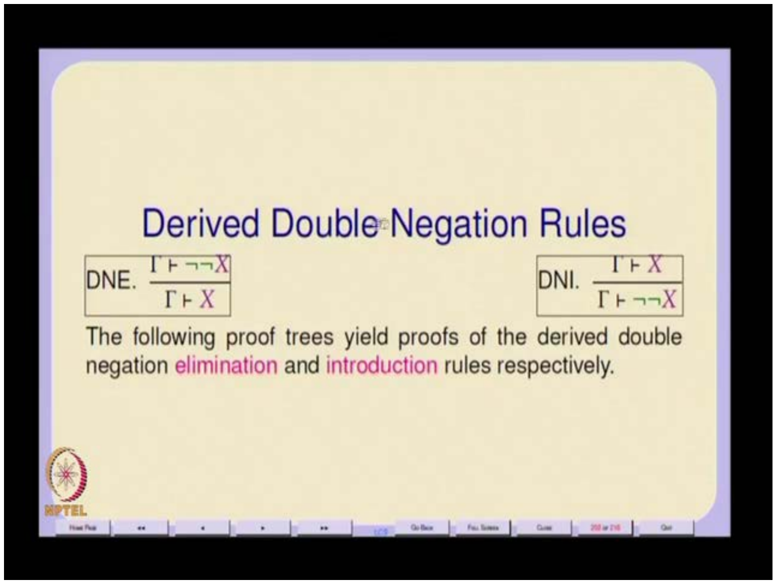
So, here is a proof for that so for the first time now we are using the negation axiom. So, axiom schema we had K S and N so far we have not used N. So, for double negation elimination we use this N so here. So, I have not specified the substitution that, have to be performed. But, now there are obvious so, which is that I did not have a link for that N rule. But, I should have a link for the N rule so anyway.

(Refer Slide Time: 21:26)



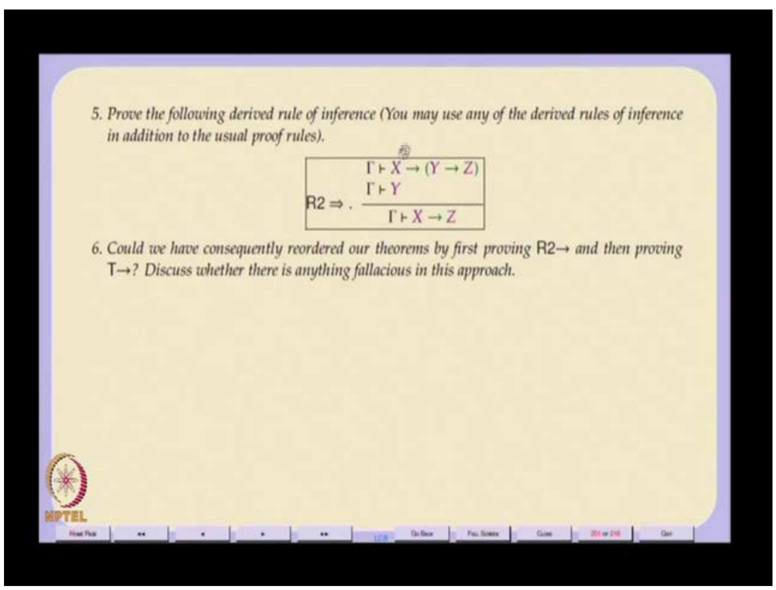
So, this rule is essentially says naught X arrow naught Y arrow naught Y no Y arrow naught X arrow Y I think that is what it list should naught Y arrow naught X arrow naught Y arrow X arrow Y. So, essentially I am substituting phi for Y and naught phi for X I think. So, that is lets go back to that phi for Y. So I get phi naught phi here and naught phi for X so I get naught phi and this is Y and this is naught X arrow X Y phi for Y Y. So, this is an instance of the rule N and of course we have already proved for all formulae we have proved the rule R R arrow. So, this between thing naught phi arrow naught phi holds and therefore, I can conclude and from R2 which was the previous one.

(Refer Slide Time: 22:27)



Where you could remove this middle thing if you have proven it follows that. That I can remove this middle portion and naught phi arrow naught phi to give me naught phi arrow naught phi arrow phi.

(Refer Slide Time: 22:29)



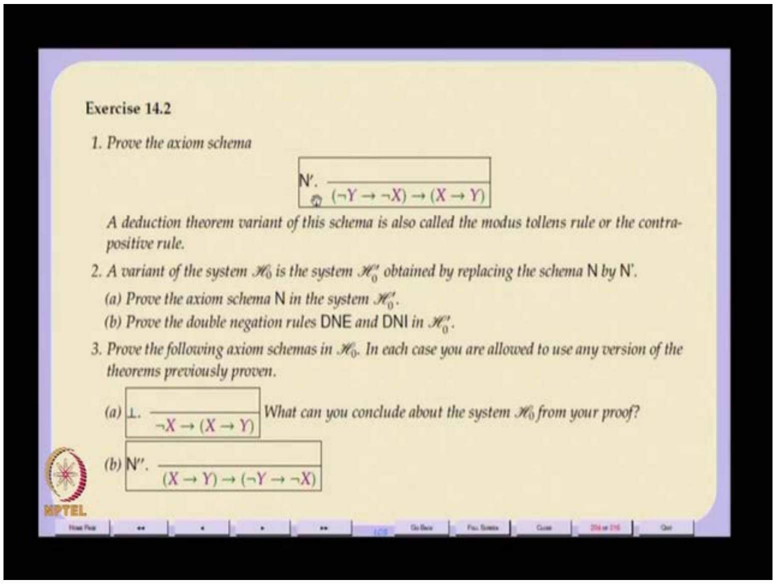
Then from K of course you get naught phi arrow or naught phi arrow naught phi. And we have the transitivity of conditional applied to these two. So, naught phi arrow naught phi arrow naught phi and naught phi arrow naught phi arrow phi. So, these two give me naught phi arrow phi and then there I move this antecedent to an assumption and get naught phi proves phi I mean it is not at all easy to think about this proofs. But, it becomes much easier once you have a collection of basic theorems you know. And basic therefore, in our case basic derived rules so some fundamental derived rules.

This, the double negation introduction requires using triple negation. Because yet to so here is the first application of the N rule with Y being naught phi X being phi. So, you do that substitution and you get this. And an this double negation elimination is can already be taken as a rule in which I can take triple negation phi arrows single negation phi. And then I can apply modus phonons I can get this again I can take an application of the K rule. And then I can apply transitivity of arrow to give me phi arrow naught phi. And there then I can move phi to the left of the tensile. So, you see that I have mixed as I did not start actually in a sequent style. But, that is because you assume that if I do not put it in a sequence style then I am assuming an empty set of assumptions gamma is empty.

So, when I actually move the when I get into a sequent style I put in the turn style whenever necessary. So, this is all informal most logic books would actually go either completely rigorously non sequent style or completely rigorously in a sequent style. But, we will mix these things because then it becomes too par antic otherwise. So, it is a good idea for us to mix it, provided we make sure we know what we are doing and that we are not in consistent in anyway. Student: Do not have (Refer Time: 25:33) which is derived N rules to proof K N K N N N S U. So, that we do not even (Refer Time: 25:39) do these or looking rules K and N N S. Once you have a got a sufficient number of intuitive derived rules. It would not be necessary to use K and S. Because you will most of the time be using that using the derived rules. But, you have to look at the philosophy of Albert Einstein proof system you are looking for a minimal set of axioms. And rules of inference from which everything else can be derived.

So, therefore for your initial basic theorems you will require K and S K S and N actually as we have seen. In fact they taught N we cannot have proved this see that what I am trying to say now, is look at this. Once you have got this these two rules become very convenient to reduce an odd number of negations by removing all by removing even number of them and simplifying them. But, the point is to get these derived rules you do need to use basic rules. And David Hilbert in his greatness came up with these two rules K and S and N. And actually those rules are complete, I mean there you require only one proof rule that is modus phonens. And in fact all model all logics would have a modus phonens rule. And he just requires three axioms schemas. But, till you have derive all those all are important basic theorems you will not you will have to use K and S.

Student: So, we can use we can have a set of rules derived rules with K and (Refer Time: 27:40)That is what we are going to do that is what we are going to do we going to do it for our convenience. You can have a set of derived rules which you can completely use you will see. (Refer Slide Time: 22:29)

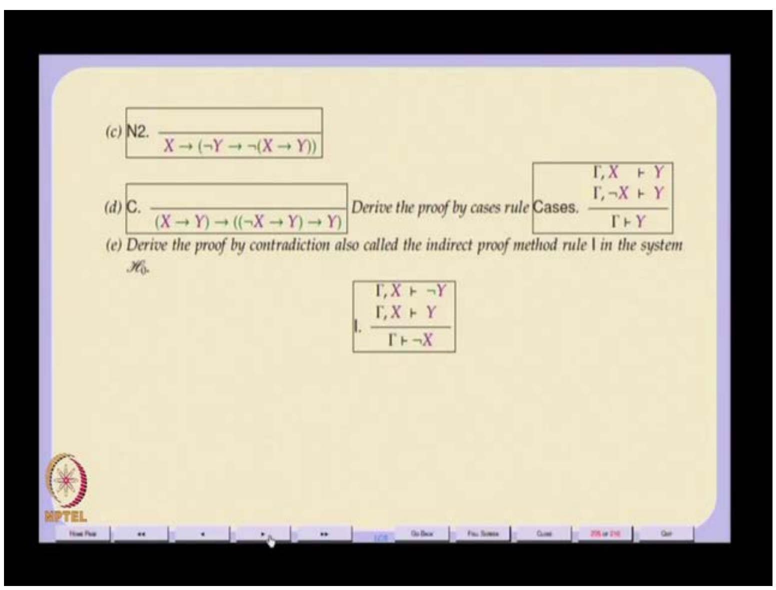


So, there is another version of the Hilbert (Refer Time: 28:00) system which instead of N uses N prime. This, looks somewhat simpler and actually more intuitive. You know it i well the converse of this is that if X implies Y then, naught Y implies naught X. The converse of it would be that but this is somewhat simpler to. So, it is possible to remove N put an N prime instead keep the K S and the modus pohens rule has they are and still get a complete system. And these exercises here, are ask you to prove that it for example you can derive n from n prime that is one question. Can you do the deduction theorem in this new system H naught prime. Well the deduction theorem need not use N or N prime. So, the deduction theorem will actually hold without any change.

So, you will get those derived rules anyway the only thing therefore you need to do is to derive is to prove N from N prime. And then, after that you can essentially follow Hilberts. So, these things actually tell you these exercises are meant to show that there may not be a unique system. There may not be a unique minimal system you can have more than one unique minimal more than one minimal system and this is a slight variation of that. The next thing is of course is this is the N double prime is the actual contra positive that i was talking about. If X or a Y then, naught Y arrow naught X but, of course we are talking about syntax in our proof theory pure formal proof theory. So, you cannot assume that N prime and N double prime are the same.

They both have to be derived by using only K S N and whatever, derived rules you have proved so far. So, as a general principle will assume that whatever, we have proved so far we can use that is how theories are built. This is how for example geometry works there are some basic axioms and postulates about lines, plains and points using just those basic axioms and postulates you prove some elementary things. And later actually in the later parts of (Refer Time: 30:38) geometry you hardly refer to these postulates. You always use the previously proven theorems that is exactly what happens and in this case too.

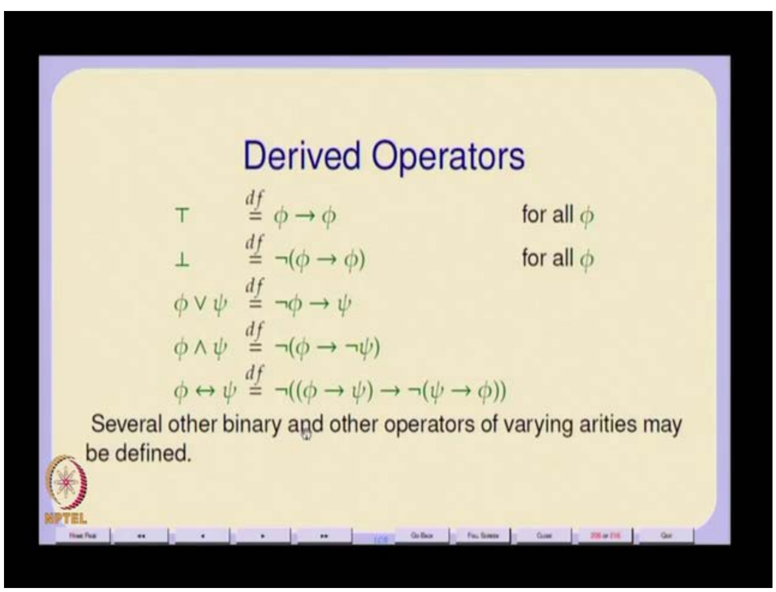
(Refer Slide Time: 30:54)



So, there are certain things for examples there is a, we should not lose the sight of the fact that we have certain principles in mathematics. One is proof by case analysis that is something.

Especially in computer science we keep doing a lot of that structural induction is really like case analysis or so on. So, proof by cases is the rule that you have to justify so this rule this is gamma if a gamma X proves Y. And gamma naught X also proves Y. then, gamma proves Y. There are two ways to interpret this either that the truth of Y depends only on gamma and does not depend on X or naught X. But, very often we do case analysis supposing this or so also there are two possible cases either this is true or this is false. And then in both cases it leads to Y so you can think of this as a proof by cases. You know justifying proof by cases and that the Hilbert system allows you to derive this proof by cases rule. And therefore, allows you to use proof by cases and most importantly indirect proofs methods there is proof by contradiction. So, from this from the assumption gamma comma X. You can prove both naught Y and Y then, from gamma you can claim to have prove naught X this is our standard proof by contradiction. So, and this can be proven also as a rule I mean so these are things that you have to do. So, these are certain basic proof methods which we have to justify from. The main question is that's the Hilbert system with just those 3 axioms schemas and one rule of influence. Is it powerful enough to capture these kind of proof methods that are used in mathematics? And answer is if you can derive these rules that it can actually give you these proof methods.

(Refer Slide Time: 30:54)

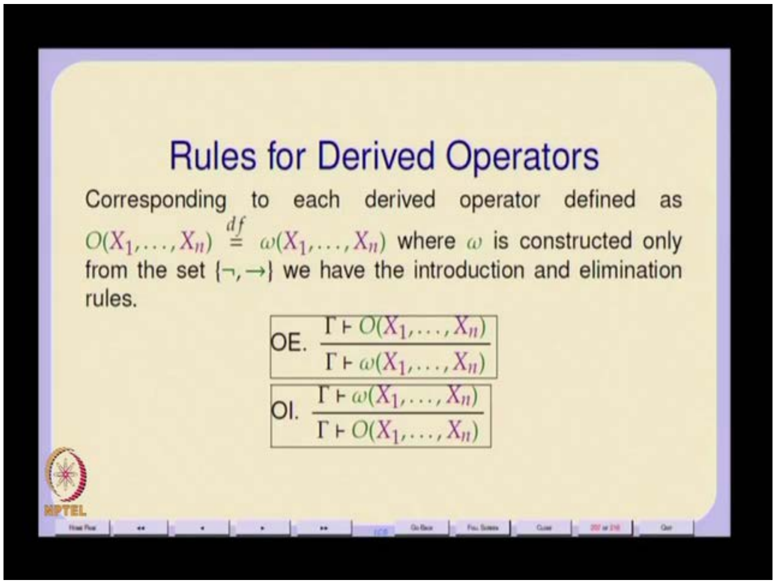


Now, the next thing is of course is that the Hilbert's system was minimalist in more ways than one. Firstly it used only two operators.= so whereas, most of time when we think of propositions we think in terms of and, or and not I know with that is our normal things are intuitively clearer when we think of them in that form. But, Hilbert chose this adequate set consisting of just not an arrow from a minimal view point. What it also what it means is that, it simplifies the number of cases he has to consider in all his proofs. If you have he has to prove any properties about, his system he needs to consider only two operators. And if he and if they are adequate then that is enough. But if, you are going to come up with derived rules we might also come up with derived operators.

So, our common and intuitive operators can be defined as derived operators. So, this is this is the standard definition for these operators. So, and for completeness we have given all of them. Since, we had this language p naught essentially we want to we where, interested in the language p naught. But, whereas Hilbert forced us to look at a very small subset of the language p naught with L naught. The other thing is that you can derive you can define any other new operators you like. And nor X or maybe a turnery operator maybe if then else maybe a quaternary operator maybe a fiveery operator sixery operator whatever. But, since you have naught and arrow as addict as functionally complete sets of all operators just express this whatever, new operator in terms of naught and arrow.

Now, it is fine to have these derived operators but it is useless you can use them also in proofs. So, just like we did in the case of double negation the natural thing to do is to actually come up with operator elimination and operator introduction rules.

(Refer Slide Time: 36:00)



So, remember that this is definition this is a definitional equality. There are other kinds of equalities we have looked at syntactic equality was one thing. The other equality then, we looked at logical equivalence for example that is an equivalence relation but that is an congruence. But this is an equality by definition and since its and equality by definition what you are saying is that, what is one the left hand side is just an abbreviated name. For what is on the hand side I mean this is what a syntactic definition equality. Essentially says this that I am using this symbol on the left hand side as an abbreviation of a certain structure a certain pattern on the hand side.

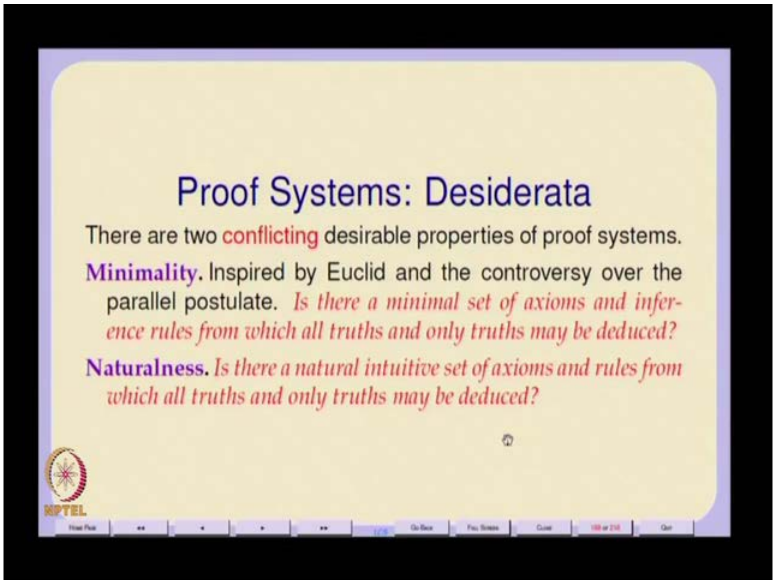
So, what we need our proof system of course otherwise will so whenever, that means by the definitional equality. What we are saying therefore is anytime in any proof we encounter one of these left hand side symbols we should be able to replace it by its definition on the hand side. Similarly if in any proof you encounter a, sub pattern which confirms to this kind of to one of these patterns. Then, I can replace that entire a sub pattern by a left hand side form. And that is exactly elimination and introduction if you encounter this operator then i can eliminate it by using the hand side body its definitional body. And if, I encounter this pattern anytime in the proof I can replace it by the left hand side and abbreviated basically.

So, the natural thing to do just like, in the case of derived other just like we did in the case of double negation. The most natural thing to do for any new operators that you might define is to have an elimination rule and an introduction rule. So now, you can prove theorems also about these derive operators. So, or, and, so on, so for so this of course so what we are saying is there is some you have a definitional definition of each of the operators expressed entirely in terms of tree structures. By the way these are suppose to be trees skeletons with hooks the purple X1 to X1 are the hooks on which you can hang other trees. So, these are so you can think of these hooks also as being classified into n different types n different colors only to distinguish them.

So, I might have more and more than one occurrence of X1. But, what I am saying is X1 is not the same as X2 but nothing prevents me from hanging copies of the same object in both X1 and X2 even though X1 and X are not necessarily the same means. So, everything that you learnt about substitution of variables actually comes up. So, these are skeletal structures of these operators are defined as essentially skeletal structures with place holders with names on them. Which identify two places with the same name and distinguish two place with two different names. But, do not prohibit the two different place holders from having the same object or copy of the same object on them. So, if I have an operator O defined in terms of some skeleton omega of my basic and derived operators just like I can use my derived rules. Now, I can also use my derived Y naught. And I can every time in any proof if, I encounter this operator O I can replace this by it is definition. And anytime in any proof if, I encounter a pattern which corresponds to this pattern omega then, I can replace it by its left hand side by preserving the names of places and the colors of places as they are that is important. So, basically for all derive operators you just requires some elimination and introduction rules.

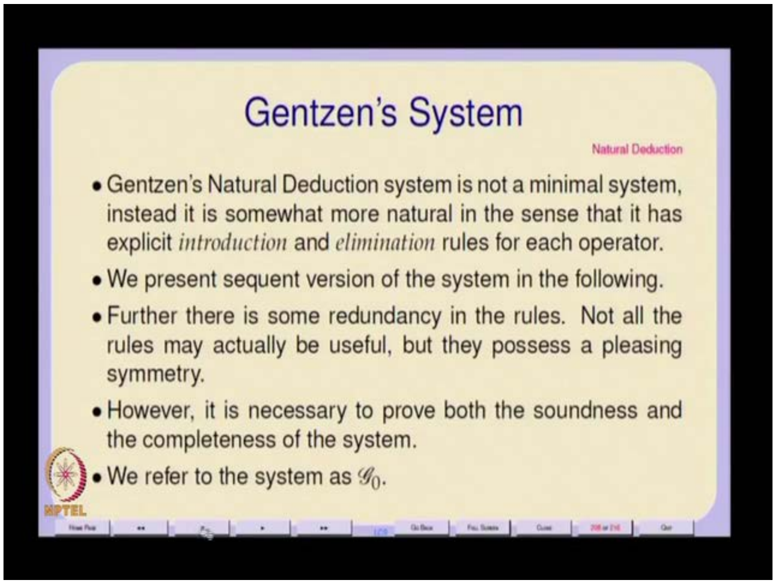
And then, whatever you get is as complete as Hilbert’s original system was as except that things become somewhat more intuitive somewhat more convenient for you to use that is it. And in fact what happened was that was one of the things that I had spoke about, previously is that they are two conflicting desirable properties of a proof systems.

(Refer Slide Time: 41:37)



One is Minimality which is essentially inspired Euclid's and the controversy over this parallel postulate. And the second thing is Naturalness how natural is it to apply certain rules it is clear that this I mean some of the proofs that you have seen are murderous. I mean it will never it will never do to be able to think about it, you require basically unlimited amount of time if you have to ever come up with that proof. And then there is no guarantee that you will be able to come up with the proof. But, so naturalness is something that gives you certain handles. Syntactic handles on how to probably go about, your proof process. And one of the fine and the most famous natural proof systems is that of Gentzen's.

(Refer Slide Time: 41:14)



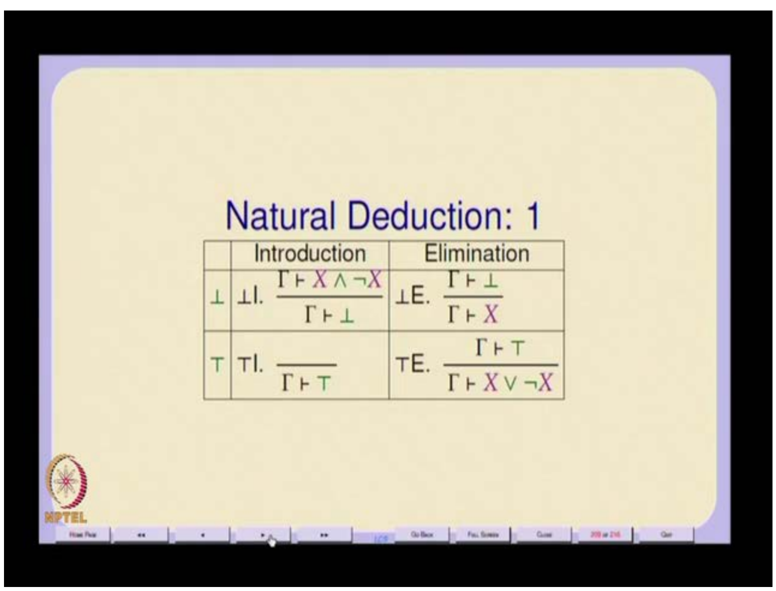
So, Gentzen's natural deduction system is something that is very similar I mean the Gentzen's natural deduction system was what inspired this way of most of you know in your programming languages course would have learnt about, structural operations semantics. So, the structural operations semantics is given in terms of proof rules. So it is actually a proof theoretic way of looking at a, computations through these rules through the operational rules. What all so, what is structural about it is the fact that for each operator in the language in the programming language there are one or two rules there is a complete set of rules which allow you to deal with the computation of that operator. So, it is structural in that so all properties of the system of SOS rules are essentially proved by induction on the structure.

Each transition in the SOS system actually give you a sort of a proof tree. Which is based on structure and breaking down the structure into smaller structure. And actually we have seen this in the form of tableaus earlier even here. Breaking down a given syntactic structure into its components using that breaking down policy that is somewhat more natural. So, in fact the tableau system itself was inspired by Gentzen. Gentzen so both plot as structural operations semantics for programming languages and Smullyan's tableau systems where actually inspired by Gentzen's natural deduction.

So, he actually called it naturally deduction because he said it is the most natural way to try to do deduction with syntactic objects. You look at the syntactic object it has a sort of tree structure and has a root operator. You apply some rule for the root operator and explicit in terms of sub components of that tree of that syntactic object. So, what it means is that then, of course you should be able to form this you should be able to not just break down syntactic object. You should also be able to form the syntactic objects from proofs. So, he had both introduction and elimination so in fact whatever I have written about introduction and elimination is related to essentially Gentzen's system.

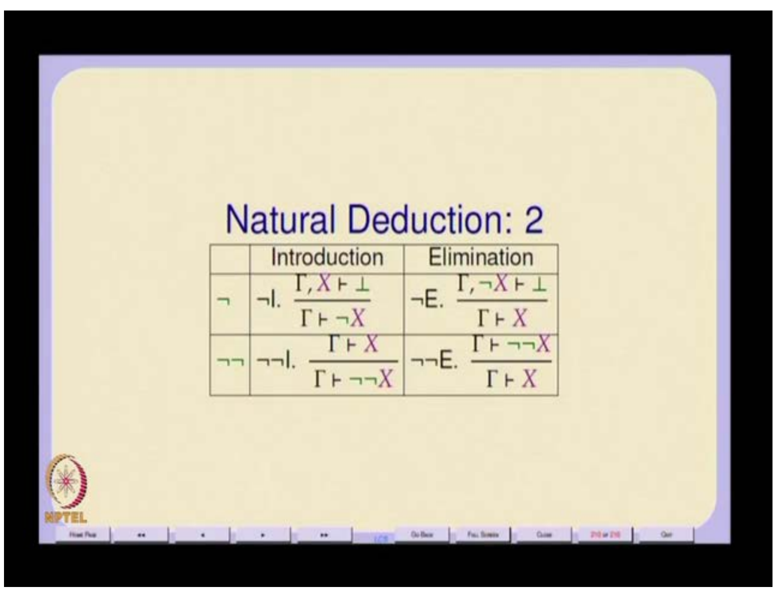
So will present a sequent version of the system it is not exactly what other people present. But I think it is a good adequate system further there is some redundancy in the rules very often we have in logic we have in interested in preserving certain symmetry is rather than check for Minimality. Once you have decided that Minimality is not important other astetic considerations are as important or more important. Then there is there is some redundancy in rules in fact some of the rules will never be applied at all. But, they are there for a kind of maintaining a, symmetry of the chart. First one thing is that, when you have these redundant rules you have to ensure that your system does not become unsound does not become inconsistent. That is a matter of of concern so, will call this system G naught after Gentzen's.

(Refer Slide Time: 46:34)



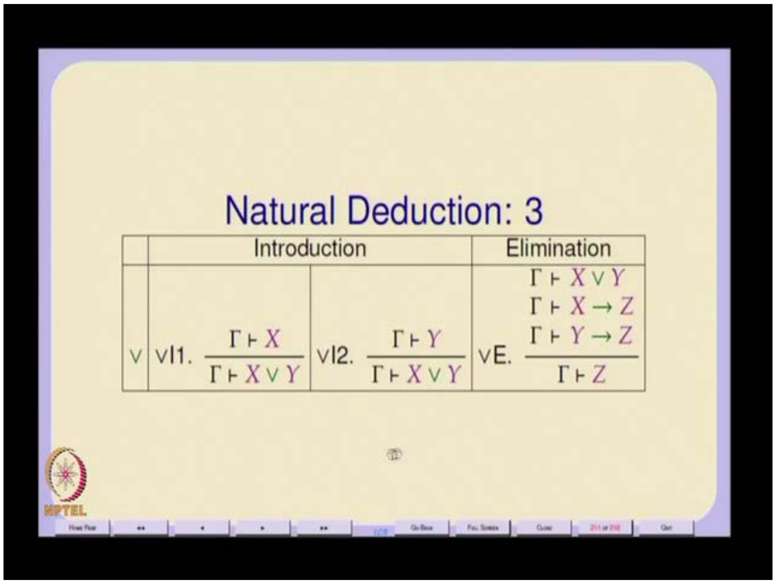
So, now you take all the operators that we have in p naught and define introduction elimination rules. So, for example take this top introduction basically what we are saying is that top can be introduced anywhere, I mean top is true. So, but the point is never actually going to use this rule anywhere I mean it is not going to be of much use. But, just in case for completeness to preserve both sides is we do this bottom introduction well anytime I have a contradiction in my proof I can claim bottom. Which is exactly the form of bottom I mean the definition this is sort of definition. And if i have ever proved a contradiction essentially then, I can actually infer for anything I want. You take a set of assumptions or axioms so, a system is inconsistent. If all the formulas in the system are proved let us go back to inconsistency from a model theory tech point of view. The semantics a set gamma is inconsistent means that it is already got contradiction. So, and of gamma would imply any formula would imply each and every formula apply in the system. A corresponding analog for provability is that if a system is if your system of set of assumptions is inconsistent. Then you can essentially conclude anything you like, any formula of the system is then the theorem of that system of that is a is a provable consequence of that set of assumptions.

(Refer Slide Time: 48:51)



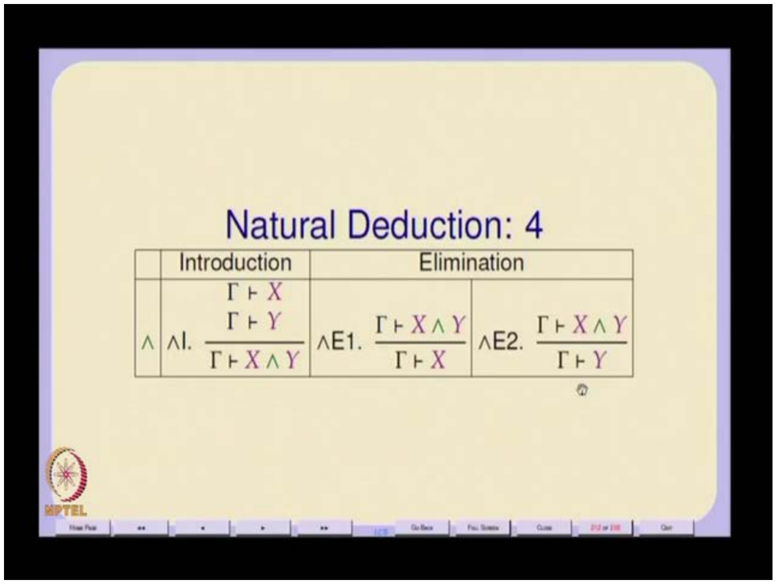
So, these are the introductions and elimination rules for bottom and top these are for negation and double negation. We have already seen that, I mean this is the so Lemma X bottom. Then, gamma proves naught X gamma naught X proves bottom then gamma proves Z. And so, you can see that this they are not completely mutually exclusive everything. We are just filling up all the gaps in a pattern back. So, this is the negation double negation introduction and double negation elimination we do need one of these at least.

(Refer Slide Time: 49:28)



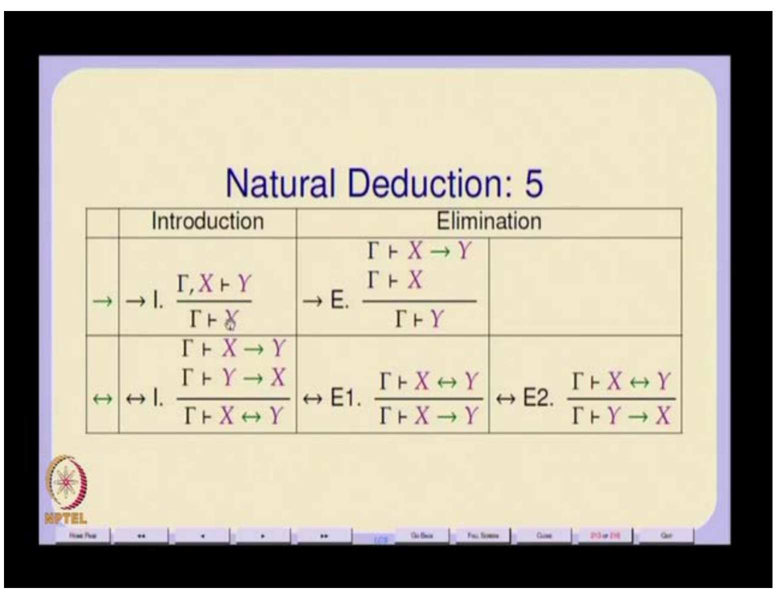
But, and then you have or introduction and or elimination the interesting thing that of course. So, or introduction is very simple you can introduce if you have proved X from gamma then you can introduce any Y it does not matter, And but the elimination is interesting. So, essentially what you are saying is if you have proved two cases X or Y if you have reduced your notion of proof to just the truth of X or Y. And if, X from X you can prove Z and from Y also you can prove Z. Then you do not require the case analysis X or Y. So, this is the more generalist form of case analysis and it says from gamma you can prove Z.

(Refer Slide Time: 50:14)



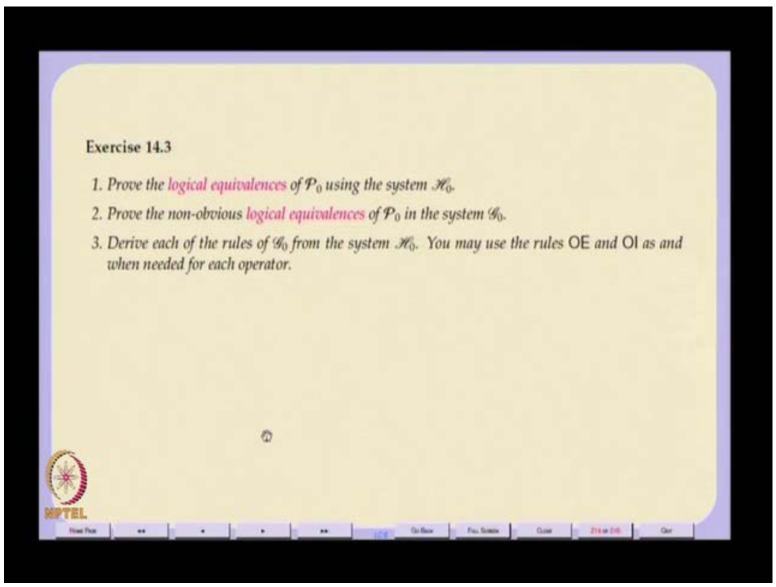
And introduction is very simple you have to be able to prove both of them in order to take a conjunction. And Elimination is very simple we have to prove the conjunction and then, you can take any component of the conjunction. Arrow Introduction is very much like our deduction theorem. So, if from gamma and X you can prove Y then you can claim that.

(Refer Slide Time: 50:42)



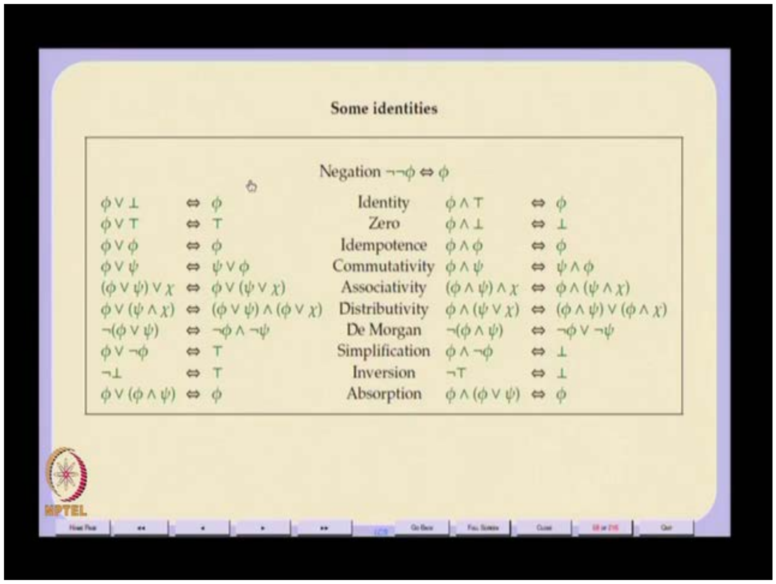
This is wrong it should it should read gamma prove X arrow Y I will correct that this is the modus phones rule arrow elimination is just the modus phones rule. And then, by the bi-conditional introduction and elimination are just as in they just it is the bi-conditional is just viewed as a conjunction of the arrow and its converse. So, these are all the thing the other important interesting thing of course is these are so you can see that, so Gentzen's system has a natural structure in terms of operators. So, for each operator you have introduction and elimination rules and you can use them in a definite. Instead of making it in definitional form like we did before, in the Hilbert style proof system you can explicitly use these rules.

(Refer Slide Time: 51:47)



So, now you are your job is to prove all of Gentzen's rules from the Hilbert's style proof system that is an exercise.

(Refer Slide Time: 51:58)



The other thing is of course all those logical equivalences should be provable as two way conditions in a Hilbert style proof system or some of them are already there in the natural deduction proof system. But, the others may have to be proven so with this but what we still have to worry about, are a few things soundness and completeness. The soundness of the systems are quite obvious because, all our axioms schemas are essentially structures of tautologies. At least in the Hilbert style proof system and you just have to prove the soundness of modus phones. That if a, and the other only important question is it complete.

That means can all logical consequences be also proven. Now, what does, what happens is there are there things called infinitely proofs. Infinite proofs are there some logical consequences which are only consequences of an infinite set of axioms or infinite set of assumptions well the compactness theorem tells us that that is not true that is not possible. So it is possible which is a finite set of assumptions to find to prove a, logical consequence. So, which means you should be able to get finite proofs for all. So, our completeness it is enough to restrict it to the question when Lemma is a finite set. And we have show that it is complete.