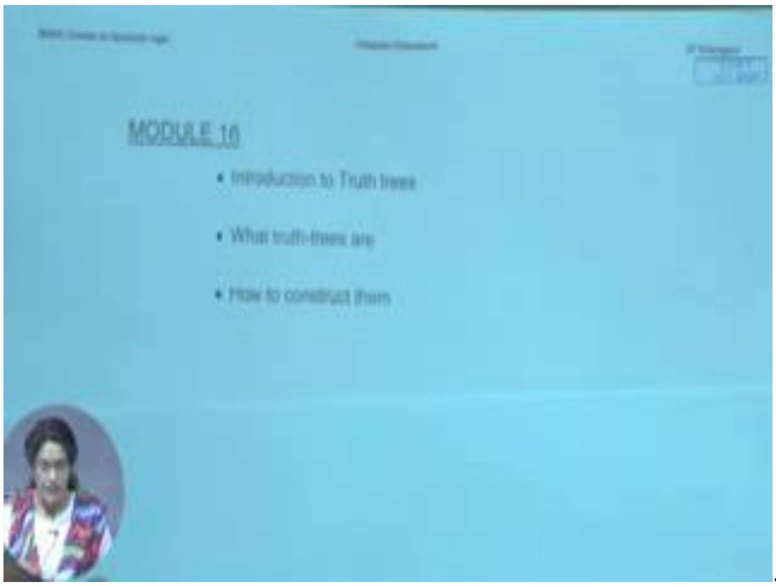
Symbolic Logic

Prof. Chhanda Chakraborti
Department of Humanities and Social Sciences
Indian Institute of Technology, Kharagpur

Lecture - 16
Introduction to Truth-Tress
What Truth-Trees are
How to Construct them

Hello! We are here back with more on the symbolic logic course under NOC. So, we have been through the procedures, semantic procedures of truth tables. Now today onwards this is our module 16, and from sixteen onwards we are going to talk about the truth trees.

(Refer Slide Time: 00:38)


MODULE 16

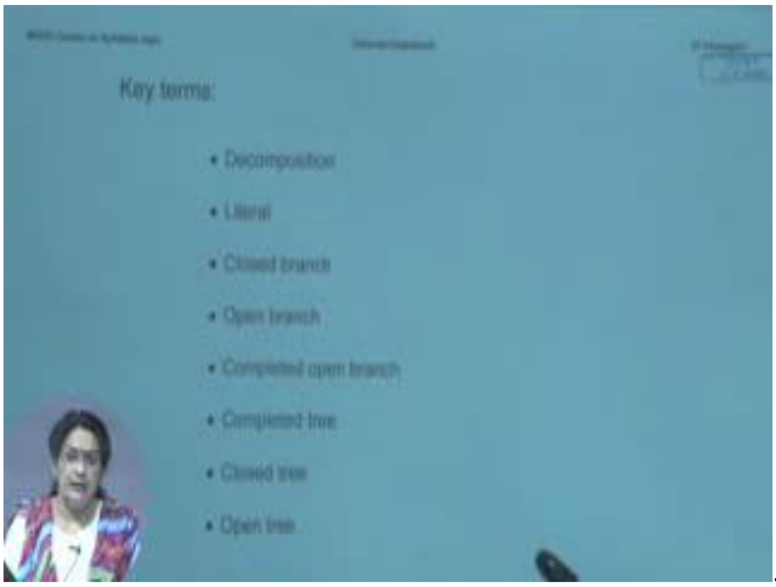
- Introduction to Truth trees
- What truth-trees are
- How to construct them

.

From 16 to 20, we will try to devote the time only on this procedure called the *truth trees*. So, this would be the module where we are going to be introducing to you, the truth trees: What they are and how we can construct them, etcetera.

So, basically we… I will be acquainting you with the procedure, explaining the steps as well as the basic components, basic ideas and so on and so forth. So, this is what is coming up in our this module, and there will be a spill over from the introduction to truth trees in our module 17 also. So, if there are things that are slightly unclear in the module 16, you try to learn from module 17 also. Of course there will be a continuation and so on and there will be also small problems to… I mean sort of, do with me. So, have your pen pencil and the paper ready and let’s go and start this truth tress.

(Refer Slide Time: 01:59)



Key terms:

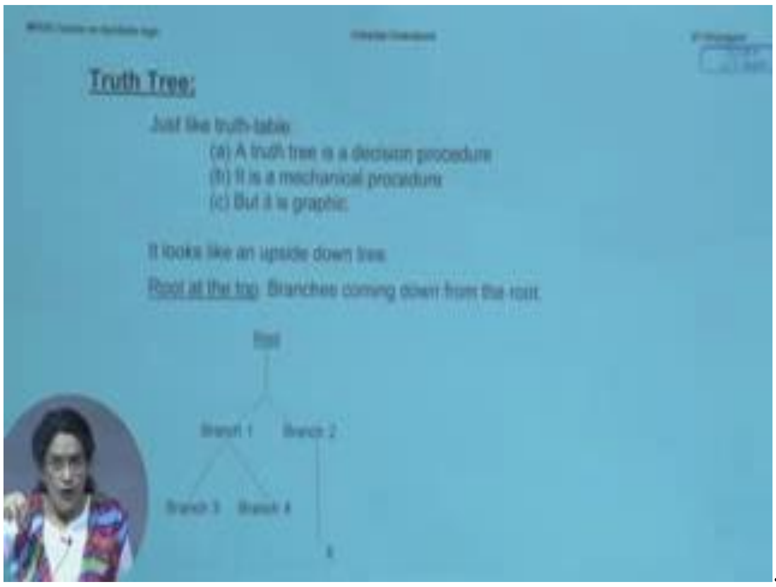
- Decomposition
- Literal
- Closed branch
- Open branch
- Completed open branch
- Completed tree
- Closed tree
- Open tree

.

First, I think what is needed is to get acquainted with the *key terms* of truth trees. So, just like, you know, when you go a new city, you need to know the name of the area, the names of the streets, the name of the place. Similarly, when you are coming into something new, truth trees, then we need to know these new key terms. I say this is a new procedure, because I have a feeling that you may have heard about truth tables, may be you have done truth tables, but you didn’t… were not so familiar with the idea of truth trees. So, we are all learning it together for the first time.

So, there will be terms like *decomposition*, and I will explain as we go along the processes. Just note that these are terms that you have to get acquainted with, then there will be *literals*. Of course, if it is a tree, there will be branches, but there are certain things called *closed branch*, *open branch*, *completed open branch*, *completed tree*, *closed tree*, *open tree*, etcetera. So, each of this we will try to, sort of, visit as we go along, understand and then try to see how this can be a procedure and what this tells us. In terms, you know, a procedure by itself, as I told you, is often meaningless, unless you pose some questions to it. So, similarly to the truth table, we will try to get the truth trees work for us. So, before that, it’s important that we sort of understand, pause, take time to get acquainted with these things. So, this is where we are starting out.

(Refer Slide Time: 03:39)


Truth Tree:

Just like truth-table

(a) A truth tree is a decision procedure

(b) It is a mechanical procedure

(c) But it is graphic

It looks like an upside down tree

Root at the top. Branches coming down from the root

.

First, what is a truth tree? Let me just tell you that, you know , it is… just like the truth table, what it is, is a decision procedure. By decision procedure, we mean that within finite steps, we will be able to find an answer to the question that we are posing to it. So, basically it will give you certain answers, that you are going to pose questions to. But before that, the procedure has to be learnt. And that this module is about that; this and the next module is about that: To learn, to master this procedure that we are encountering. Just like truth table, it is also a mechanical procedure. A mechanical procedure, as in the sense that there is algorithm for doing this, and then you can start somewhere, and so on, you continue with it. But it is less mechanical than the truth tables. I say this with full responsibility that in truth tables it's absolutely mechanical. You fill out the rows and the columns, and the answer sort of comes out. But you will see soon that there is something about the *tree management*. The truth tree, unlike the table, can be managed and needs to be managed from time to time. You know, just like the trees, you know, they have a way to grow. So, if you not careful it might grow completely uncontrollably, which is not what you want. You want the procedure to be effective, efficient and give you the right kind of answers that you are looking for. So, it is a mechanical procedure, but as I said, in comparison to truth table, it is less mechanical and you will see that, you will feel that.

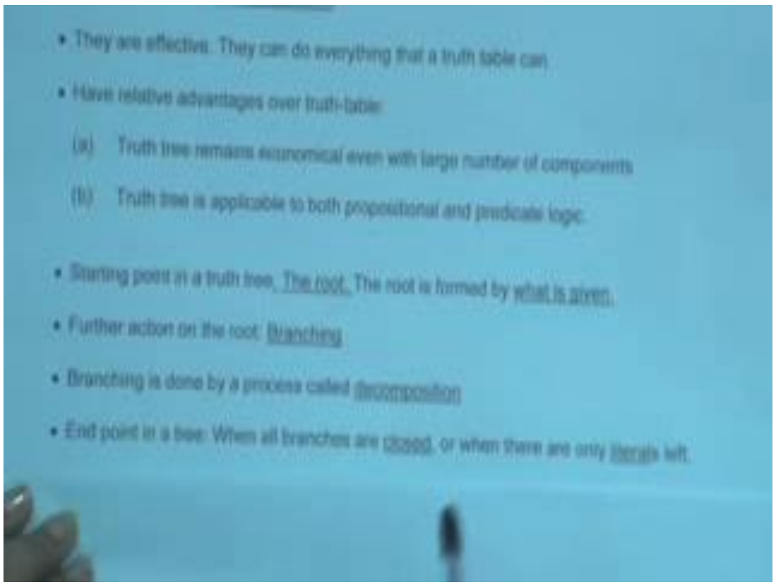
What is rather important about the truth tress that it's rather visual, it's graphic, it shows the whole structure in front of you. And we will show you some examples like that.

When I say it is graphic and visual, there I have a structure in mind. So, when you say *tree*, normally the picture that comes to your mind the roots are down and the branches are all out; they all full of leaves, and so on. But the direction of the leaves and the branches is up; the root is down. Now reverse that picture. So, the truth trees are like *inverted tree*; up-rooted tree. You know, in a storm you may have seen the trees get uprooted and they just fell over and so on. So, think about like an upside down tree. So, the roots are up and the branches are down. Let's see. So, this is something that you need to first of all get acquainted or sort of get oriented to.

So, this, for example, is a schematic picture of a tree what the tree might be look like. See, this is the root, this is what we will call the root. How the root is formed? What goes into the root? All of that will be discussed. Right now we are trying to understand the structures. So, here is the root and then there will be the branches coming out. The branches coming out. So, all the branch operations will come from the roots and towards down, and that is how the tree is to be done.

So, as we said, the first thing is that get used to the idea that this is what is needs to be done: That we will have the roots, and then slowly you will do some operations to generate branches, and the branches will come down and down in processes. This is what truth trees going to be like. This process that we are calling truth trees is also known as the *semantic tableau*.

(Refer Slide Time: 07:20)



- They are effective. They can do everything that a truth table can.
- Have relative advantages over truth-table:
  - (a) Truth tree remains economical even with large number of components
  - (b) Truth tree is applicable to both propositional and predicate logic.
- Starting point in a truth tree: The root. The root is formed by what is given.
- Further action on the root: Branching
- Branching is done by a process called decomposition
- End point in a tree: When all branches are closed, or when there are only literals left.

So, semantic tableau; this if you are looking for the truth tree material, then you might look it under semantic tableau also. One thing that needs to be established at the very outset is that they are effective procedure, and they are as capable as the other procedure; namely, a truth table. They can do everything that the truth table can. So, there will be occasions when if you know both the procedures, truth tables and truth trees, you would like to see where to apply the table, and where to do the trees. Because there is some sort of a relative advantage of this procedure. So, one thing that we have to mention, as I said, is: Why are we learning the truth trees even after we have done the truth tables? Is there any benefit? Is there any relative advantage of doing that? The answer is yes. The answer is yes. So, doing the truth trees in certain occasions, in certain situations, will give you better result. In fact, there will be occasions when you cannot apply truth tables, but that is where you are going to apply truth trees.

So, in a way the procedure is available to you, that itself is a benefit. So, first of all, that you know, when truth tables, if there are too many variables, too many constants, then the number of rows grow uncontrollably large and sometimes that presents a complexit;, sometimes that creates sort of operational difficulty, and there may be errors so on. Whereas, the truth trees, even with large number of components, may remain economical. So, right there, there is an operational benefit. So, the time, processing time is less, as well as, you know, you can keep the, you can do the results without getting confused about your own work. Second, and this is a very important point, is that, you know, we are right now into the level of propositional logic and I have mentioned that there is going to be First Order predicate logic. These two domains are very different. Truth tables are extremely effective in propositional logic level; so are truth trees. But when we go to predicate logic domain, even in first order predicate logic domain, the truth tables are not effective in the sense they cannot even be applied. So, because they need certain kind of statements to apply to, and those kinds of statements are not the only kind of statements available in predicate logic. There are different first order predicate logic statements, there are even propositions are also very different in nature. And the truth table cannot do any justice to that, truth table cannot be applied to them. Whereas, truth trees can be applied to those propositions. So, there is some relative advantage of learning these procedures.

Now, let’s talk about little bit about the basics. So, you know, every procedure has to have a starting point. Where do I start in the truth tree? The answer is in the root. As I showed, you got to start with the root. Now what is this root all about? Who gives the root? Where do we start? The answer is : Whatever is given. You know, in truth table also you had a starting point, right? What was the starting point? Namely, the given proposition, or the given set of propositions, or an argument and so on. Similarly, some given problem will be there on which you are going to apply the truth tree procedure. Remember that given is what is going to constitute the root. So, you start with the specifications given, the premises or the statements in a state of a proposition or a single proposition, for example. That is what the root will be consisting of: With what is given.

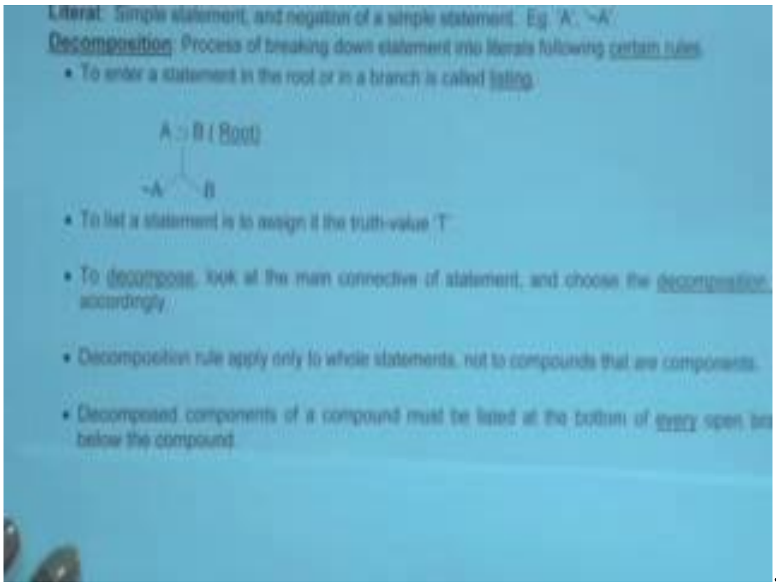
Then comes what is known as adding to that. So, the operation of adding to the root and you will see when this separation has to be done, we will try to explain it. But right now we are very much in the beginning. So, just understand that once this root is formed, then the operation that is to be done on the root is called *branching.* Ok? Branching. So, there you… the logical operation done on the roots is branching. How do you do the branching? We will try to show you. But there are branching rules which we will have to learn and we are coming there, we are slowly going to get there.

The process by which you do the branching is called *decomposition*. You will see in the next module, I will explain it in greater details, is what this is, is a process of breaking down. Decomposition , you understand, that we are taking things apart. So, what was composed, you are decomposing. Now here what you are decomposing is going to be truth-functional compound statements. So, if they are there in your root, then you try to break them, those compounds into the simpler components; and there are rules to do that. So, we will show you that. So, this process as you do this you will see that the truth tree allows you to generate branches. And these branches, you know, just like in the trees sometimes they are linear, there is a single branch that is coming out, sometimes they bifurcate. And we will try show you when this happens and under what conditions this can happen, when the line remains linear, when it bifurcates, and so on; and we will try to show you that. But that is what the decomposition process is all about.

This was the starting point, the root. This is where as you can see the tree is growing through the branching, through decomposition process the tree is growing. But how long? Where does it end? If there is a starting point, what is the termination point for this procedure? And the answer is : When either when all the branches are closed, we'll try to understand this little later, either when branches are *closed* or when there are only literals left. Both of these terms, I understand, you don't understand, because it has not been discussed yet. So, hold it. But let me put it like this : That end point in the tree you are going to encounter when all your operations to be done are finished. All the decompositions that needed to be done are finished. That is the state when the decomposition process is completely over and only literals are left, or as a result of this decomposition, you have only generated closed braches. Closed branches are branches where no operation is to be done.

So, this is going to be a clear indication when the tree cannot grow anymore and you call it the closed of operation, the tree is done. So, this is the first thing that we need to sort of see. So, let me just recount what we just said. There is a definite staring point and there is a definite end point for the truth tree and in between the root, the starting point and the finishing point what we do are the this is called decomposition, the branching process and so on. We will take examples and will try to show you as we go along.

(Refer Slide Time: 15:24)


Literal: Simple statement, and negation of a simple statement. Eg. 'A', '~A'
Decomposition: Process of breaking down statement into literals following certain rules.

- To enter a statement in the root or in a branch is called listing.

$$\begin{tikzcd}[column sep=small] & A \supset B \text{ (Root)} \arrow[dl, dash] \arrow[dr, dash] & \\ \sim A & & B \end{tikzcd}$$

- To list a statement is to assign it the truth-value 'T'.
- To decompose, look at the main connective of statement, and choose the decomposition accordingly.
- Decomposition rule apply only to whole statements, not to compounds that are components.
- Decomposed components of a compound must be listed at the bottom of every open br
below the compound.

.

Let's talk about… I have introduced to you to the idea of literals, and decomposition I am going to explain in a second. What are literals? I have used this word. Remember literals in truth tree parlance is going to be a simple statement. Simple as in structurally simple statement, and its negation or the negation of a simple statement. Either way. So, for example; A would be a literal, similarly not-A or ~A also would be a literal. Fine? So, once more, literals would mean not compound statements, but simple statements such as A, B , C, you know, capital, we are using constants. So, either that, or negations of those simple statements. Now remember, negation of a statement is no longer simple according to propositional logic. But in truth tree parlance they count as literals. So, earlier when we said … when can you stop the tree? When you have completely brought the tree to the level of literals, we mean that only these kind of things are now left in the tree. And remember these are simple; so, literals you cannot break them further, you cannot decompose them further. So, at this level the decomposition process is completely over.

What is the decomposition? As I said, the decomposition is nothing but taking the truth-functional compound propositions and breaking them down into literals. How do you do that? There are certain rules. Right now these rules and everything we'll try to introduce to you slowly. So, maybe I will introduce you some of these rules today and save it for the next module tomorrow , because we all beginners. But remember that this is what the process we are talking about. This is how you generate branching. So, what are you doing? You are breaking the truth-functional compound statements which are given in the tree or in the root, and breaking them into what ? You continue to break them until you reach this kind of level. So, level of literals.

So, there are certain rules of decomposition. We will learn them together.

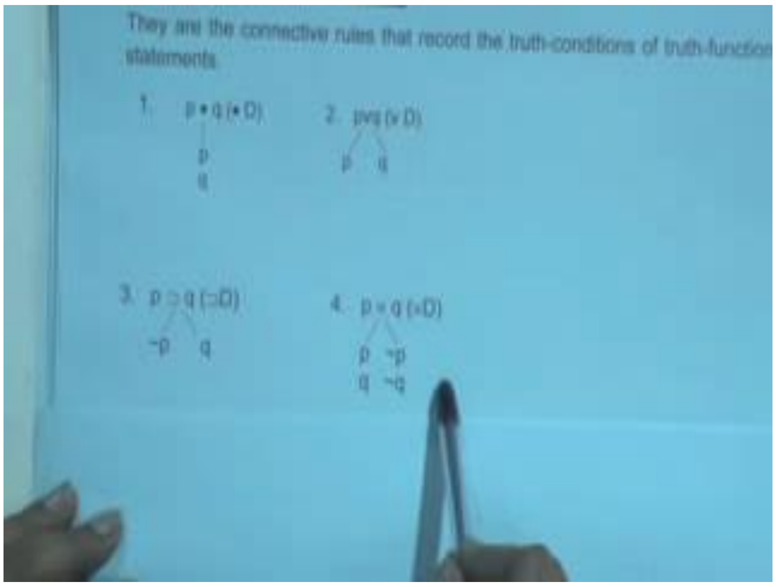
Next point that I think is very important for having an insight into the procedure of truth tree is that whenever you are entering a statement in the truth tree, either in the root or in the branch, as a result of decomposition , that process is called *listing*. You are entering something in to the tree and that process is called listing. So, whether it is in the root, whether it is in the branches, when you are entering something, its called listing. Why is that important? Because whenever you list a statement in a truth tree, you are claiming or assuming that its truth-value is true. This if you recall, if you remember these two points, you will have a much better grasp why the procedure… what the procedure is talking to you. Remember the procedure is… does not by itself say something unless you pose a question. But to find the answer, you need to have a perception about the procedure. So, the insight that will help you a lot to get inside this procedure is this. That when, what you doing in in terms of listing is that you are claiming certain components or whatever you are listing in the tree as *true*. So, nothing that appears in the tree is going to be false, alright? So, that is going to be your guiding vision in fact.

See this is a small example of how to decompose. This is in the root, and these have generated as a result of decomposing this compound. This is how the decomposing is going to look like. And we'll explain this when we see the rules better. But remember this is listing A ⊃ B, this is listing ~A, this is listing B also. Each time what you have claimed that this is true, this is true, this is true, right?

How do you decompose? And the answer is that you try to choose the rule that fits into the main connective of the statement. So, the decomposition rule is actually the main connective decomposition. Remember the main connective is the one that has the maximum scope over the sentence. And you try to see the rule that fits into the main connective. Right now I am going to mention this, but in the next module perhaps I will explain it. Remember, decomposition rule will apply to the whole statement and not to the part. This is another way of saying the same thing that it will only apply to the main connective and not to the sub-connectives. So, we'll see the examples as we go along, but right now we as I said I will briefly mention this. I will show with examples in the next module to carry the point forward better.

And this is another point to learn is that when you are doing the decomposition the result of the decomposition must be listed *at the bottom of every open branch below the compound*. Now this point, at this point, will be completely unclear to you because you do not understand the technical terms used here. But this is going to be a crucial point also for doing tree and I will show you what this means with examples as we go along. First I think this is time to at least introduce you to some of the decomposition rules and with this we'll end this module.

(Refer Slide Time: 21:32)



They are the connective rules that record the truth-conditions of truth-function statements

1. p • q (• D)
p
q
2. p∨q (∨ D)
p q
3. p ⊃ q (⊃D)
~p q
4. p ≡ q (≡D)
p ~p
q ~q

So, remember what is it that this truth tree decomposition rules are? What they do, is they take the connective and record the *truth conditions* of that connective. Remember in the tree whatever you are listing is assumed to be true or considered to be true. So, what are you doing when you are taking the connective? You are breaking it into what makes that connective true. Which conditions would make that connective true? Right? If you recall your truth tables, if you recall the connective's truth tables for example, tilde (~), horseshoe (⊃), wedge (∨), dot (•), you will have no problem at this point. Ok? So, try to remember that and then you will see that these decomposition rules are rather self-evident or intuitively clear.

So, let us see. We'll take first two. So, here is your root. The first one is p • q and this is your main connective dot (•) and you are asking the question what conditions make the dot (•) true? The answer is when both the conjuncts are true. And that is what is recorded here. See? This linear… that is a branch; it does not look like a branch to you, but it is a branch, it is a single direction, linear branch. And when p is true and q is true, p • q is true. This is what you saying. And that is a rule, a small rule when you are encountering a dot proposition or a statement in a truth tree how to decompose that. And the rule is going to be call dot (•) decomposition, in short we are writing • D, right?

So, this is one example of a truth tree decomposition rule.

Similarly what we have is a wedge (∨) decomposition rule. So, here in the root you have p ∨ q and when is p ∨ q true? When either p is true or q is true or when both of them are true. Remember the truth table? So, similarly then you see that we have captured the condition, truth condition of the wedge (∨) connective and we are calling it the wedge ∨ D. Get it? So, these are two examples. So, whenever you find a statement which has the ∨ as the main connective in your truth tree, you know that by applying this wedge ∨ D rule we can generate these two branches.

So, what is the difference between these? Here is a single branch. Whenever you have joint conditions to be true, you write it like this. Whenever it is 'either or', we sort of go into this bifurcated branch. This is branching, this is also branching. This looks to you usually probably more like branching. But this is also branching. Got me? So, this is for joint condition, this is for alternate conditions, but also together.

The last two rules for today are like this: here is your horseshoe (⊃), decomposition rule. So, what you are saying is that when is p ⊃ q true? And when is that? If you look into your truth table of the horseshoe (⊃), you will find either when p is false or when q is true. These are the two times when the horseshoe (⊃) is coming out be true. And that takes care of it. Remember it is also true when p and q are both true, but specifically if you look at it that whenever q is true, it doesn't matter what the antecedent value is; whenever p is false, it doesn't matter what value of the consequent is; you are going to have the horseshoe (⊃) as true. So, this captures the horseshoe (⊃) truth conditions and therefore, it is called horseshoe (⊃), decomposition rule.

This is your triple bar (≡) decomposition rule. So, triple bar (≡) decomposition rule, as you know very well, that this is the equivalence. So, whenever the values match, we are going to have the horseshoe, sorry, triple bar (≡) coming out to be true; whenever they do not match that is when the triple bar (≡) will be false. Remember in the decomposition rule, we are capturing only the truth conditions. What makes the triple bar (≡) true? The answer is either when p, q both are true, or when p and q are false. Get me? So, there are two sets of conditions, 'either-or'. So, that is why it has generated this kind of branching. But in one branch you have p true, q true and on the other side you have p false, meaning not-p is true, q is false which means not-q is true. Remember whenever we list, we list only the truth conditions. We cannot capture the falsity conditions. So, in order to express that p is false , what we need and we can do in a binary logic system is to capture that fact like this. That we say that not-p is true. Fine? Not-p is also literal. So, not-p is true and not-q is true. So, this gives us two separate conditions when the triple bar (≡) decomposition will be true.

So, right now we have four truth tree decomposition rules in front of you. So, if you see this kind of connectives appearing as main connective in the proposition of the truth tree, you know which rule to apply. With this I will end this module. There is a lot to learn yet, but we have made a good beginning and I hope things have been so far clear. So with that I will close this module.

Thank you.