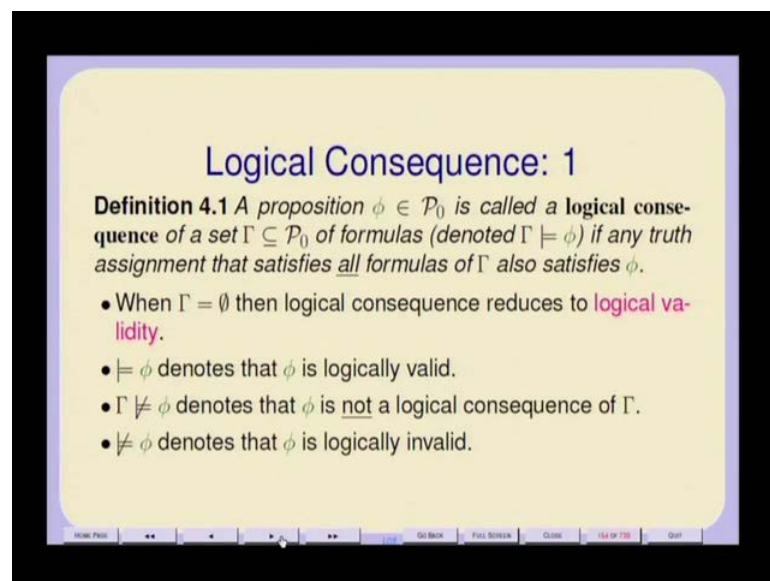


Logic for CS
Prof. Dr. S. Arun Kumar
Department of Computer Science
Indian Institute of Technology, Delhi

Lecture - 5
Identities and Normal Forms

Let us start lecture five, so before we do that I will like to quickly go through some of the stuff that we have already done in lecture four.

(Refer Slide Time: 00:46)



Logical Consequence: 1

Definition 4.1 A proposition $\phi \in \mathcal{P}_0$ is called a logical consequence of a set $\Gamma \subseteq \mathcal{P}_0$ of formulas (denoted $\Gamma \models \phi$) if any truth assignment that satisfies all formulas of Γ also satisfies ϕ .

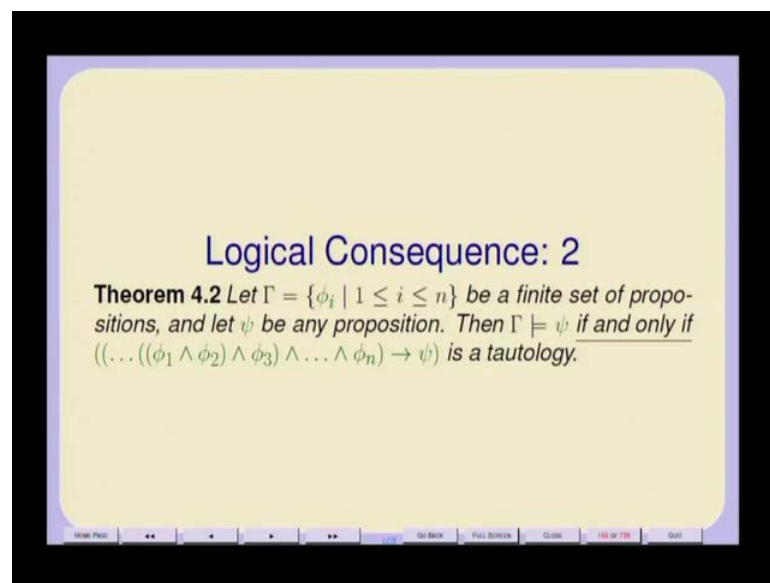
- When $\Gamma = \emptyset$ then logical consequence reduces to **logical validity**.
- $\models \phi$ denotes that ϕ is logically valid.
- $\Gamma \not\models \phi$ denotes that ϕ is not a logical consequence of Γ .
- $\not\models \phi$ denotes that ϕ is logically invalid.

For example, some of the concepts, here are some let us quickly go through these logical validity does not belong to the language. So, you are asking if you are asking whether topologic can ever be logically invalid, the answer is it cannot be there. I mean if you look at this validity, for example the validity symbol is black in color, it is metallurgical symbol. It is a symbol of the mathematics of the mathematical logic and not and is not a symbol of the language of propositional logic.

So, whatever is symbol of language of propositional logic would be green in color like this phi is green in color, though I am not sure whether you can actually see it, see the see the color differences. So, this we defined the notion of logical consequence, so we just say that a proposition phi is a logical consequence of some set of propositions gamma, where this set of this set of proposition gamma is a, it could be finite or infinite in general.

Let us say we will think of it as being finite, so basically if any truth assignment that satisfies all the formulas of gamma also satisfies phi, then you would say that phi is the logical consequence of gamma. When the set gamma is empty, we just call it logical validity, so essentially what you are saying is that the truth of phi does not depend upon on the any other the truth of any other assumptions. For example, that denotes logical validity and of course if you just strike through that symbol, then it denotes in validity or that something is logically in valid or it s not a logical consequence of some of a set of formulae.

(Refer Slide Time: 03:12)



This is logically in valid does not mean contradiction every contradiction is logically invalid that is true, but even every contingent we have is also logically invalid. A contingent proposition is one for which there is at least one truth assignment for which it will be true. There is at least truth assignment for which it will be false, so that is logically valid simply because there exists a truth assignment in which all the assumptions are true. They are not assumptions, so that is that revile holds and the conclusion is false.

So, that is any contingent or contradiction is logically invalid, so we have this notion of logical, so we have various these are actually theories of these are thermos of the theory of logic. Essentially, what these theorems tell us is that there is a translation of many of the metrological concept in to concepts in the language itself. So, in particular if gamma

is a finite set, then size any formulae then size a logical consequence of gamma and only if you can form this huge formulae and prove that if it is a tautology, then size trivially a logical consequence of gamma. So, that shows that many of the reasoning concepts that we use can also be expressed with in the language of logic itself in some other terms.

(Refer Slide Time: 05:18)

Proof: (\Rightarrow) Assume $((\dots((\phi_1 \wedge \phi_2) \wedge \phi_3) \wedge \dots \wedge \phi_n) \rightarrow \psi)$ is not a tautology. Then there exists a truth assignment τ such that

$$\begin{aligned} \mathcal{T}[(\dots((\phi_1 \wedge \phi_2) \wedge \phi_3) \wedge \dots \wedge \phi_n) \rightarrow \psi]_{\tau} &= 0 \\ \text{iff } (\mathcal{T}[(\dots((\phi_1 \wedge \phi_2) \wedge \phi_3) \wedge \dots \wedge \phi_n)]_{\tau} \leq \mathcal{T}[\psi]_{\tau}) &= 0 \\ \text{iff } (\mathcal{T}[(\dots((\phi_1 \wedge \phi_2) \wedge \phi_3) \wedge \dots \wedge \phi_n)]_{\tau}) &= 1 \text{ and } \mathcal{T}[\psi]_{\tau} = 0 \\ \text{iff } \mathcal{T}[\phi_1]_{\tau} = \dots = \mathcal{T}[\phi_n]_{\tau} &= 1 \text{ and } \mathcal{T}[\psi]_{\tau} = 0 \end{aligned}$$

which contradicts the notion of logical consequence.

(\Leftarrow) Assume $((\dots((\phi_1 \wedge \phi_2) \wedge \phi_3) \wedge \dots \wedge \phi_n) \rightarrow \psi)$ is a tautology, and suppose $\Gamma \not\models \psi$. Then there exists a truth assignment τ such that

$$\mathcal{T}[\phi_1]_{\tau} = \dots = \mathcal{T}[\phi_n]_{\tau} = 1 \text{ and } \mathcal{T}[\psi]_{\tau} = 0$$

From the previous proof, we obtain

$$\mathcal{T}[(\dots((\phi_1 \wedge \phi_2) \wedge \phi_3) \wedge \dots \wedge \phi_n) \rightarrow \psi]_{\tau} = 0$$

from which it follows that $((\dots((\phi_1 \wedge \phi_2) \wedge \phi_3) \wedge \dots \wedge \phi_n) \rightarrow \psi)$ is not a tautology, contradicting our assumption. ■

The following results may also be proved using the semantics of propositional logic.

So, the proof for this as I said will have to be done through the semantics of by using the semantics of propositional logic and please go through this proof, I will not go through it, now then there are other theorems.

(Refer Slide Time: 05:32)

Other Theorems

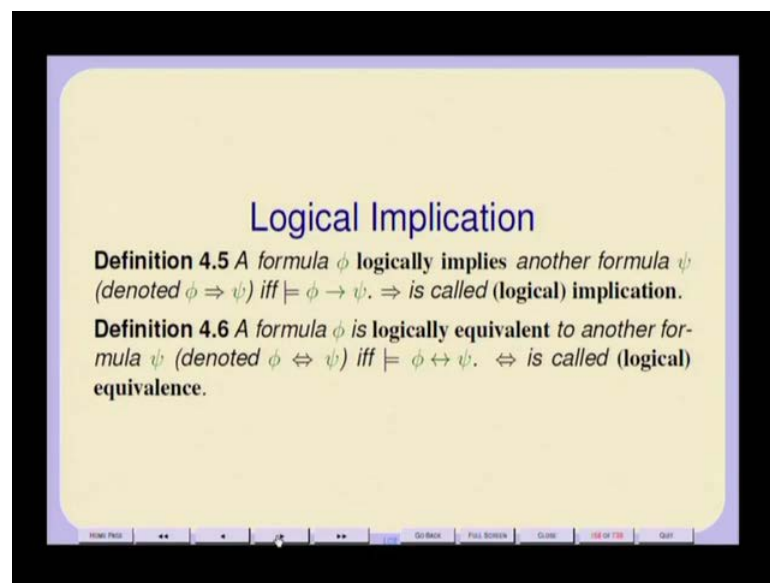
Theorem 4.3 Let $\Gamma = \{\phi_i \mid 1 \leq i \leq n\}$ be a finite set of propositions, and let ψ be any proposition. Then

1. $\Gamma \models \psi$ if and only if $\models \phi_1 \rightarrow (\phi_2 \rightarrow (\dots (\phi_n \rightarrow \psi) \dots))$
2. $\Gamma \models \psi$ if and only if $((\dots((\phi_1 \wedge \phi_2) \wedge \phi_3) \wedge \dots \wedge \phi_n) \wedge \neg\psi)$ is a contradiction.

Corollary 4.4 A formula ϕ is a tautology iff $\neg\phi$ is a contradiction (unsatisfiable). ■

For example, here if again Γ is a finite set and then ψ is a logical consequence of Γ if and only if this entire sequence of conditionals is a valid formulae, which is the same as saying that this huge formula should be a tautology. Similarly, here is the next one, where we construct this entire formula and negate the conclusion and put a conjunction of all of that. That should be a contradiction, this entire formula should be a contradiction, and then we would say that ψ is a logical consequence of Γ . So, one thing is of course a formula is a tautology if and only if the negation of phi is a contradiction.

(Refer Slide Time: 06:34)



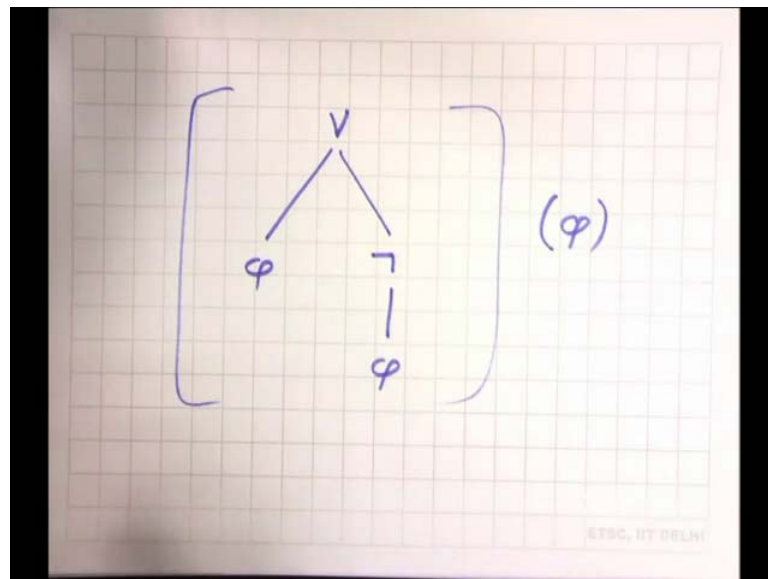
Then, we had the notion of logical implication, now just like logical consequence logical implication is also a concept of the theory of logic rather than something within logic itself. So, it is not within the language of logic it only relates two formulae, so we would say that phi logically implies xi if and only if phi arrowed xi. Now, this is a single formula, these are two formulas related by this relation, where this is a single formula in the language phi arrowed xi and we are saying that should be logically valid. In other words means that it has to be a tautology, so logical implication also can be brought down to the conditional.

Similarly, there is logical equivalent, normally the word the term tautology is employed only for those forms which come from propositional tautology in any higher order in any other logic like first order logic. So, we are talking about propositional forms, so as I said

p or $\neg p$ is the tautology in fact ϕ or $\neg \phi$ is also a tautology for any formula ϕ even in any first order logic or any other logic. This is the form that you have and a positive and its negation on the two sides of or that form.

That shape of formula is a tautological form, so the ϕ in those logics can be some other formula within that logic, but any logic in which propositional logic is some sub set all the tautological forms will be called tautologies. In addition, there could have logically valid formulas which do not satisfy propositional tautological forms. So, that is where the difference is going to come, you can think of it as a structure, in which you can plug in things.

(Refer Slide Time: 09:14)



So, the ϕ or $\neg \phi$ you can think of it as you can think of it as this structure and in any logic and you can think of this structure as a tautological form. If it takes in some parameter in any logic, it takes in only a single parameter not 2, not 2 or more parameters, but think of it as a single parameter form in which you can put copies of this anywhere. Then, this is always going to be a tautology, it does not matter whether you whether it is propositional logic or first order logic or some model logics or any other kinds of logics in which extension of propositional logic.

On the other hand, when you other operators in other logics they might have logically valid forms which do not have this propositional structure, I mean which are not entirely up of propositional operators. For example, they could, but they are not called

tautologies, there only called valid forms, not necessarily the notion of reduction is also something that needs to be formulized. So, we cannot we cannot use that terms blindly if what you are saying is that all tautological forms, all tautologies are logically equivalent to each other.

That is fine because you are saying that that is semantic statement reduction has to do is a proof theoretic statement. It is not a semantic statement, so this is what we had was logical implication and its and we had you have logical equivalence also, which we did and last time I think we made sure that, logically equivalence is the carnal of logical implication.

(Refer Slide Time: 11:13)

The slide is titled "Implication & Equivalence" in a blue font. Below the title, it says "Fact 4.7" in bold. There are four numbered points:

1. $\phi \Rightarrow \psi$ iff $\{\phi\} \models \psi$.
2. $\phi \Leftrightarrow \psi$ iff $\phi \Rightarrow \psi$ and $\psi \Rightarrow \phi$.
3. \Rightarrow is a preordering (reflexive and transitive) relation on \mathcal{P}_0 .
4. \Leftrightarrow is the kernel of \Rightarrow i.e. $\Leftrightarrow = \Rightarrow \cap \Rightarrow^{-1}$ and is hence indeed an equivalence relation on \mathcal{P}_0 .

At the bottom of the slide, there is a navigation bar with buttons for "Home Page", "Go Back", "Print Screen", "Close", "18 of 18", and "Quit".

So, you take the intersection of implication and its converse and that common set is the set of all is the equivalence relation on rho naught on p naught. The other thing we show we did last time was that we showed that logical equivalence is the congruence which boils down to in any algebraic system to show some congruence. It boils down to showing that equals those the congruent elements are replaced, replaceable by congruent elements in any context and usually that boils down.

(Refer Slide Time: 12:00)

The slide is titled "Logical Equivalence as a Congruence". It contains the following text:

Theorem 4.8 Logical equivalence is a congruence relation on \mathcal{P}_0 i.e. if $\phi \Leftrightarrow \psi$ then

- $\neg\phi \Leftrightarrow \neg\psi$ and
- for each $*$ $\in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$ and every formula χ we have

$$\phi * \chi \Leftrightarrow \psi * \chi$$
$$\chi * \phi \Leftrightarrow \chi * \psi$$

The slide also features a navigation bar at the bottom with buttons for "Go Back", "Full Screen", "Close", "14 of 78", and "Quit".

These facts that the particular case of this language, means that it is showing that it is preserved under negations. So, we would say that relation is being preserved under negation and similarly, we will say that this equivalence relation has to be preserved under these operations. So, these are all up binary operations, so what it means is that you take any formula, what you are saying is that and you take any of these binary operators. So, I am using star to denote any of them and you are saying phi star should be logically equivalent to size, if i is logically equivalent to xi and similarly this is like taking a frame work and plugging replacing phi which is there in it.

If it is not in any way perturbed by that, then you would say that is a congruence relation, so most of our theories of numbers. For example, real numbers the notion of equality is a congruence relation, so in fact all your addition, subtraction, multiplication, they are all they all preserve equality on numbers in theory of sets. For example, set equality is also a congruence relation, on the other hand in the in the theory of finite sets, I could define an equivalence relation, which says that a set a is equivalent to set b. They both have the number of elements, they need not be the same elements, and they have the same number of elements.

So, finite sets, then this equivalence relation is not a congruence because I can take two sets a and a prime, which are equivalent in terms of cardinality, but a union b for arbitrary be may not be same as a prime union b. For example, that equivalence relation

is not a congruence relation, so you have to for any equivalence relation is necessary to determine whether it is preserved under the operators. That are of interest to you, so logical equivalence is the congruence and that brings us to essentially defining these identities.

(Refer Slide Time: 14:28)

Logical Equivalence as a Congruence

Theorem 4.8 Logical equivalence is a congruence relation on \mathcal{P}_0 i.e. if $\phi \Leftrightarrow \psi$ then

- $\neg\phi \Leftrightarrow \neg\psi$ and
- for each $* \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$ and every formula χ we have

$$\phi * \chi \Leftrightarrow \psi * \chi$$

$$\chi * \phi \Leftrightarrow \chi * \psi$$

So, for example, these identities actually most of them directly come from the equilateral relation on Boolean algebras.

(Refer Slide Time: 14:34)

Some identities

$\phi \vee \perp \Leftrightarrow \phi$	Negation $\neg\neg\phi \Leftrightarrow \phi$	Identity $\phi \wedge \top \Leftrightarrow \phi$
$\phi \vee \top \Leftrightarrow \top$	Identity $\phi \wedge \top \Leftrightarrow \phi$	Zero $\phi \wedge \perp \Leftrightarrow \perp$
$\phi \vee \phi \Leftrightarrow \phi$	Idempotence $\phi \wedge \phi \Leftrightarrow \phi$	Idempotence $\phi \wedge \phi \Leftrightarrow \phi$
$\phi \vee \psi \Leftrightarrow \psi \vee \phi$	Commutativity $\phi \wedge \psi \Leftrightarrow \psi \wedge \phi$	Commutativity $\phi \wedge \psi \Leftrightarrow \psi \wedge \phi$
$(\phi \vee \psi) \vee \chi \Leftrightarrow \phi \vee (\psi \vee \chi)$	Associativity $(\phi \wedge \psi) \wedge \chi \Leftrightarrow \phi \wedge (\psi \wedge \chi)$	Associativity $(\phi \wedge \psi) \wedge \chi \Leftrightarrow \phi \wedge (\psi \wedge \chi)$
$\phi \vee (\psi \wedge \chi) \Leftrightarrow (\phi \vee \psi) \wedge (\phi \vee \chi)$	Distributivity $\phi \wedge (\psi \vee \chi) \Leftrightarrow (\phi \wedge \psi) \vee (\phi \wedge \chi)$	Distributivity $\phi \wedge (\psi \vee \chi) \Leftrightarrow (\phi \wedge \psi) \vee (\phi \wedge \chi)$
$\neg(\phi \vee \psi) \Leftrightarrow \neg\phi \wedge \neg\psi$	De Morgan $\neg(\phi \wedge \psi) \Leftrightarrow \neg\phi \vee \neg\psi$	De Morgan $\neg(\phi \wedge \psi) \Leftrightarrow \neg\phi \vee \neg\psi$
$\phi \vee \neg\phi \Leftrightarrow \top$	Simplification $\phi \wedge \neg\phi \Leftrightarrow \perp$	Simplification $\phi \wedge \neg\phi \Leftrightarrow \perp$
$\neg\perp \Leftrightarrow \top$	Inversion $\neg\top \Leftrightarrow \perp$	Inversion $\neg\top \Leftrightarrow \perp$
$\phi \vee (\phi \wedge \psi) \Leftrightarrow \phi$	Absorption $\phi \wedge (\phi \vee \psi) \Leftrightarrow \phi$	Absorption $\phi \wedge (\phi \vee \psi) \Leftrightarrow \phi$

So, what the equilateral relation on Boolean algebra is the typical example of congruence and these identities are essentially linguistic versions of those Boolean identities. So, you can see that, so for example for any five and bottom five or bottom is going to be equal to 5. Then, there are the usual things distributive laws or distribute, so where and distribute, so what or there are de Morgan's laws not of or is the not gets distributed or gets replacement. Similarly, then there are these things like simplification and inversion and there is this negation law, for example double negation cancels both of them.

So, what this means now is that in any logical context if you see a sub term of that logical context, which can be replaced by another logical equivalent formula. Then, you can replace it and that is what we have been using throughout whatever we did in Boolean algebra in hardware circuits. So, most of the things that we did except for you know in the case of hardware, you have this carload map technique.

If you do not have, then do not care states then what you are doing is you are preserving logical equivalence, but the introduction of do not care states creates a different complication. You are you are no longer preserving equivalence, you are preserving some kind of greater than or equal to relation, but you are not you are definitely naught preserving equivalence. So, it is important to keep that in mind now and when you just look at the operators of the logic.

(Refer Slide Time: 16:50)

Adequacy

Some Other Important identities are:

$$\phi \leftrightarrow \psi \Leftrightarrow (\phi \rightarrow \psi) \wedge (\psi \rightarrow \phi) \quad (5)$$

$$\phi \rightarrow \psi \Leftrightarrow \neg\phi \vee \psi \quad (6)$$

Definition 5.1 A set of operators $O \subseteq \Omega$ is said to be **adequate** for propositional logic, if for every formula in \mathcal{P}_0 there is a logically equivalent formula using only the operators in O .

Let us look there are some other important identities, for example this bi conditional $\phi \leftrightarrow \psi$ is logically equivalent to $(\phi \rightarrow \psi) \wedge (\psi \rightarrow \phi)$. This is something that you can easily prove and $\phi \rightarrow \psi$ is logically equivalent to $\phi \wedge \neg \psi$ this is also something that most of you are familiar with. Now, what we are saying is that in a certain sense these two logical constructs are redundant in a certain sense and I would so given this Ω is all those operators that by which we originally define propositional logic top bottom not and or arrow.

By condition, all those operators, so take any set of operators which is the sub set of this Ω that is set to be adequate for propositional logic. If every formula in \mathcal{P} is true, there is a logically equivalent formula using only the operators in \mathcal{O} . So, in these two logical equivalences, essentially say that if from Ω I remove these two operators the by conditional and conditional, then what I have left is still an adequate set for propositional logic. It is not necessary for me to have kept on adding more and more operators in order to get whatever is the power of those six, those two constants and those five operators. So, that is like seven operators those two constants can be regarded as two 0 operators.

So, instead of the 7, the 7 the power of the 7 operators is the same as the power of the set, without these two operators by conditional and the conditional, this is there. Another concept that I am going to talk about and in a certain sense most books on logic do not distinguish between adequacy and that concept which I call functional completeness, but I am going to distinguish between them. So, this is adequacy, so all I am saying is my propositional logic consist of the two constraints, bottom and top, a unary operator negation and the four binary operators and or conditional and by conditional.

When I look at adequacy of a set of operators, I am looking at it from the point of view of the set. So, it is always related to this set Ω and I am saying that, so set of operators is adequate for propositional logic. If every statement in propositional logic, which could be written using all the operators can also be written in a logically equivalent form using this subset of operators.

(Refer Slide Time: 20:18)

Adequacy: Examples

Example 5.2

- From the identities (5) and (6) and the two **Simplification** identities it is clear that $O = \{\neg, \wedge, \vee\}$ is an adequate set of operators for \mathcal{P}_0 .
- Further given that $O = \{\neg, \wedge, \vee\}$ is adequate and using the **De Morgan** identity and **Negation**, we have that

$$\phi \wedge \psi \Leftrightarrow \neg(\phi \wedge \psi) \Leftrightarrow \neg(\neg\phi \vee \neg\psi)$$
 and hence $\{\neg, \vee\}$ is an adequate set.
- We may use the other **De Morgan** identity

$$\phi \vee \psi \Leftrightarrow \neg(\phi \vee \psi) \Leftrightarrow \neg(\neg\phi \wedge \neg\psi)$$
 to conclude that $\{\neg, \wedge\}$ is adequate.

So, from these identities five and six, what we essentially and a two simplification, we get that this set o consisting of three operators is an adequate set of operators. By the way, it is possible to also get rid of this bottom and top, we do not need them because of the fact that I can choose because of the fact that instead of top.

(Refer Slide Time: 20:50)

Some identities

$\phi \vee \perp \Leftrightarrow \phi$	$\phi \wedge \top \Leftrightarrow \phi$	Negation $\neg\neg\phi \Leftrightarrow \phi$	
$\phi \vee \top \Leftrightarrow \top$	Identity $\phi \wedge \top \Leftrightarrow \phi$	Zero $\phi \wedge \perp \Leftrightarrow \perp$	
$\phi \vee \phi \Leftrightarrow \phi$	Zero $\phi \wedge \perp \Leftrightarrow \perp$	Idempotence $\phi \wedge \phi \Leftrightarrow \phi$	
$\phi \vee \psi \Leftrightarrow \psi \vee \phi$	Idempotence $\phi \wedge \phi \Leftrightarrow \phi$	Commutativity $\phi \wedge \psi \Leftrightarrow \psi \wedge \phi$	
$(\phi \vee \psi) \vee \chi \Leftrightarrow \phi \vee (\psi \vee \chi)$	Commutativity $\phi \wedge \psi \Leftrightarrow \psi \wedge \phi$	Associativity $(\phi \wedge \psi) \wedge \chi \Leftrightarrow \phi \wedge (\psi \wedge \chi)$	
$\phi \vee (\psi \wedge \chi) \Leftrightarrow (\phi \vee \psi) \wedge (\phi \vee \chi)$	Associativity $(\phi \wedge \psi) \wedge \chi \Leftrightarrow \phi \wedge (\psi \wedge \chi)$	Distributivity $\phi \wedge (\psi \vee \chi) \Leftrightarrow (\phi \wedge \psi) \vee (\phi \wedge \chi)$	
$\neg(\phi \vee \psi) \Leftrightarrow \neg\phi \wedge \neg\psi$	Distributivity $\phi \wedge (\psi \vee \chi) \Leftrightarrow (\phi \wedge \psi) \vee (\phi \wedge \chi)$	De Morgan $\neg(\phi \wedge \psi) \Leftrightarrow \neg\phi \vee \neg\psi$	
$\phi \vee \neg\phi \Leftrightarrow \top$	De Morgan $\neg(\phi \wedge \psi) \Leftrightarrow \neg\phi \vee \neg\psi$	Simplification $\phi \wedge \neg\phi \Leftrightarrow \perp$	
$\neg\perp \Leftrightarrow \top$	Simplification $\phi \wedge \neg\phi \Leftrightarrow \perp$	Inversion $\neg\top \Leftrightarrow \perp$	
$\phi \vee (\phi \wedge \psi) \Leftrightarrow \phi$	Inversion $\neg\top \Leftrightarrow \perp$	Absorption $\phi \wedge (\phi \vee \psi) \Leftrightarrow \phi$	

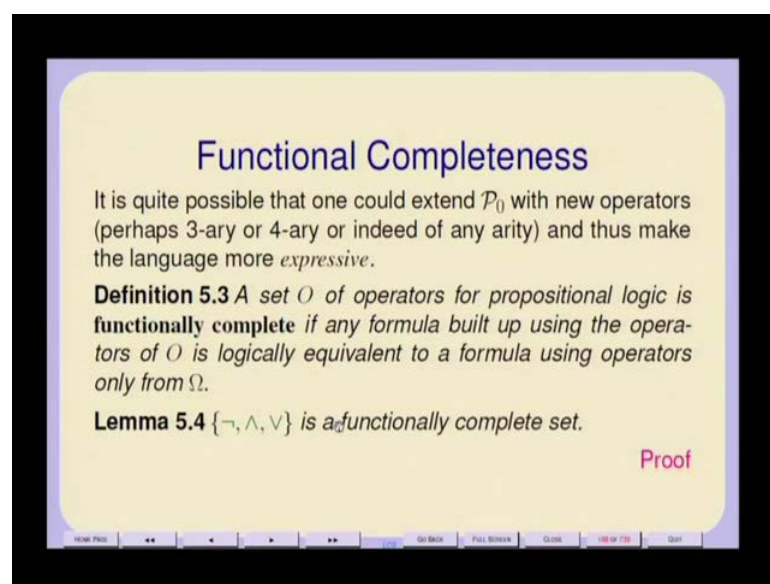
I can choose any atomic proposition and top as p, let us say atomic proposition p and right top as p or naught p and similarly, I can write bottom as p and naught p. For example, even this bottom and top are actually not required, so this means that with this

three operators, I have an adequate set, which captures the expressive power of all of propositional logic up to logical equivalence. Further of course, which we have seen in hardware courses probably is that you can get rid of one of these two. The de Morgan's identity essentially says that can be replaced by not and or right and vice versa or can be replaced.

Therefore, you just require this require negation and one of this one or to get an adequate set and in fact what you have learnt in your hardware course is that it is not even necessary to have this it is necessary to just have a single operator. Let us say that is adequate for all propositional logic, similarly you can just use an, but let us leave with this, since we are interested more in reasoning than in getting minimal hardware circuits.

We will keep at least these three operators, usually this three operators, there is another there is another concept which I am going to define and that is called functional completeness. So, you just take this think of it this way, you take supposing I define an arbitrary ternary operator on Boolean algebra or on propositional logic and then I want to know whether that ternary operator is somehow primitive. So, primitive in the sense that is it possible to express things with which it is not possible to express with the other operators that that I already have, for example this question for some reason most of is never asked.

(Refer Slide Time: 24:26)



Functional Completeness

It is quite possible that one could extend \mathcal{P}_0 with new operators (perhaps 3-ary or 4-ary or indeed of any arity) and thus make the language more *expressive*.

Definition 5.3 A set O of operators for propositional logic is **functionally complete** if any formula built up using the operators of O is logically equivalent to a formula using operators only from Ω .

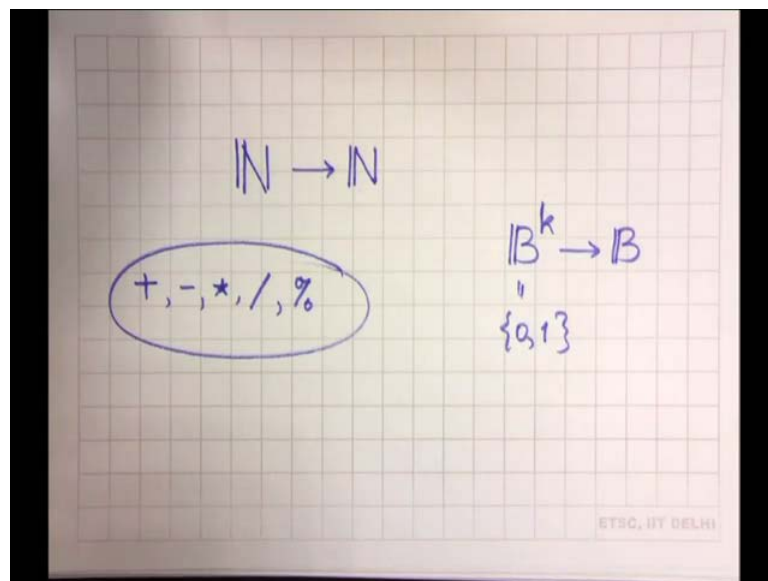
Lemma 5.4 $\{\neg, \wedge, \vee\}$ is a functionally complete set.

Proof

Navigation icons: Home, First, Previous, Next, Last, Search, Refresh, Close, 18 of 19, Quit

If you take the naturals or take the naturals the natural numbers, we usually define these operations like addition subtraction multiplication coefficient remainder and essentially we have done all our mathematics with that. However, we are aware that there are functions from the naturals or naturals raise to some power k , there are k functions from n raise to k to k . This may not be expressible in terms of this four or five operator operators that we have got, I mean if there is the class of functions.

(Refer Slide Time: 24:41)



Even the class of unary functions from n to n is unaccountably large, we have just limited ourselves to basically these binary operators let us say, but that is not mean that there are not functions on the naturals. It does not mean that every function on the naturals can be expressed up to equality only in terms of this; on the other hand Boolean algebra is sort of specially constituted.

That special constitution is that the operators that we have got form of functionally complete set; you take any arbitrary Boolean operator you define your own Boolean operator. Let us say you define a Kerry Boolean operator from b raise to k to b , where b is the set $0, 1$ whatever may be the nature of that operator.

It can still be written entirely in terms of the of an of a of the of the set of operators that we already have and that is the property of functional completeness. It is not true for n , it is not true for r , for example in u , I am sorry \min and \max may not all be completely defined, \min is defined may be, but \max is not defined always. For example, \min will not

be definable in terms of these five operators any one, so there are and you conceive of other operators which cannot be defined, but in the case of the in case of the Boolean logic Boolean algebra. Therefore, propositional logic also every adequate set is also functionally complete and the proof for that, actually once you see it, it is actually quite easy what are we saying you take, so what we are saying is this.

(Refer Slide Time: 27:16)

Proof of lemma 5.4.

Proof: By the semantics of propositional logic, every operator of propositional logic corresponds to an operator of the same arity on the boolean set $\{0, 1\}$. The proof follows from the construction of truth tables in the boolean algebra \mathcal{B} , since every truth table may be expressed using the boolean operators $\{\neg, \wedge, \vee\}$.

Let $\alpha_n : 2^n \rightarrow 2$ be any n -ary operator on boolean values. Typically there exists a truth table for the function $\alpha_n(a_1, \dots, a_n)$, which defines for each possible set of boolean values of the arguments the boolean value of $\alpha_n(a_1, \dots, a_n) = b$. This truth table consists of 2^n rows and $n + 1$ columns as shown below where $b_0, \dots, b_{2^n-1} \in 2$ (we have numbered the rows by the decimal equivalent of the binary number that the bit-vector (a_1, \dots, a_n) denotes in each case).

	a_1	\dots	a_n	b
0	0	\dots	0	b_0
\vdots	\vdots	\vdots	\vdots	\vdots
\vdots	\vdots	\vdots	\vdots	\vdots
\vdots	\vdots	\vdots	\vdots	\vdots
$2^n - 1$	1	\dots	1	b_{2^n-1}

This property would be true for the propositional logic if it is true for Boolean algebra, so essentially the problem our semantics of propositional logic maps, this not to inversion the Boolean inversion. The Boolean product and or to the Boolean sound right, so this will be functionally complete if and only if the corresponding operation on Boolean on Boolean algebra are functionally complete.

So, the problem reduces to proving that the corresponding operators in Boolean algebra are functionally complete, so let us assume that little α is some arbitrary unary operator on the set $0, 1$, so that is it takes its unary operator which gives you a Boolean result. It also gives you an element from the set its co domain is also 2 and set $0, 1$, so what we are saying is such an operator clearly has something like a truth table associated with it. So, if you think of this operator applied to n arguments a_1, \dots, a_n where each of a_1 to a_n takes values either 0 or 1 , then there are essentially 2^n different possible sequence vectors of values that can be assigned.

This operator should give you a result, since this is the total function, it should give you a 0 or 1 result for each of those 2^n possible vectors. So, that is this b lets say b_0 to $b_{2^n - 1}$, now that means in Boolean algebra partly because we are talking about a 2 element set the finiteness of the set is important. In fact one of the reasons why n raised to n is you cannot get finite set of functionally complete. You cannot get a finite functionally complete set of operators for n to n is because n is infinite and the number of functions from n to n is unaccountably infinite.

Here, we are dealing with everything that is finite, so that means there is a table which should tell you for each vector, what the result should be right and that vector is a sufficient and complete vector complete definition of this operator o on n . If there is such a vector, then what I have claimed is I claim that this truth table, this is just the truth table for that operator. I claim that this truth table can only be expressed in terms of just inversion dot and plus and how do I express it that way I just take, I think of this truth table it has 2^n rows.

I can think of this truth table as an big or of this 2^n rows of the result of this 2^n rows, each row I can think of it is a big and of the individual elements a_1 to a_n such that take any a_i a_j if it is 0 in that row make it a \bar{a}_j because \bar{a}_j will be 1. So, it is just a product of those, so I am calling it a \bar{a}_j star, the star indicates that if a_j in that row is 1, then you keep one if a_j in that row is 0, basically you are looking at a \bar{a}_j . So, I take this i take this \bar{a}_j stars this product of this \bar{a}_j stars that is one and i take this b_i for the i th row. Here, I decide whether I should give that value to be 1 or 0 based on the value of that b_i . So, I have this star at that top here, which will be determined whether it is a inversion or not depending on the value of that b at that row.

(Refer Slide Time: 31:32)

We may express the function as

$$o_n(a_1, \dots, a_n) = \sum_{0 \leq r \leq 2^n - 1} \left(\prod_{1 \leq j \leq n} a_j^{*r} \right) \Phi_r$$

where for each row r , and each j , $1 \leq j \leq n$, $a_j^{*r} = 1$ if $a_j = 1$ and $a_j^{*r} = 0$ otherwise. Similarly the product $(\prod_{1 \leq j \leq n} a_j^{*r})$ is inverted if $b_r = 0$ otherwise $(\prod_{1 \leq j \leq n} a_j^{*r})^{*r} = 1$. ■

So, I take the product of the individual components in each row and the sum over all the rows and now that is just some of products. So, every operator has been compliantly defined in a sum of products form. So, that shows, now reverting back that shows that you take any adequate set of operator of propositional logic that set is also functionally complete up to truth values.

(Refer Slide Time: 33:27)

Duality

Definition 5.5

- Two formulas ϕ and ψ are called **duals** of each other if each can be obtained from the other by simultaneously replacing all occurrences of
 - \wedge by \vee ,
 - \vee by \wedge ,
 - \perp by \top and
 - \top by \perp
- \wedge and \vee are duals of each other and
- \top and \perp are duals of each other

The other thing the other important concept that we should look at is that we will think of the principal duality. One of the nice things about your Boolean algebra is really that and

also therefore, propositional logic is that except for these notions like validity and so on. So, for true and false, essentially from have other it is possible at least in Boolean algebra. It is possible to think of 0 and 1 as having equal stratus, so it is possible to think of every think of 0 and 1 has been duals of each other. So, we will say that two formulas phi and xi, so we are looking at all this in the in the context not a logical validity, but in the context of logical equivalence which is purely algebraic concept.

So, we are looking at the system of propositional logic under logical equivalence as a simple algebraic system. So, then truth and falsehood are duals of each other and things can be inverted, so the notion of dual is just this two formulas phi and xi are called duals of each other. If each formula can be replaced, can be obtained from the other by simultaneous replacement of this, so you replace all AND s by OR and all OR s by AND s, all bottoms by tops and all tops by bottoms.

So, you take the formula phi and you get it phi dual which the tree structure of phi and phi dual. In fact, even this sentential structure is otherwise the same except that all the operators have been replaced by their duals. Now, essentially we this and suppose to duals of each other bottoms and tops are also duals of each other right, but now what happens is that the duality the two the two formulas are phi and its dual are not equal. They do not even necessarily logically imply each other, but what does happen is that if I take the inversions, so if I take the negations of the atoms in the dual.

(Refer Slide Time: 36:13)

Principle of Duality

Theorem 5.6 If $\text{atoms}(\phi) = \{p_1, \dots, p_n\}$ and $\phi \equiv o(p_1, \dots, p_n)$ then

$$\neg\phi \Leftrightarrow o^*(\neg p_1, \dots, \neg p_n)$$

where o^* is the dual of o .

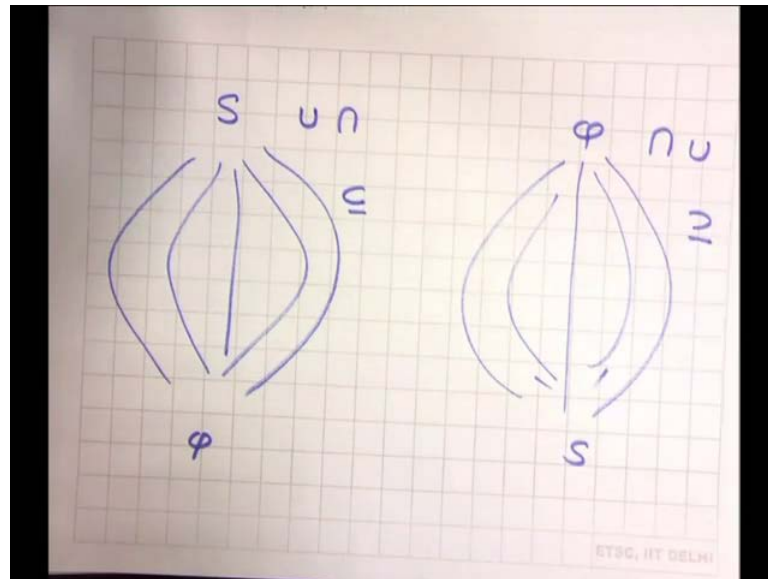
Proof: By structural induction and the use of the De Morgan and Simplification laws

□

Navigation bar: Home, Prev, Next, Go Back, Full Screen, Close, 1/1 of 1/1, Quit

Then, I would get the negation of the original formula is that, so the duality comes through negation its sort of mirror reflection that negation does which gives you the property of duality. It is possible to talk about duality, even when there is no negation, for example you take any set take the power set of that set under the subset ordering the structure.

(Refer Slide Time: 37:02)

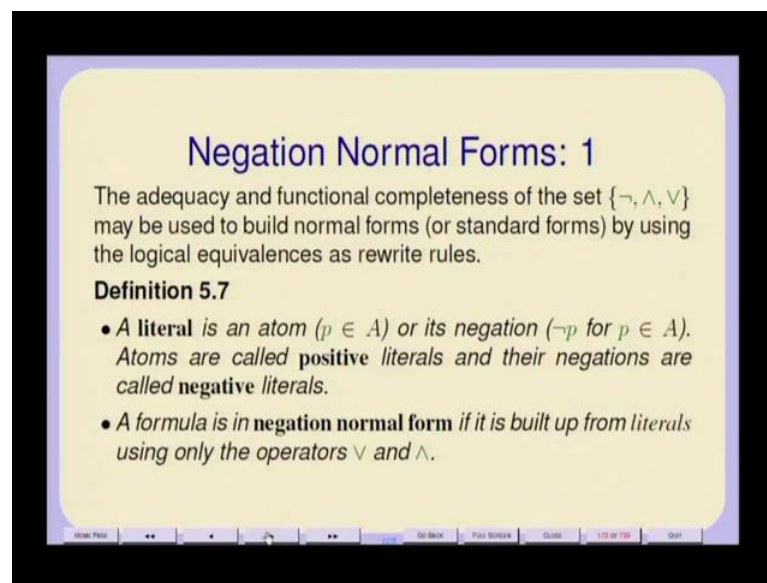


If you want to draw under the subset ordering you will get, so you take a set s and you have the empty set and you have basically various kinds of subsets in between this structure can be inverted. So, this is let us say under the subset ordering, it can be inverted and essentially under the superset ordering I will get in almost symmetrical inversion which looks the same. So, the duality in this case is that every union will be replaced by an intersection, here subset will be replaced by super set and every intersection here will be replaced by union. I do not need to necessarily consider inversions at all, for example I do not need to think of complement complements at all.

Even with just union and intersection, I get a duality, so if a set is defined as a union of some other sets, I can think of another set defined as a intersection, but there is an inversion which takes place. So, it is possible to define duality here, but under inversion the duality becomes absolutely precise and exact and the equality becomes notable. So, duality does hold for what I know as this is actually an example of complete lattice, it does hold for complete, lattice is to once you put in inversions.

This becomes an Boolean algebra, so inversions come in naturally, but the point is otherwise the principle of duality does hold for complete lattices which are not Boolean algebra. There is there is certain symmetry in the structure which can be inverted and that is what mean by duality. So, this principle of duality for proposition of logic is essentially by you can prove it by structural induction and the use of de Morgan's and implication laws am I am not going to do that once you have logical equivalence.

(Refer Slide Time: 40:04)



Negation Normal Forms: 1

The adequacy and functional completeness of the set $\{\neg, \wedge, \vee\}$ may be used to build normal forms (or standard forms) by using the logical equivalences as rewrite rules.

Definition 5.7

- A **literal** is an atom ($p \in A$) or its negation ($\neg p$ for $p \in A$). Atoms are called **positive literals** and their negations are called **negative literals**.
- A formula is in **negation normal form** if it is built up from literals using only the operators \vee and \wedge .

Once you have functional completeness and adequacy, you know that logical equivalence is a congruence relation. You are essentially ready to do things like what you did in hardware there, what you did, you actually try to use car, no maps. For example, to simplify circuits to get the least numb of elements, for example in order to perform a certain function, then we are going to work the same way, but towards some system of reasoning. So, the first thing that we will look at is the notion of a normal form, normal forms also exist, also existed and in the case of your hard ware circuits, your sum of products forms and product of sums forms were normal forms.

So, let us look at some kinds of normal forms, so I will take, so I will look at my atoms and negations of atoms. So, I will atoms positive atoms and there negation, I will call them negative, I will call them, so I will call the atoms positive literals and their negation negative literals. Now, our formulae is essentially a sentence in propositional logic is in negation normal form, if it is built up from literals using only the operators OR an AND.

So, this is why is it important to look at normal forms because then what you are looking at is not the entire language, but an equivalent subset of the original language. So, this way for example, you take a formulae in negation normal form, you are actually removing all those sentences, where negation can appear outside and or and so on, you are constraining the language to a smaller set.

(Refer Slide Time: 42:10)

Negation Normal Forms: 2

Lemma 5.8 Every formula in \mathcal{P}_0 is logically equivalent to one in negation normal form.

Proof: It suffices to consider only formulas containing the operators \neg , \wedge and \vee , and for every occurrence of negation to push it inward using the **De Morgan** identities. ■

It is clear that every formulae in propositional logic is logically equivalent to a formulae in negation normal form, what do I do? I just use de Morgan's identity to push the negation inside and bring them down to the leaves, there is one thing more, also I need to use that, I need to use that negation law this one.

(Refer Slide Time: 42:22)

Some identities		
$\phi \vee \perp \Leftrightarrow \phi$	$\Leftrightarrow \phi$	
$\phi \vee \top \Leftrightarrow \top$	$\Leftrightarrow \top$	
$\phi \vee \phi \Leftrightarrow \phi$	$\Leftrightarrow \phi$	
$\phi \vee \psi \Leftrightarrow \psi \vee \phi$	$\Leftrightarrow \psi \vee \phi$	
$(\phi \vee \psi) \vee \chi \Leftrightarrow \phi \vee (\psi \vee \chi)$	$\Leftrightarrow \phi \vee (\psi \vee \chi)$	
$\phi \vee (\psi \wedge \chi) \Leftrightarrow (\phi \vee \psi) \wedge (\phi \vee \chi)$	$\Leftrightarrow (\phi \vee \psi) \wedge (\phi \vee \chi)$	
$\neg(\phi \vee \psi) \Leftrightarrow \neg\phi \wedge \neg\psi$	$\Leftrightarrow \neg\phi \wedge \neg\psi$	
$\phi \vee \neg\phi \Leftrightarrow \top$	$\Leftrightarrow \top$	
$\neg\perp \Leftrightarrow \top$	$\Leftrightarrow \top$	
$\phi \vee (\phi \wedge \psi) \Leftrightarrow \phi$	$\Leftrightarrow \phi$	
		Negation $\neg\neg\phi \Leftrightarrow \phi$
		Identity $\phi \wedge \top \Leftrightarrow \phi$
		Zero $\phi \wedge \perp \Leftrightarrow \perp$
		Idempotence $\phi \wedge \phi \Leftrightarrow \phi$
		Commutativity $\phi \wedge \psi \Leftrightarrow \psi \wedge \phi$
		Associativity $(\phi \wedge \psi) \wedge \chi \Leftrightarrow \phi \wedge (\psi \wedge \chi)$
		Distributivity $\phi \wedge (\psi \vee \chi) \Leftrightarrow (\phi \wedge \psi) \vee (\phi \wedge \chi)$
		De Morgan $\neg(\phi \wedge \psi) \Leftrightarrow \neg\phi \vee \neg\psi$
		Simplification $\phi \wedge \neg\phi \Leftrightarrow \perp$
		Inversion $\neg\top \Leftrightarrow \perp$
		Absorption $\phi \wedge (\phi \vee \psi) \Leftrightarrow \phi$

I will require using this sometimes to cancel out the negations though I have not mentioned it here, so I will require that, but sorry essentially using de Morgan's identities, we can and we can take any. So, one thing is know that NOT and AND are an adequate and functionally complete set, so we just consider formulas only using NOT and AND, OR and where ever not occurs outside somewhere.

We push it and use in de Morgan's laws and if double negations occur, we cancel them out and this process will ensure that every formula is logically equivalent to 1 in negation normal form. This means that it consists of the negation occurs only in front of an atom, it does not occur outside, so it consists of positive and negative laterals. It is made up of positive and negative literals using only the operators and AND, OR that is the next thing we can do.

(Refer Slide Time: 43:38)

Conjunctive Normal Forms

Definition 5.9

- A disjunction of literals is a formula δ of the form
$$\delta \equiv L_1 \vee L_2 \vee \cdots \vee L_m \equiv \bigvee_{1 \leq i \leq m} L_i$$

where $m \geq 0$.

- A conjunctive normal form is a formula γ of the form $\delta_1 \wedge \delta_2 \wedge \cdots \wedge \delta_n$ where δ_j for each $1 \leq j \leq n$ is a disjunction of literals.

We may analogously define a conjunction of literals and a disjunctive normal form (DNF).

This is equivalent to your product of sums formed, which I am going to call conjunctive normal form, conjunction and disjunction. So, I can take a disjunction literals, so I take literals and I take an and of those disjoints and what do I get, I get a conjunctive normal form, which is essentially what have learnt in hardware. It will come out as a product of sums formed, so analogously we can define a conjunction of laterals and disjunction of conjuncts. That would be what is known as a disjunctive normal form, which in your hardware circuit, you came across as sum of products form, but we will call them CNF and DNF conjunctive normal form and disjunctive normal form.

(Refer Slide Time: 44:45)

CNF

Theorem 5.10 Every formula in \mathcal{P}_0 is logically equivalent to a conjunctive normal form.

Proof: It suffices to consider only negation normal forms. In each case use the distributive laws to distribute \vee over \wedge . ■

So, the next theorem is again very easy every formula in p naught is logically equivalent to a form to a conjunctive normal form. Now, it is enough to assume that the formulae is in negation normal form, so there are any way no negation signs all the negation signs when they appear. They appear only in front of the atoms, there are no double negations and the rest of the formulae the formula is just built up using AND s and OR s, all you need to do is to distribute the OR s over the AND s in order to get a conjunctive normal form. Similarly, if you distribute the AND s over the OR s, you get a formulae in disjunctive normal form, but there is a price to be paid and what is the price to be paid?

Ultimately, we have to look at all these things as somehow being part of some implementation in some theorem proving we have to design algorithms for them. So, what is the price that you pay here, I mean it is not just pure mathematical theory, now suppose you have to think of it in terms of what is the price that you are paying think of algorithms and so on. So, the distributive laws create copies, so take this phi and xi or xi phi and xi or phi and xi.

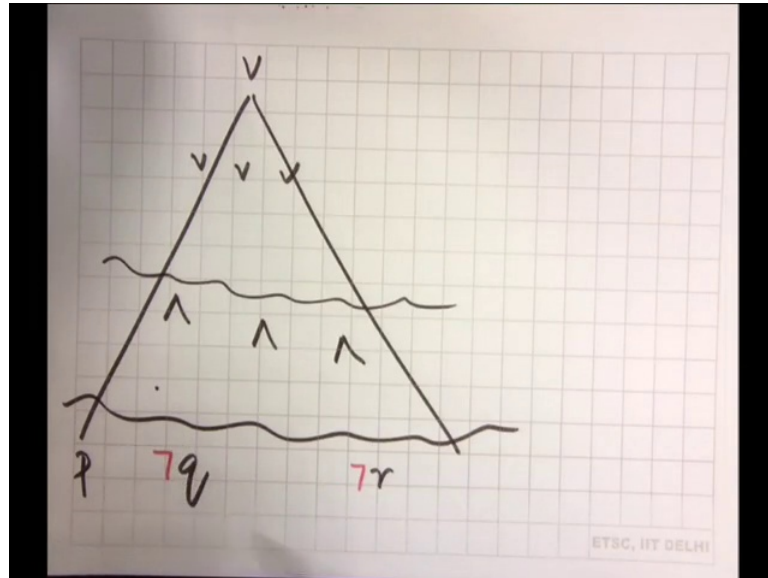
(Refer Slide Time: 46:32)

Some identities			
$\phi \vee \perp$	$\Leftrightarrow \phi$	Negation	$\neg\neg\phi \Leftrightarrow \phi$
$\phi \vee T$	$\Leftrightarrow T$	Identity	$\phi \wedge T \Leftrightarrow \phi$
$\phi \vee \phi$	$\Leftrightarrow \phi$	Zero	$\phi \wedge \perp \Leftrightarrow \perp$
$\phi \vee \psi$	$\Leftrightarrow \psi \vee \phi$	Idempotence	$\phi \wedge \phi \Leftrightarrow \phi$
$(\phi \vee \psi) \vee \chi$	$\Leftrightarrow \phi \vee (\psi \vee \chi)$	Commutativity	$\phi \wedge \psi \Leftrightarrow \psi \wedge \phi$
$\phi \vee (\psi \wedge \chi)$	$\Leftrightarrow (\phi \vee \psi) \wedge (\phi \vee \chi)$	Associativity	$(\phi \wedge \psi) \wedge \chi \Leftrightarrow \phi \wedge (\psi \wedge \chi)$
$\neg(\phi \vee \psi)$	$\Leftrightarrow \neg\phi \wedge \neg\psi$	Distributivity	$\phi \wedge (\psi \vee \chi) \Leftrightarrow (\phi \wedge \psi) \vee (\phi \wedge \chi)$
$\phi \vee \neg\phi$	$\Leftrightarrow T$	De Morgan	$\neg(\phi \wedge \psi) \Leftrightarrow \neg\phi \vee \neg\psi$
$\neg\perp$	$\Leftrightarrow T$	Simplification	$\phi \wedge \neg\phi \Leftrightarrow \perp$
$\phi \vee (\phi \wedge \psi)$	$\Leftrightarrow \phi$	Inversion	$\neg T \Leftrightarrow \perp$
		Absorption	$\phi \wedge (\phi \vee \psi) \Leftrightarrow \phi$

So, if you have a huge number of AND s and OR s and you are going to start distribution, then you are going to keep creating multiple copies. In the worst case, supposing you got your formulae in disjunctive normal form and you have to convert it in to conjunctive normal form by this method. In the worst case, you will create two copies of every sub formulae, which means you suddenly explored the formulae to an

exponential size by the time you have come down to the leaves. You think of it as a tree, you have a tree, you have a huge tree, let us say in which you had disjunctions all over the place up to some front here.

(Refer Slide Time: 47:27)



Then, you had conjunctions all over the place up to some front here and then of course you had may be some negations scattered, but otherwise everything else is an atom. Let us say now this is a disjunctive normal form, if you have to convert it to conjunctive normal form, you will be distributing the OR over the AND, which means you will be replicating nodes whole sub trees. So, you will get a greatly expanded sub tree, which in the worst case can be exponential in size to the original tree. So, other use of distributive laws is going to prove expensive in that sense, but theoretically of course you just use the distributive laws to convert from one to another, but there is a there is a price to be paid for it.

I may not be as bad as exponential because of the fact that it is not the entire tree that you have. It is not the every sub tree is going to be replicated, but in the worst case that is what that is the upper bound.

Usually, it will be much less, but that is that is one thing, so that is the price to be paid very often in in obtaining a conjunctive normal form, because in obtaining a conjunctive normal form whenever AND, OR occurs over a over AND, you will have to distribute it. So, you will be replicating some sub trees in that abstract syntax tree, so next time what

we will do is, we will look at how to use these CNF s for the purpose of some kind of verification validity proving some proving some arguments. So, we look at the design of a tautology checker and the we will look at more algorithms that will be our first with computer science, so I will stop here.