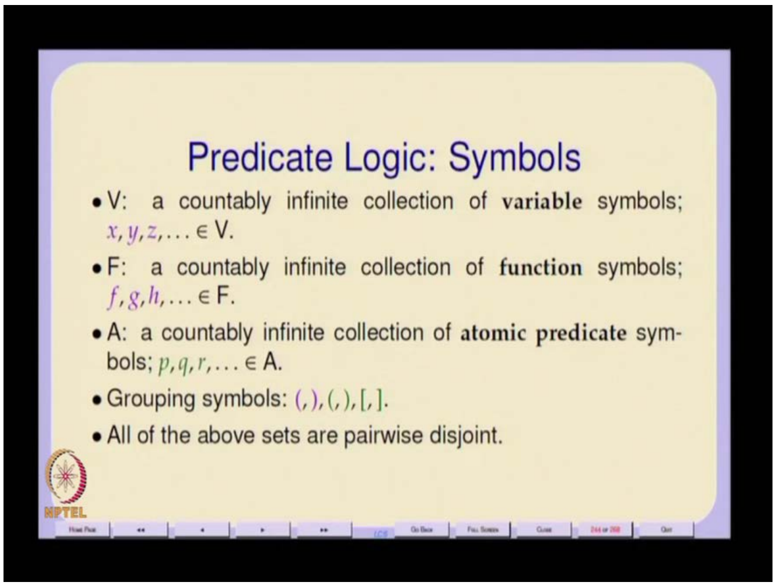
Logic for CS
Prof. Dr. S. Arun Kumar
Department of Computer Science
Indian Institute of Technology, Delhi

Lecture - 18
The Semantics of Predicate Logic

So, today we will start the semantics of predicate logic; before that let us just quickly go through the syntax of predicate logic.

(Refer Slide Time: 00:54)

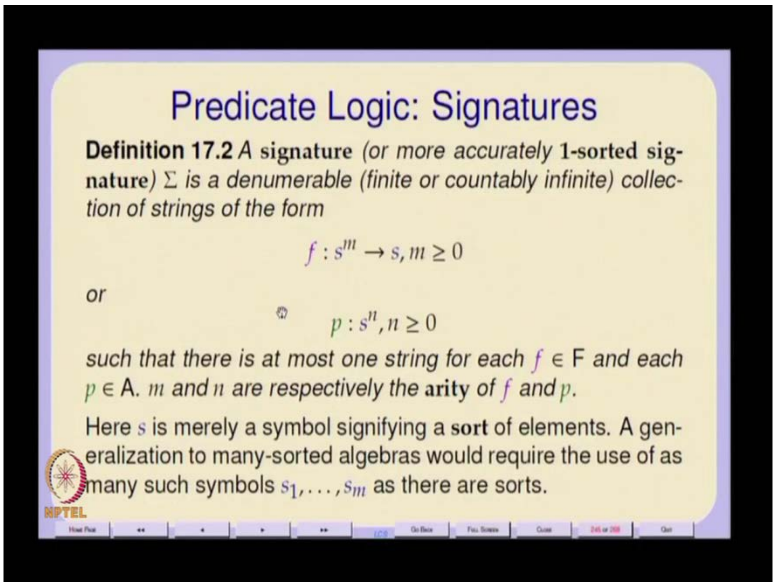


Predicate Logic: Symbols

- V: a countably infinite collection of **variable** symbols; $x, y, z, \ldots \in$ V.
- F: a countably infinite collection of **function** symbols; $f, g, h, \ldots \in$ F.
- A: a countably infinite collection of **atomic predicate** symbols; $p, q, r, \ldots \in$ A.
- Grouping symbols: (,), (,), [,].
- All of the above sets are pairwise disjoint.

So, we have a countably infinite collection of variable symbols and a set of terms and a countably infinite collection of atomic predicate symbols.

(Refer Slide Time: 01:03)



Predicate Logic: Signatures

Definition 17.2 *A* **signature** *(or more accurately* **1-sorted signature***)* Σ *is a denumerable (finite or countably infinite) collection of strings of the form*

$$f : s^m \rightarrow s, m \geq 0$$

or

$$p : s^n, n \geq 0$$

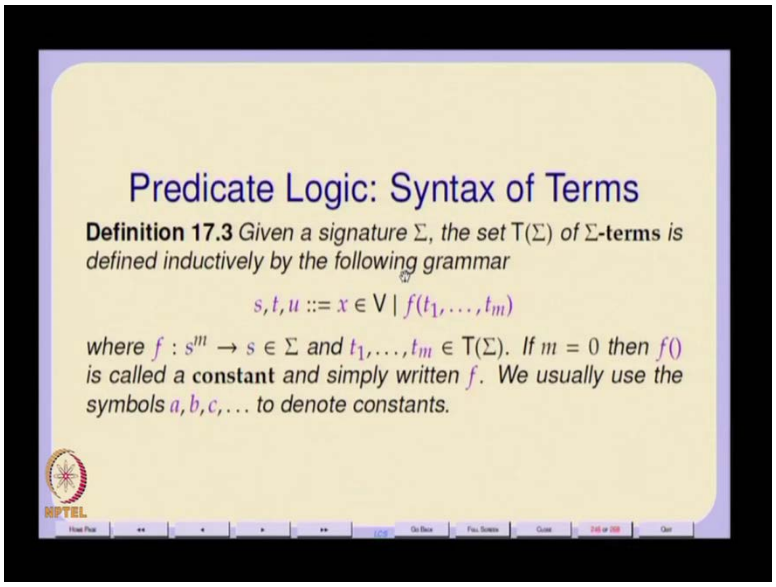
such that there is at most one string for each $f \in F$ *and each* $p \in A$. m *and* n *are respectively the* **arity** *of* f *and* p.

Here s is merely a symbol signifying a **sort** of elements. A generalization to many-sorted algebras would require the use of as many such symbols $s_1, \ldots, s_m$ as there are sorts.

And we are usually interested in a signature. So, that reflects the algebraic nature of most of whatever we do. For simplicity we will assume a 1-sorted signature where each function and each predicate symbol atomic predicate symbol has some arity specified by this short symbol s and it is by this pattern basically, right, where s is just some pattern. So, if it was a many sorted signature what we would have is many sort symbol, say, s 1, s 2, s 3, etcetera, as many as there are sorts, which is something that usually is required when you are dealing with a programming language because for programming language.

For example, it can be regarded as a many sorted algebra based on some basic sorts like int, bull, real, string character and so on so forth. And there are functions which for example, the length of a string is a function which goes from strings to integers and so you have functions which bridge sorts; you have conversion functions from let us say real's to integers, integers to real's and so on. So, you have many such functions and predicates. So, the first thing of course, is to realize that one sort it is possible to view many sorted algebras also as 1-sorted as a single 1-sorted algebra in which the various sorts are distinguished by certain new atomic predicates which tell you whether something belongs to a certain sort or not, right. So, we will keep it simple and we will just worry about a 1-sorted signature.

(Refer Slide Time: 03:00)



Predicate Logic: Syntax of Terms

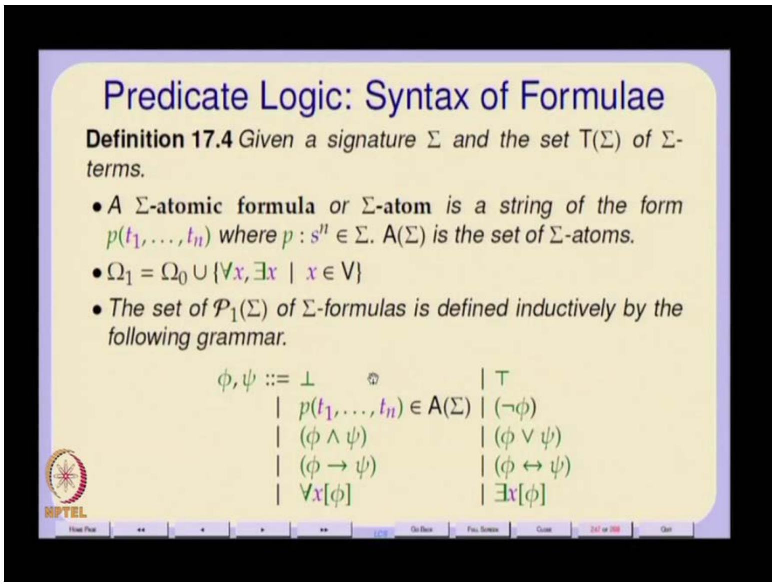
Definition 17.3 *Given a signature Σ, the set $T(\Sigma)$ of* **Σ-terms** *is defined inductively by the following grammar*

$$s, t, u ::= x \in V \mid f(t_1, \ldots, t_m)$$

where $f : s^m \to s \in \Sigma$ and $t_1, \ldots, t_m \in T(\Sigma)$. If $m = 0$ then $f()$ is called a **constant** *and simply written f. We usually use the symbols $a, b, c, \ldots$ to denote constants.*

And the other thing is right. So, this is the set of terms which depending upon arity we will use the usual mathematical notation for terms in prefix fashion, but that is in the most general case. Sometimes of course, functions are especially binary functions have got an infix notation, but since we are interested only in trees it does not matter to us whether we are looking at it in infix. We will choose the most convenient notation yeah. Then there is particular case of constants which are essentially zero array functions, very often we will not write these parentheses for those constants, and most often we will use a, b, c, d, etcetera as constant symbols.

(Refer Slide Time: 03:54)



Predicate Logic: Syntax of Formulae

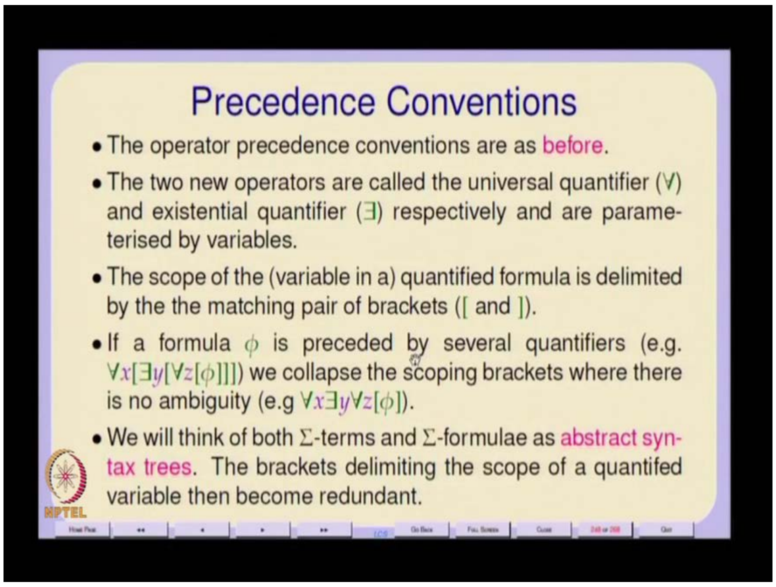
Definition 17.4 *Given a signature Σ and the set $T(\Sigma)$ of Σ-terms.*

- *A* **Σ-atomic formula** *or* **Σ-atom** *is a string of the form $p(t_1, \ldots, t_n)$ where $p : s^n \in \Sigma$. $A(\Sigma)$ is the set of Σ-atoms.*
- $\Omega_1 = \Omega_0 \cup \{\forall x, \exists x \mid x \in V\}$
- *The set of $\mathcal{P}_1(\Sigma)$ of Σ-formulas is defined inductively by the following grammar.*

$$\begin{array}{rlcl} \phi, \psi ::= & \bot & \mid & \top \\ \mid & p(t_1, \ldots, t_n) \in A(\Sigma) & \mid & (\neg\phi) \\ \mid & (\phi \wedge \psi) & \mid & (\phi \vee \psi) \\ \mid & (\phi \to \psi) & \mid & (\phi \leftrightarrow \psi) \\ \mid & \forall x[\phi] & \mid & \exists x[\phi] \end{array}$$

This is the syntax of predicate logic of which of course, the most important things are this specification of truth for the atomic predicate symbols, and for these two new operators called quantifiers. So, whatever notion on schematics we define should essentially take these into account and the signature in it. So, first order logic or predicate logic is a very general name, and it has to be parameterized by these signatures. So, we are talking about sigma formulas and sigma atomic formulas. So, therefore, it is parameterized on the signature, right.

(Refer Slide Time: 04:49)



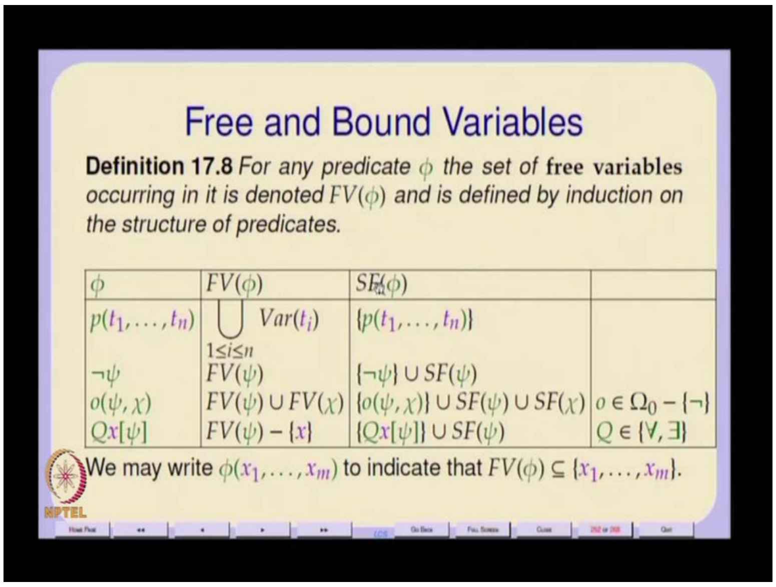
Precedence Conventions

- The operator precedence conventions are as before.
- The two new operators are called the universal quantifier ($\forall$) and existential quantifier ($\exists$) respectively and are parameterised by variables.
- The scope of the (variable in a) quantified formula is delimited by the the matching pair of brackets ([and]).
- If a formula ϕ is preceded by several quantifiers (e.g. $\forall x[\exists y[\forall z[\phi]]]$) we collapse the scoping brackets where there is no ambiguity (e.g $\forall x \exists y \forall z[\phi]$).
- We will think of both Σ-terms and Σ-formulae as abstract syntax trees. The brackets delimiting the scope of a quantifed variable then become redundant.

We will use the same precedence conventions; in addition of course, I have this scoping. I use square brackets to delimit scope, because these quantifiers they are actually parameterized on the variable. They are operators which are parameterized on a variable, and that variable has a scope which is delimited by these square brackets, yeah. So, in that sense this collection of operations omega one unlike the propositional case is actually accountably infinite set of operations to view at. So, for each variable x belonging to the set of variables v for all x there is exist an operator called for all x and there exist an operators called there exist x, yeah.

So, this is actually an infinite collection of operators, and we will give the semantics appropriately, yeah. So, we will concentrate on essentially these things, the usual precedence relations; there are these usual notions like depth and size and sub terms which are basically sub trees of the abstract syntax trees. There is this notion of variables of atom.

(Refer Slide Time: 06:16)



Free and Bound Variables

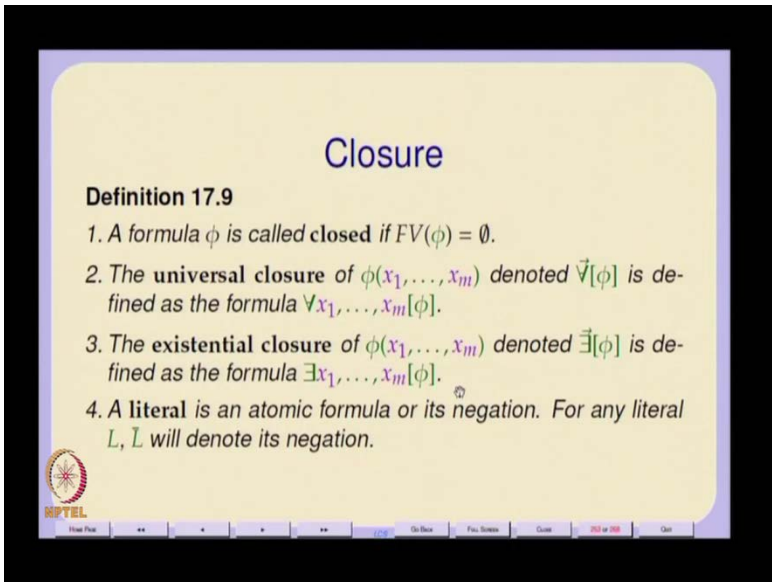
Definition 17.8 *For any predicate ϕ the set of* **free variables** *occurring in it is denoted $FV(\phi)$ and is defined by induction on the structure of predicates.*

ϕ	$FV(\phi)$	$SF(\phi)$	
$p(t_1, \ldots, t_n)$	$\bigcup_{1 \le i \le n} Var(t_i)$	$\{p(t_1, \ldots, t_n)\}$	
$\neg\psi$	$FV(\psi)$	$\{\neg\psi\} \cup SF(\psi)$	
$o(\psi, \chi)$	$FV(\psi) \cup FV(\chi)$	$\{o(\psi, \chi)\} \cup SF(\psi) \cup SF(\chi)$	$o \in \Omega_0 - \{\neg\}$
$Qx[\psi]$	$FV(\psi) - \{x\}$	$\{Qx[\psi]\} \cup SF(\psi)$	$Q \in \{\forall, \exists\}$

We may write $\phi(x_1, \ldots, x_m)$ to indicate that $FV(\phi) \subseteq \{x_1, \ldots, x_m\}$.

And because of the fact that you have quantifiers which actually have bound variables; we have to distinguish between free variables and bound variables. And for any quantifier q where q might be either for all there exist, q x psi consist of the free variables of q x psi consist of all the free variables of psi except x, and that is recursively defined. So, similarly we can talk about the sub formulas sub trees essentially of a formula regarded as an abstract syntax tree.

(Refer Slide Time: 06:56)



Closure

Definition 17.9

1. *A formula ϕ is called* **closed** *if $FV(\phi) = \emptyset$.*
2. *The* **universal closure** *of $\phi(x_1, \ldots, x_m)$ denoted $\vec{\forall}[\phi]$ is defined as the formula $\forall x_1, \ldots, x_m[\phi]$.*
3. *The* **existential closure** *of $\phi(x_1, \ldots, x_m)$ denoted $\vec{\exists}[\phi]$ is defined as the formula $\exists x_1, \ldots, x_m[\phi]$.*
4. *A* **literal** *is an atomic formula or its negation. For any literal L, $\overline{L}$ will denote its negation.*

And finally, we will think of closures. Firstly it is important to state what a closed formula is. A closed formula is simply one which has no free variables. And in that sense the entire language of propositional logic actually dealt with closed formulae, yeah. There is also another importance for closed formulae, and that is that strictly speaking most of the theorems in a mathematical theory implicitly or explicitly are closed formulae. It will be expressed as closed formulae, and very often if it is implicit then what you do usually is you have a universal closure, which essentially means you look at the way the formula looks, look all the free variables in the formula and simply put a universal closure on all of them.

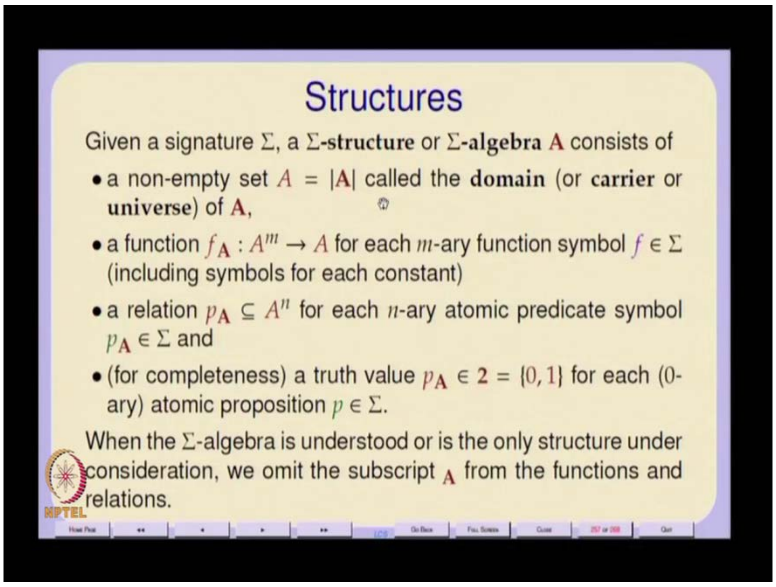
So, most of our theorems essentially are of this nature. They are universally closed statements, and that is the way we will express them. And of course, when we talk of particular cases there is you can also think of an existential closure. And usually existential closure is made implicit in our mathematical theories by using constant symbols or particular kinds of symbols devoted to show that it is not universally closed. But to make these things explicit we will also define the universal closure operation which essentially means that if I were to take all the free variables in the formula and put a universal quantifier or an existential quantifier in front of the formula, then I essentially get its universal closure or existential closure.

Of course, there could be other formulas which have some quantifier's universal and some quantifier existential. So, these universal and existential closure formulas are important and general closed formulas are important. Also this is extra piece of notation just to say that they predicate the formula phi; it is actually just the formula phi, but this implicitly says that the set of its free variables is entirely contained in the set x 1 to x m. So, we just borrow these existential from standard mathematical notation and use them, right. So, now we are ready to proceed towards the semantics.

So, one thing about the semantics of predicate logic is that the language itself is parameterized on a certain signature sigma. And so what we would call a model or semantics for the language parameterized on a certain signature sigma requires essentially what is known as sigma algebra. So, think of it this way. If you were to describe the theory of groups, then groups have a certain signature, a binary multiplicative operator, a constant element called the identity element and an inversion a unary inversion operator.

And these three actually form the operations of a group and in addition you have may be the equality relation. So, think of it. So, that equality is an atomic predicate, right. So, you have a group in general defined by this signature of these three operations, a constant, a unary operation and a binary operation, and an atomic predicate namely equality.

(Refer Slide Time: 11:10)



Structures

Given a signature Σ, a **Σ-structure** or **Σ-algebra $\mathbf{A}$** consists of

- a non-empty set $A = |\mathbf{A}|$ called the **domain** (or **carrier** or **universe**) of $\mathbf{A}$,
- a function $f_{\mathbf{A}} : A^m \rightarrow A$ for each m-ary function symbol $f \in \Sigma$ (including symbols for each constant)
- a relation $p_{\mathbf{A}} \subseteq A^n$ for each n-ary atomic predicate symbol $p_{\mathbf{A}} \in \Sigma$ and
- (for completeness) a truth value $p_{\mathbf{A}} \in \mathbf{2} = \{0, 1\}$ for each (0-ary) atomic proposition $p \in \Sigma$.

When the Σ-algebra is understood or is the only structure under consideration, we omit the subscript $_{\mathbf{A}}$ from the functions and relations.

So, when you describe groups, therefore, you are looking at essentially statements involving only elements taken from the signature, terms from that signature, form from that signature, right. And what you are saying is examples of groups would be particular sets; let us say like the integers with addition, negation and zero as identity element, and of course, integer equality as the only binary predicates and that is a group. So, integers under addition with the unary negation and zero form a structure for the signature sigma. You can also take let us say the real's under multiplication, what? No, you cannot; there is a problem there, but anyway.

So, you could take for example, the positive reals under multiplication and with reciprocal and one as identity element, and that is the structure which satisfies the same signature, okay. So, when you are looking at statements about a certain statements first order logic or predicate logic statements parameterized on a certain signature, then you are essentially looking at models also by models you are looking at structures which satisfy that statement for which that statement is true, right. So, that is what we will. So, we will given any signature sigma; we will look at a sigma structure or a sigma algebra which consist of a nonempty set. This non empty set like in the case of groups, it might be integers or it might be real's, a positive real's.

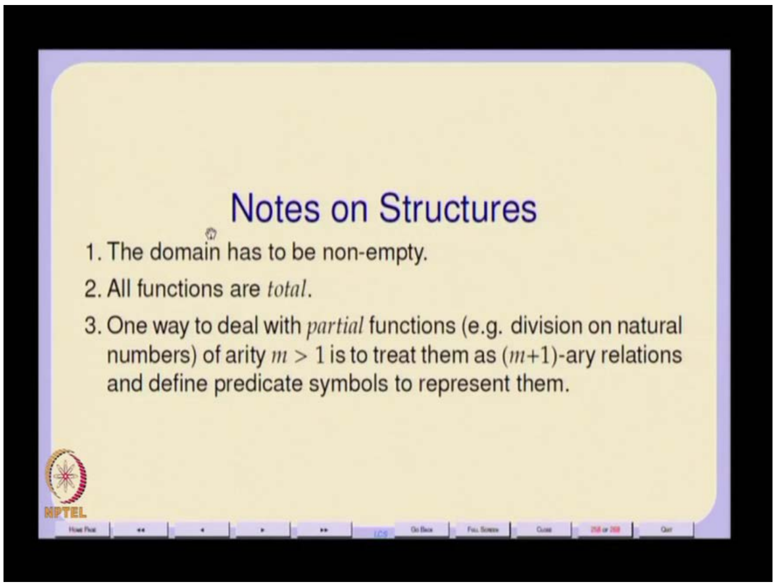
So, this nonempty set is called the carrier of the algebra that is the algebraic terminology. A logical terminology is that it is a universe of discourse or it is a domain of the algebra, right. So, all elements that we are considering in the model essentially belong to this set A which is a carrier of the algebra. And of course, for each emery function symbol in the signature, you have an emery function on the carrier set, okay. So, what we will do is for definiteness we will subscript; of course, I am using the same thing; my models are brown in color. So, the color itself should tell you which is something that belong to the model, and what is something that belongs to the language.

And of course, within the language I am distinguishing between a color of terms and the color of predicates. So, the green and violet should distinguish between them, but at the level of the model I am keeping everything brown in color, right. So, corresponding to every violet symbol f in sigma, there is a function of the same arity on the domain or on the universe of discourse, right, defined on that universe of discourse. So, including in the case of symbols which are constant symbols or zero array functions that they have to identified constant symbols also in their domain, right.

So, in particular the identity element for groups has to be identified in the domain. So, in the case of group of positive real's and multiplication the identity element is one; in the case of integer-integer group under addition, the identity element is zero and those have to be identified. And of course, a relation p A for each atomic predicate symbol, I am sorry this subscript A should not be there; for each atomic predicate symbol p in sigma there is a relation on this carrier set of this same arity and for completeness though it is not important. In case you have a predicate symbol which is of which is zero array, then essentially we regard that as an atomic proposition.

So, which means it takes a value for which there is to be a value assigned from the set zero one which is our set of truth values, but this last point is just I have just written it for completeness, but usually it is not important. We never have a Boolean constant hanging around our theory of groups. I mean as simple as that we are. So, when the sigma algebra is understood or there is only one structure under consideration, then we will just omit this subscript A from all these functions and predicate. It tends to platter up the notations. So, we will throw that out. There are certain important things to realize here. One is that all these functions that we associate with function symbols are total functions. So, there are no partial functions that we consider; that is one thing to be noted.

(Refer Slide Time: 17:17)



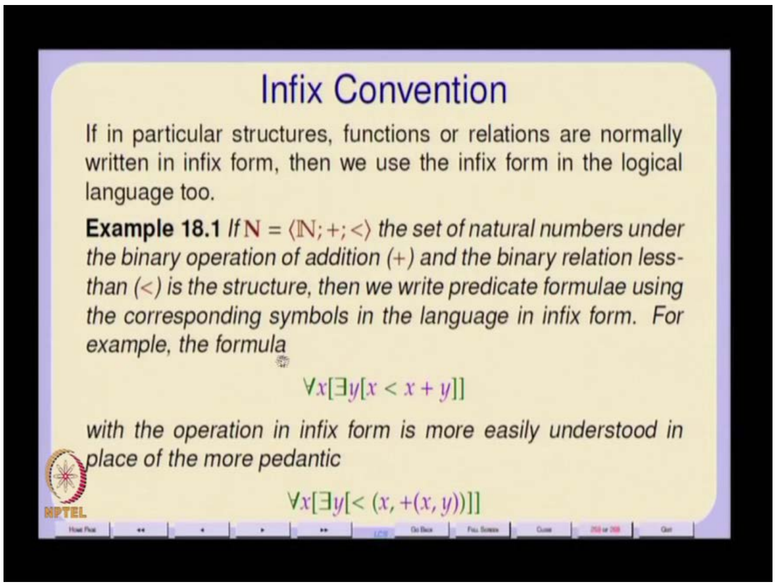
Notes on Structures

1. The domain has to be non-empty.
2. All functions are *total*.
3. One way to deal with *partial* functions (e.g. division on natural numbers) of arity $m > 1$ is to treat them as $(m+1)$-ary relations and define predicate symbols to represent them.

And the only way to deal with partial functions is to actually talk about a corresponding relation in state. So, supposing you want to talk about the theory of a real's under the division operation, right, with equality as let us say the only predicate symbol. Now the problem is normally we regard division as an operation, but division on the reals is not closed, right. So, which means that and division is therefore, a partial function on the real's which means that the only way we have to deal with it in a first order framework is to actually define division instead as a ternary relation.

So, if you have to deal with partial functions of arity m, we will just assume that there m plus 1 array relations. So, in that sense we do not actually lose anything. So, it makes things a little artificial, but on the other hand we do not lose any expressiveness because of this assumption here. So, we will treat partial functions essentially as relations, and therefore, there has to be an exact correspondence between the signatures, signature of the language and the signature structure; therefore, which means there will be corresponding atomic predicates associated with these partial functions, and they will essentially have the meaning of relations of appropriate arity in the structure, right.

(Refer Slide Time: 19:10)



Infix Convention

If in particular structures, functions or relations are normally written in infix form, then we use the infix form in the logical language too.

Example 18.1 *If* $\mathbf{N} = \langle \mathbb{N}; +; < \rangle$ *the set of natural numbers under the binary operation of addition* $(+)$ *and the binary relation less-than* $(<)$ *is the structure, then we write predicate formulae using the corresponding symbols in the language in infix form. For example, the formula*

$$\forall x[\exists y[x < x + y]]$$

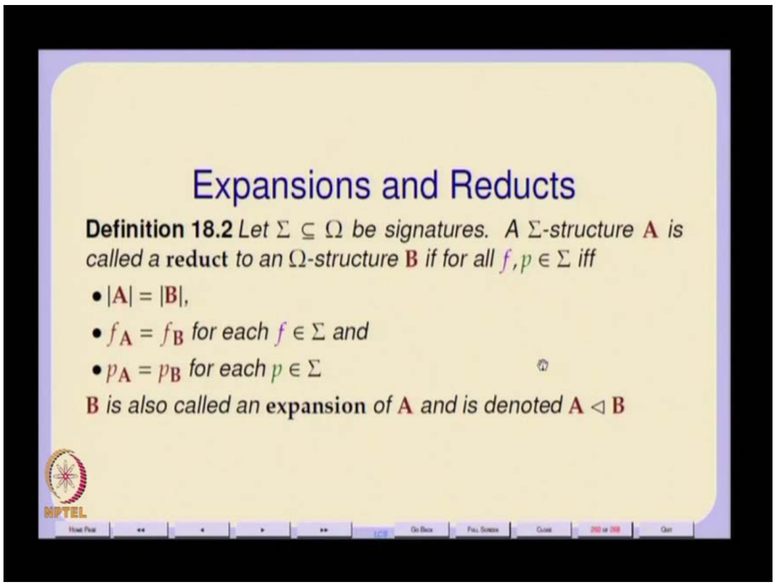
with the operation in infix form is more easily understood in place of the more pedantic

$$\forall x[\exists y[< (x, +(x, y))]]$$

The other thing is the other notational conventional we will use is. So, here is the simple example of a predicate like for all x there exist y such that x is less than x plus y. So, we will write predicates in this fashion, basically since less than and plus r usually used in infix notation, we will use them in infix notation. So, here is the structure with a carrier set which is the set of naturals.

There is a single operation binary operation addition and a single binary relation less than the relation let us say, and you want to essentially describe statements about this structure in first order logic. So, this is a typical statement; strictly speaking you would write things in a prefix notation. They become sort of messy and unreadable. So, we will keep it readable by using infix notation wherever it is conventionally used.

(Refer Slide Time: 20:21)



Expansions and Reducts

Definition 18.2 *Let $\Sigma \subseteq \Omega$ be signatures. A Σ-structure $\mathbf{A}$ is called a* **reduct** *to an Ω-structure $\mathbf{B}$ if for all $f, p \in \Sigma$ iff*

- $|\mathbf{A}| = |\mathbf{B}|$,
- $f_{\mathbf{A}} = f_{\mathbf{B}}$ *for each* $f \in \Sigma$ *and*
- $p_{\mathbf{A}} = p_{\mathbf{B}}$ *for each* $p \in \Sigma$

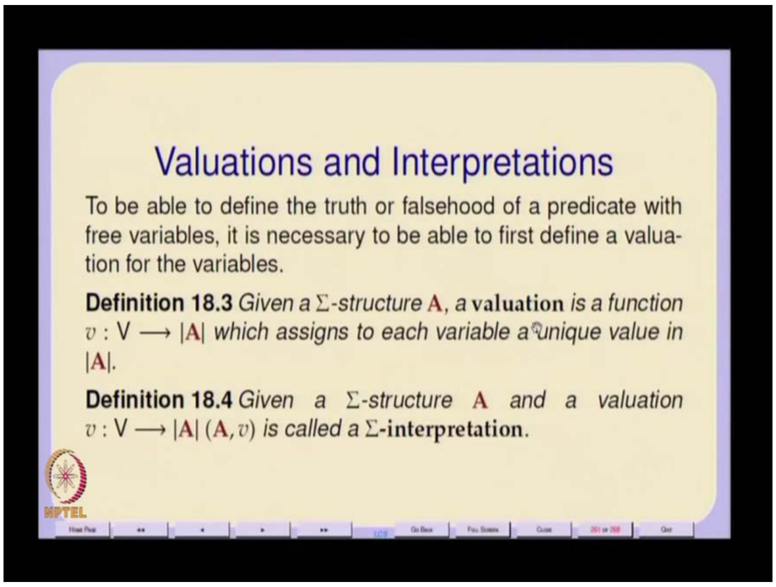
$\mathbf{B}$ *is also called an* **expansion** *of* $\mathbf{A}$ *and is denoted* $\mathbf{A} \lhd \mathbf{B}$

So, of course, what we might do is we might take the theory of integers you might take the theory of monodies for example, and the natural numbers under addition with identity element zero. And what you might do is you might actually equip some other operation also; I mean you might actually add some other relations, may be you might have a less than and then you might have a divisor of relation and so on and so forth, right. So, it is quite possible that I might expand the signature sigma to another signature omega, right. So, we would say that as. So, let us look at it this way.

Now if you expand this signature, then the corresponding structure also needs to be expanded, right. Basically there was a need to introduce some new symbols in the signature, because you are interested in certain relations of operations in that signature. So, you have to augment the structure also with those corresponding functions or relations, yeah. So, we will say that B is a structure an omega structure; given that sigma is a subset of omega, we will say that an omega structure B is an expansion of a sigma structure A if they both have the same domain of discourse.

And every function symbol in A is also there in B and every predicate symbol in A is also there in B. And essentially what we are saying is the semantics the meanings of their individual functions and predicates in the structure A are preserved, and we are just expanding the set of operations and relations by something else. So, this expansion will be denoted A triangle B. So, this is here.

(Refer Slide Time: 22:39)



Valuations and Interpretations

To be able to define the truth or falsehood of a predicate with free variables, it is necessary to be able to first define a valuation for the variables.

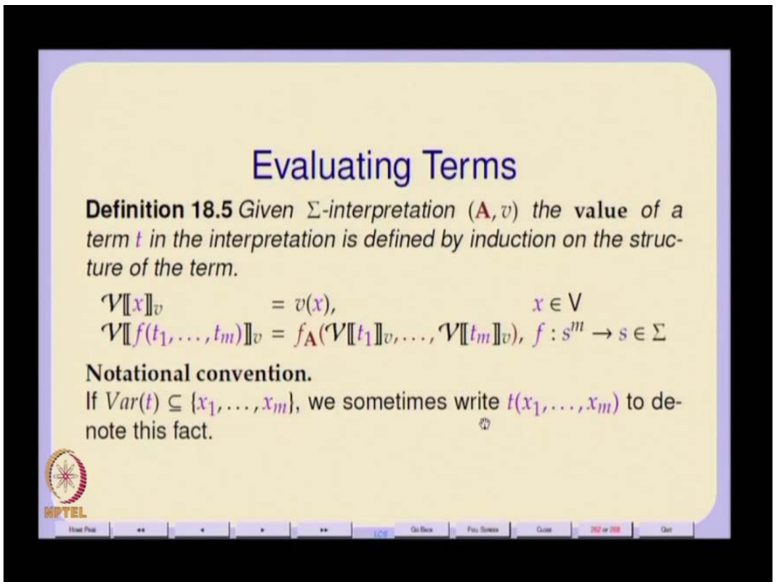
Definition 18.3 *Given a* Σ*-structure* $\mathbf{A}$, *a* **valuation** *is a function* $v : \mathsf{V} \longrightarrow |\mathbf{A}|$ *which assigns to each variable a unique value in* $|\mathbf{A}|$.

Definition 18.4 *Given a* Σ*-structure* $\mathbf{A}$ *and a valuation* $v : \mathsf{V} \longrightarrow |\mathbf{A}|$ $(\mathbf{A}, v)$ *is called a* Σ-**interpretation**.

Now in the case of propositional logic of course, we had the notion of truth assignments where directly atomic propositions were given truth values from the set zero one. In the case of predicate logic, we are talking about parameterized propositions. So, the truth of a proposition might depend on its parameters. So, in particular what it means is it depends upon the values of the terms that occur in the proposition. So, what we require is a notion of a valuation, yeah So, we define a valuation as a function v which assigns to each variable in our variable set v and element from the domain of the sigma structure A.

So, you have a particular model in mind let us say which has a carrier set and functions and relations for the various symbols in your signature. And now what we do is we assign to each variable symbol a value from that carrier set, yeah, and this structure along with a valuation will be called a sigma interpretation, yeah. So, what you are saying is you take any statement in predicate logic, and we are going to interpret it in a structure. And that statement and predicate logic it might have lots of free variables in which case you need some way to interpret those free variables as essentially values from the domain of that structure right. So, that is what the valuation v gives you. So, this ordered pair structure A along with a valuation is called as an interpretation sigma interpretation, right.

(Refer Slide Time: 24:50)



Evaluating Terms

Definition 18.5 *Given Σ-interpretation $(\mathbf{A}, v)$ the* **value** *of a term t in the interpretation is defined by induction on the structure of the term.*

$$\begin{aligned} \mathcal{V}[\![x]\!]_v &= v(x), & x \in V \\ \mathcal{V}[\![f(t_1,\ldots,t_m)]\!]_v &= f_{\mathbf{A}}(\mathcal{V}[\![t_1]\!]_v,\ldots,\mathcal{V}[\![t_m]\!]_v), & f : s^m \to s \in \Sigma \end{aligned}$$

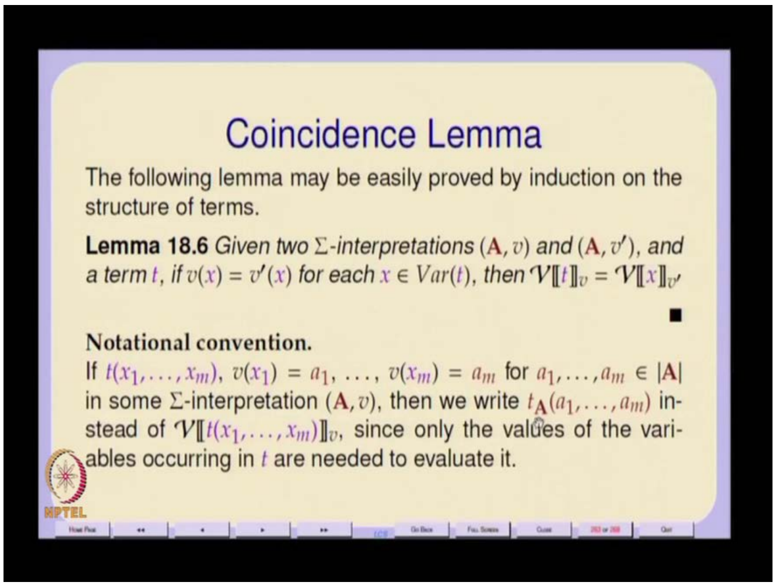
Notational convention.
If $Var(t) \subseteq \{x_1,\ldots,x_m\}$, we sometimes write $t(x_1,\ldots,x_m)$ to denote this fact.

So, given a sigma interpretation the first thing of course, is to evaluate the terms, right. So, we proceed with this evaluation of terms, the value of a term t in the interpretation is defined by induction on the structure of that term. So, for any variable x I have this function called v which is parameterized on some. So, we are looking at the value of this term under this valuation v actually in the structure A, right; that is the way to look at it. So, for any variable of course, it is just whatever the valuation assigns to that variable; that is the value of the variable. In a certain sense that is the only meaning of that variable and for any term f of t 1 to t m where t 1 to t m is terms themselves. Corresponding to f in the structure there is an emery function f a; you apply it to the meanings of the individuals terms, right.

So, this is your meaning function which for terms which courses through by structural induction to the lowest level. As usual sometimes we will write things like this, just like for predicates the free variables might be listed out; for terms also we might just all the variables that occur in that term could be listed out. Or in fact, more than the variables that occur in the term could be listed out, yeah. So, once you have evaluated that terms.

So, once every term has been given a meaning in the structure A, you are essentially ready to interpret all the sentence all the predicates of the language, right.

(Refer Slide Time: 27:11)



Coincidence Lemma

The following lemma may be easily proved by induction on the structure of terms.

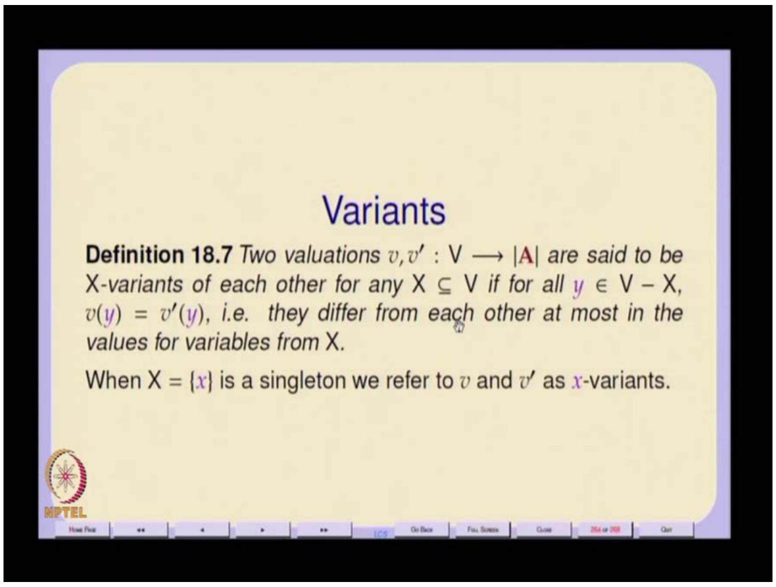
Lemma 18.6 *Given two Σ-interpretations $(\mathbf{A}, v)$ and $(\mathbf{A}, v')$, and a term t, if $v(x) = v'(x)$ for each $x \in Var(t)$, then $\mathcal{V}[\![t]\!]_v = \mathcal{V}[\![x]\!]_{v'}$* ■

Notational convention.
If $t(x_1, \ldots, x_m)$, $v(x_1) = a_1, \ldots, v(x_m) = a_m$ for $a_1, \ldots, a_m \in |\mathbf{A}|$ in some Σ-interpretation $(\mathbf{A}, v)$, then we write $t_{\mathbf{A}}(a_1, \ldots, a_m)$ instead of $\mathcal{V}[\![t(x_1, \ldots, x_m)]\!]_v$, since only the values of the variables occurring in t are needed to evaluate it.

But before that we have something called the coincidence lemma. This is a very simple lemma. Supposing you have two different valuations for the same structure; so, you have a single structure A and two different interpretations which differ only in the valuations. And for any term t, if the two valuations v and v prime give you the same value, then essentially the value of t under v is the same as the value of. I am sorry this should be t here, the value of t under v prime, yeah.

And we will use a lot of shortcuts; sometimes given a term t of x 1 to x m we will and given that the variables x 1 to x m have some values a 1 to a m some constant values in this domain of the structure we will just write t a of a 1 to a m to denote the value of t of x 1 to x m instead of this more complicated way of writing things, yeah. Because especially this coincidence lemma essentially says that sometimes we are only interested in a certain subset of the variables; we are not interested in the entire lot of infinite variables set you have. So, it makes it more convenient to just write this, okay.

(Refer Slide Time: 28:39)



Variants

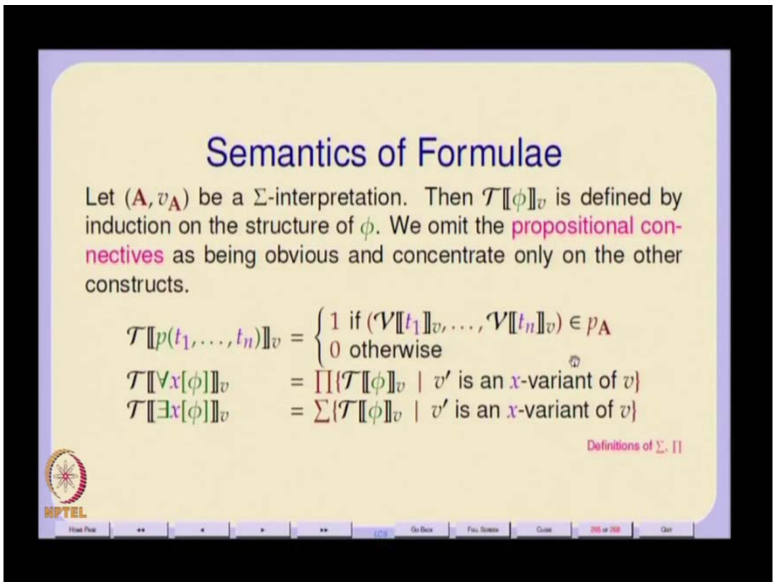
Definition 18.7 *Two valuations $v, v' : \mathsf{V} \longrightarrow |\mathbf{A}|$ are said to be* X*-variants of each other for any* $\mathsf{X} \subseteq \mathsf{V}$ *if for all* $y \in \mathsf{V} - \mathsf{X}$, $v(y) = v'(y)$, *i.e. they differ from each other at most in the values for variables from* X.

When $\mathsf{X} = \{x\}$ is a singleton we refer to v and v' as x-variants.

So, now I come to an important thing called a variant, yeah. So, I take two valuations. So, assume that I have got a single structure A for a single sigma structure A and I take two different valuations v and v prime and let me also take a subset of the variables v, a subset x of the variables v, then I would say v prime is an x variant of v or they are actually x variants of each other. If for every y for every variable that is not in x, the two valuations give the same value to the variable; assign the same value to the variable, okay.

So, by x variants what you are saying essentially is that the two valuations might differ only for the values they are assigned to the variables in x, and for all other variables they give you the same value, right. So, very often when x is a singleton set, we just refer to them as singleton set containing a single variable x, we just refer to them as x variants, yeah, a small x variants.

(Refer Slide Time: 30:14)



Semantics of Formulae

Let $(\mathbf{A}, v_{\mathbf{A}})$ be a Σ-interpretation. Then $\mathcal{T}[\![\phi]\!]_v$ is defined by induction on the structure of ϕ. We omit the propositional connectives as being obvious and concentrate only on the other constructs.

$$\mathcal{T}[\![p(t_1, \ldots, t_n)]\!]_v = \begin{cases} 1 \text{ if } (\mathcal{V}[\![t_1]\!]_v, \ldots, \mathcal{V}[\![t_n]\!]_v) \in p_{\mathbf{A}} \\ 0 \text{ otherwise} \end{cases}$$

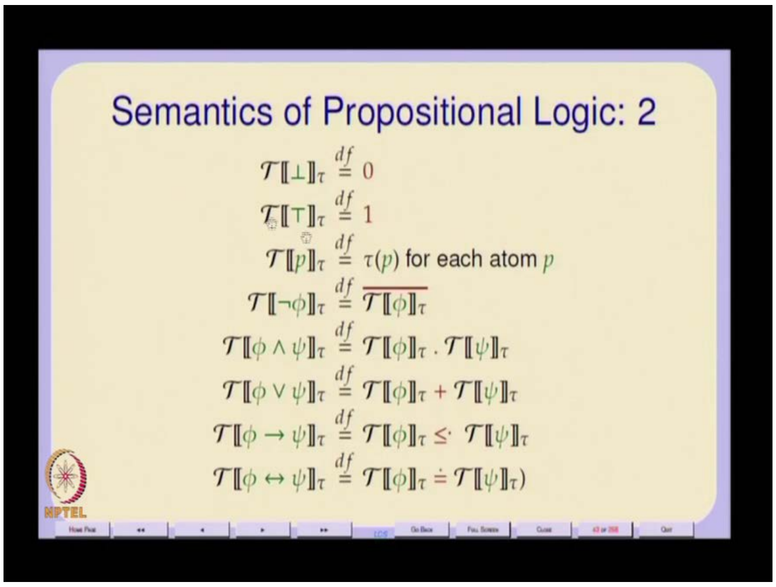
$$\mathcal{T}[\![\forall x[\phi]]\!]_v = \prod\{\mathcal{T}[\![\phi]\!]_{v'} \mid v' \text{ is an } x\text{-variant of } v\}$$

$$\mathcal{T}[\![\exists x[\phi]]\!]_v = \sum\{\mathcal{T}[\![\phi]\!]_{v'} \mid v' \text{ is an } x\text{-variant of } v\}$$

Definitions of $\sum$, $\prod$

And now we are ready to give the semantics of formulae. And what I will do is I will not worry about the propositional connectives, because the schematics of propositional connectives once you have got truth values are known to us.

(Refer Slide Time: 30:33)



Semantics of Propositional Logic: 2

$$\mathcal{T}[\![\bot]\!]_\tau \overset{df}{=} 0$$

$$\mathcal{T}[\![\top]\!]_\tau \overset{df}{=} 1$$

$$\mathcal{T}[\![p]\!]_\tau \overset{df}{=} \tau(p) \text{ for each atom } p$$

$$\mathcal{T}[\![\neg\phi]\!]_\tau \overset{df}{=} \overline{\mathcal{T}[\![\phi]\!]_\tau}$$

$$\mathcal{T}[\![\phi \wedge \psi]\!]_\tau \overset{df}{=} \mathcal{T}[\![\phi]\!]_\tau \,.\, \mathcal{T}[\![\psi]\!]_\tau$$

$$\mathcal{T}[\![\phi \vee \psi]\!]_\tau \overset{df}{=} \mathcal{T}[\![\phi]\!]_\tau + \mathcal{T}[\![\psi]\!]_\tau$$

$$\mathcal{T}[\![\phi \to \psi]\!]_\tau \overset{df}{=} \mathcal{T}[\![\phi]\!]_\tau \leq\cdot\ \mathcal{T}[\![\psi]\!]_\tau$$

$$\mathcal{T}[\![\phi \leftrightarrow \psi]\!]_\tau \overset{df}{=} \mathcal{T}[\![\phi]\!]_\tau \doteq \mathcal{T}[\![\psi]\!]_\tau)$$

So, we just assume that this is true for all of them. The only difference here is that here in the case of propositional logic we had this subscript tau which is a truth assignment; in our case now what we are going to have is a valuation. So, once you have specified truth value is for atomic proposition, then it is a trivial matter to look upon them as truth values themselves and give the semantics of the proposition connectives in an identical fashion, yeah, in an absolutely analogues fashion. So, we will not waste time on this.

So, for any atomic predicate p of t 1 to t n under a valuation v, this predicate is true if having evaluated t 1 to t n under the same valuation v, you get essentially at n tupil. So, here we are assuming that p is a unary predicate for some value of n. This n tupil of values actually belongs to that relation p A which was defined as the corresponding relation for the atomic predicate symbol p. So, if it does belong to it of course, then you give the truth value one; otherwise, you give the truth values here, right. So, this is what.

So, essentially what we are saying is atomic predicates, they become propositions once you give values to the terms, and the truth of that depends upon whether it belongs to the corresponding relation that you define for that predicate symbol or not, right; otherwise it is false. So, it is true if this n tupil belongs to p A and it is false otherwise. So, this is of course more interesting case. First you suppose that when your intuitive understanding of the universal quantifier is essentially that phi is true for whatever value you might assign x; in that sense the truth of this if this universally quantified statement for all x phi is true regardless of any value that might be assigned to x, yeah; that is an intuitive understanding of this universal quantifier.

And in the in the case of the existential quantifier the intuitive understanding is that there is a particular value which you must give to x which will make phi true, and if such a particular value does exist in the structure A then you would say that there exist x phi is true. Note that in both cases in any mathematical theory the universally quantified and the existentially quantified statements are independent of that bound variable. And moreover in the case of the existential quantifier, you are not actually interested in the actual value unless you are designing an algorithm for it, right. As a pure mathematical statement you are not actually interested in the value; you just want to know the truth of existential statement.

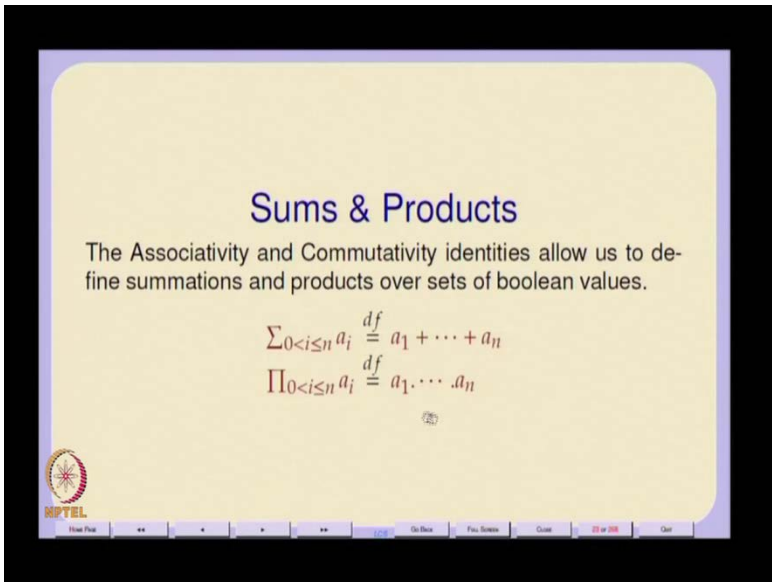
So, all you need to know is that there exist there exist such value x. In particular the reason this becomes important is that there is a difference between a constructive approach and a non-constructive approach. So, an algorithmic way of proving existence would actually find a value x which would make the statement true. A non-constructive approach might prove it by contradiction, but assume that there is no value and then attain some contradiction. And if you attain the contradiction you have proved that there exist a value, but since you did not construct the value you do not know what it is, and yet the statement would be true, right.

So, there is there is something non-constructive about the way we are doing all our mathematics anyway; the moment you do proofs by contradiction you are being non-constructive. So, what we are saying, therefore, is that all these quantified statements their truth of false it is independent of the values assigned to the bound variables right. But what does it mean now? I have got a particular valuation v and I am saying that this statement is true if and only if or at least I want to say that this universally quantified statement is true if and only if for any value in the domain that I might assign x, the predicate phi is true which means I am looking at all possible x variants of B.

So, the free variables of this predicate phi will anyway get their values from the valuation B. The bound variables do not get the values from the valuation B; instead you consider all possible x variants for each bound variable x and then evaluate phi. So, think of it this way. If your domain A was infinite, then you are essentially considering an infinite number of different x variants. You are evaluating phi for all that infinite number of different x variants, and you are evaluating its truth. So, in each case you will get a zero or a one, but this set and then what I am saying is.

So, this universally quantified statement is true if the product is one, right, once you have got zeros and ones this product. So, this product is not going to be a product over an infinite set. It is going to be a product over a finite set; in fact it cannot have more than two elements, right.

(Refer Slide Time: 37:58)



Sums & Products

The Associativity and Commutativity identities allow us to define summations and products over sets of boolean values.

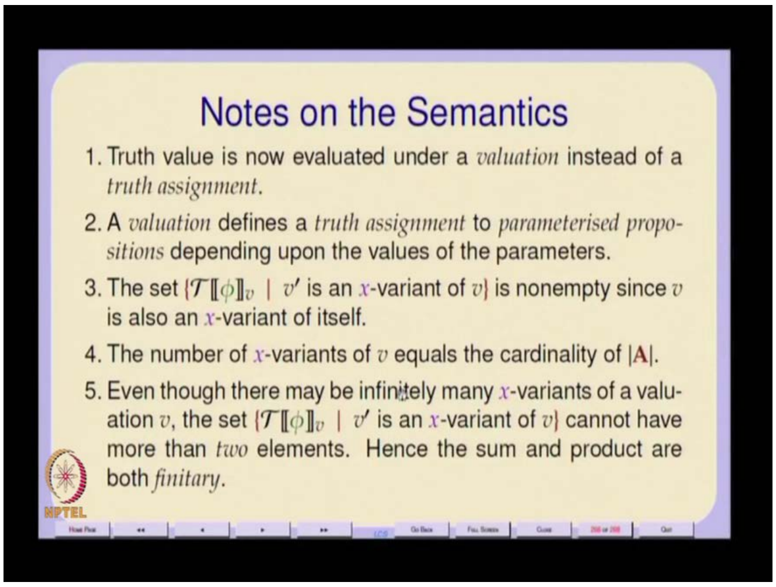
$$\sum_{0<i\leq n} a_i \overset{df}{=} a_1 + \cdots + a_n$$
$$\prod_{0<i\leq n} a_i \overset{df}{=} a_1 . \cdots . a_n$$

So, we have already defined finite products and finite sums here, right. So, your universal and existential quantifiers essentially are just products and sums over essentially at one element or a two element set, right. It cannot be more than that. Of course, the important thing is that these sets are not empty. They have at least one element because there is at least one valuation v if A is nonempty. It is necessary to go back to this notion of a structure. Yeah this structure is a nonempty set. This domain has to be nonempty; that is important.

If you allow the possibility of an empty domain, then actually except for your existential statement all universal statements should be true, right. So, the domain has to be nonempty and what we are saying is that empty domains are not interesting. The only interesting domains are nonempty, and then once your domain is nonempty, it implies that this set is nonempty. It will have at least one element especially for the valuation v itself. By the way the notion of variant is such that every valuation is an x variant of itself for any variable x, right, because all we are saying is that for every variable other than x, the two valuations gives the same value.

We are not denying the fact denying the possibility that for x also they might have the same value in which case they have the same valuation. So, every valuation is an x variant of itself for any variable x. So, this set t phi of v by the way this has to be v prime here in the subscript; here also it has to be v prime. And for every variant v prime of v, this set is nonempty. It will have at least one; if your domain is nonempty it will have this will be nonempty also, but because this is only truth values it cannot have more than two elements.

(Refer Slide Time: 40:37)



Notes on the Semantics

1. Truth value is now evaluated under a *valuation* instead of a *truth assignment*.
2. A *valuation* defines a *truth assignment* to *parameterised propositions* depending upon the values of the parameters.
3. The set $\{\mathcal{T}[\![\phi]\!]_{v'} \mid v' \text{ is an } x\text{-variant of } v\}$ is nonempty since v is also an x-variant of itself.
4. The number of x-variants of v equals the cardinality of $|\mathbf{A}|$.
5. Even though there may be infinitely many x-variants of a valuation v, the set $\{\mathcal{T}[\![\phi]\!]_{v'} \mid v' \text{ is an } x\text{-variant of } v\}$ cannot have more than *two* elements. Hence the sum and product are both *finitary*.

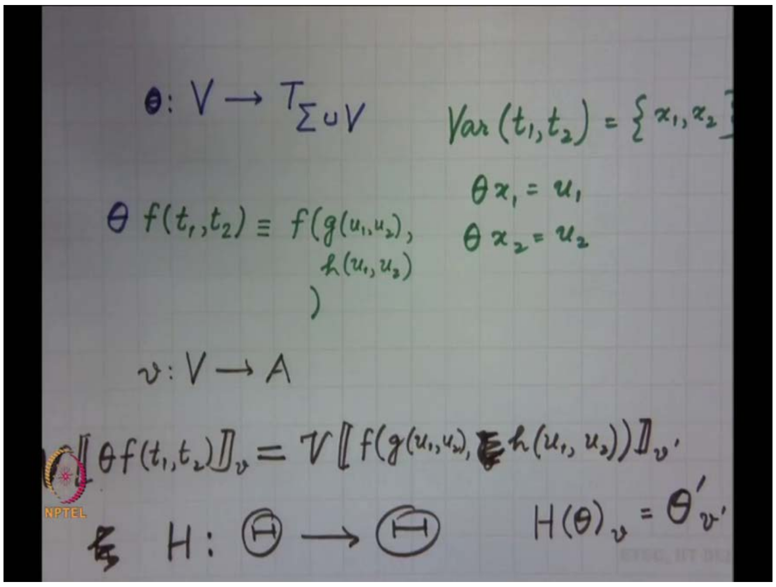
So, the empty subset is out. So, either it is just a set containing zero or a set containing one or a set containing both zero and one; that is it. And in the case of the universal quantifier the product of the set containing zero and one will give you zero anyway. So, which essentially means that it is not universally true, and for the existential quantifier this sum will give you one only if one belongs to this set, right. So, what we have done is we have actually given a finite tree definition of this semantics in certain set because this set that we are considering is going be finite. So, this finite products and.

So, we are not we are not violating the finite treeness of the definitions of sigma phi; that is what I am trying to say. So, the truth value is not evaluated under a valuation instead of a truth assignment. A valuation defines a truth assignment to a parameterized proposition depending upon the values of the parameters. And this set is nonempty since our domain is nonempty and every valuation is a variant of itself. So, the number of variants of a valuation equals cardinality of the set for any variable, and this set cannot have more than two elements, right.

So, we have given completely finite tree schematics, though actually in principle we are essentially looking at finite tree descriptions of possibly infinite structures, yeah. So, if you are looking at properties of integers or properties of reals, then they are essentially infinite structures. And when you make inverse the statements about them, we are essentially trying to give a finite tree description for essentially infinite number of properties, yeah. So, that is the schematics.

And in a certain sense it is intuitively an obvious schematics, and of course, we can take the propositional by including the propositional connectives and their schematics from this table we essentially get full semantics for the full language, right, but I will not write that down here. So, the other thing we should realize is that this notion of x variants is somewhat closely related to the notion of substitutions, yeah.

(Refer Slide Time: 43:55)



$\theta : V \rightarrow T_{\Sigma \cup V}$

$Var(t_1, t_2) = \{x_1, x_2\}$

$\theta x_1 = u_1$

$\theta x_2 = u_2$

$\theta f(t_1, t_2) \equiv f(g(u_1, u_2), h(u_1, u_2))$

$v : V \rightarrow A$

$[\![\theta f(t_1, t_2)]\!]_v = V[\![f(g(u_1, u_2), h(u_1, u_2))]\!]_{v'}$

$H : \Theta \rightarrow \Theta$

$H(\theta)_v = \theta'_{v'}$

So, there is a particular. So, let us look at substitutions. I will not go into great detail; substitution theta is a function from the variables to terms. So, we have this set of terms t sigma union v, right. So, look at this notion of evaluating terms and look at this notion of substitution. When you apply theta to some term I mean just to some term which might be some f of let us say some t 1 to t 2, then you are doing a replacement of the variables in t 1 and t 2 by some terms. So, let us assume that the variable of t 1 and t 2 are the union, let us say there are just two variables x 1 and x 2. And let us assume that theta of x 1 is some new term u 1 and theta of x 2 is some new term u 2.

So, what we are saying is that. So, you take the substitution theta and essentially this will give you a syntactically a term f of some other let us say g of u 1 and u 2 and may be a some h of u 1 and u 2, right. You look at valuations; valuations of functions from v to essentially the domain. There is some connection between valuations and substitutions. So, the effect of a substitution; so, under a valuation v, so if you evaluate this theta of f t 1, t 2 then under some valuation v then the effect of the substitution theta can be captured by another valuation v prime such that if you evaluate v f of g u 1, u 2 h u 1, u 2 under this valuation v prime, then these two values are equal, right.

So, there is some connection between substitutions and valuations. Notice that valuations are in the semantic domain, substitutions are in the syntactic domain, right. So, you take any substitution, its meaning can be expressed in terms of a higher order function on valuations, purely schematically. So, there is function on. So, corresponding to each theta there exist a function h from this set of all possible substitutions theta to the set of all possible substitutions theta such that h of theta under v is equal to some theta prime under v prime, yeah. So, this is the connection that we need to establish between purely syntactic substitutions and their schematic counterparts namely valuations.

So, each substitution can be thought of as a function which actually transforms valuations from one to another, right. So, vowel formula is this and we will prove some lemma before we proceed ahead, yeah. So, this is a concept that we need to formulize, and this will be useful for a full description. Then if you can do this if you can specify this function h, then you have essentially completely captured these schematic notions of valuations in terms of the syntactic notion of substitution and that two can mutually interact, yeah. So, we will look at this in the next lecture, yeah. We will stop here today.