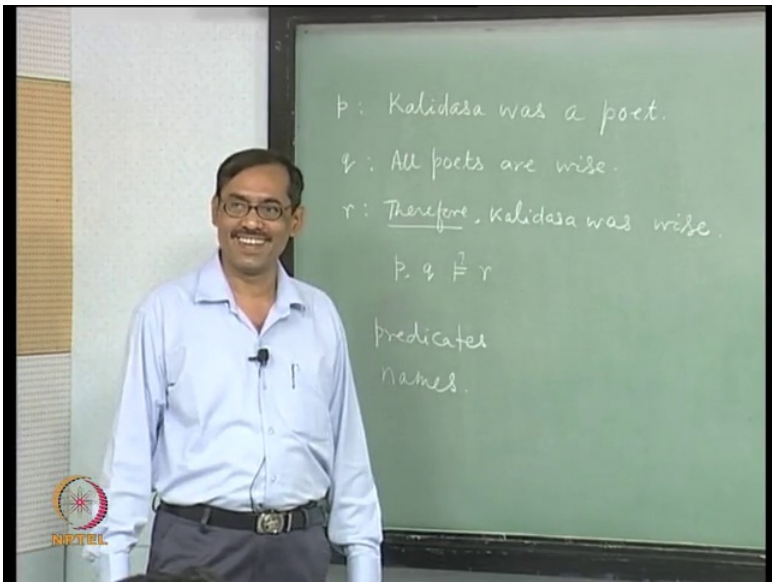
Mathematical Logic
Prof. Arindama Singh
Department of Mathematics
Indian Institute of Technology, Madras

Lecture - 22
Syntax of First Order Logic (FL)

So, we started with the propositional logic where the building blocks are the propositions. The sentences without any connectives there, and the connectives only help us to breakdown the propositions in to smaller parts, then the smaller parts are being taken to be true or false, by semantics, 1 or 0. And then we have come across proof methods also, one proof method is the axiomatic approach, which is the propositional calculus. Next we considered resolution, also we have considered, and then the analytic tabula, right? From the semantic itself we got two methods which are informal still, it could be made formal; but we did not choose to make it formal. One was calculation, another was informal proofs. Let us consider another argument; see what happens.

(Refer Slide Time: 01:13)



These you are going to accept to be true. Right now, let us symbolize it in propositional logic PL. For symbolization, first principle is find the connectives or connective looking like, that is, and then break it into smaller units. There is no connectives in the first sentence, there is no connective in the second sentence; there is no connective in the third sentence; only that therefore, but that is for the consequence relation. If you assume these two are premises, the next sentence that follows is a conclusion.

Student: a plus b and b plus c implies a equals c.

This is not implies. It is not a sentence; Kalidasa is not a sentence. In the propositional logic the building blocks are the propositions; Kalidasa is a not a proposition.

Student: Sir, we are converting english to propositions, right?

Yes, tell. Kalidasa is not proposition; it is a name; it refers to one person; it cannot be true or false, right?

Student: Kalidasa was a person who was a poet also.

As a person, that will be a sentence; so Kalidasa was a poet; that will be another person that will be another sentence, but then, for is it implies, even if the Kalidasa could have been an animal; he could have also been a poet; if it is not a person, still it is a poet. The sentence does not say, there is an implies in it; do you see the problem? At best what we can do is, I can write p for this, Kalidasa was a poet; I can write p it itself is a sentence. There is no connective in it, according to propositional logic; so I have to denote it as p, some sentence. Then all poets are wise, similarly, another sentence q, forgetting this 'therefore'; the other sentence is r, right. Now, the whole consequence, we can write as: p, q as premises, and r as a conclusion. Of course, we do not know that this consequence is correct or not, right. So, it is a consequence; the whole thing is translated to a consequence.

Now, in propositional logic this will not be a valid consequence; fine. Because you can make one interpretation which makes this true, but r false. Let us take p equal to 1, q equal to 1, r equal to 0. They are independent units, independent proposition, that is why you are trying to do it some other way, right. We are going to step ahead that; because independent propositions, and they are not really independent. Somehow they are depending, right that is why the conclusion holds true. So, the question is how are they depending, that is what we want to find out.

Let us take the first sentence: Kalidasa was a poet. It looks like there can be other poets, Omar Khayam was a poet, right. We can take many more. That means this poet, being a poet, is a property. The sentences are such that 'being a poet' is correctly applied to Kalidasa; that is the information we get from the first sentence, right.

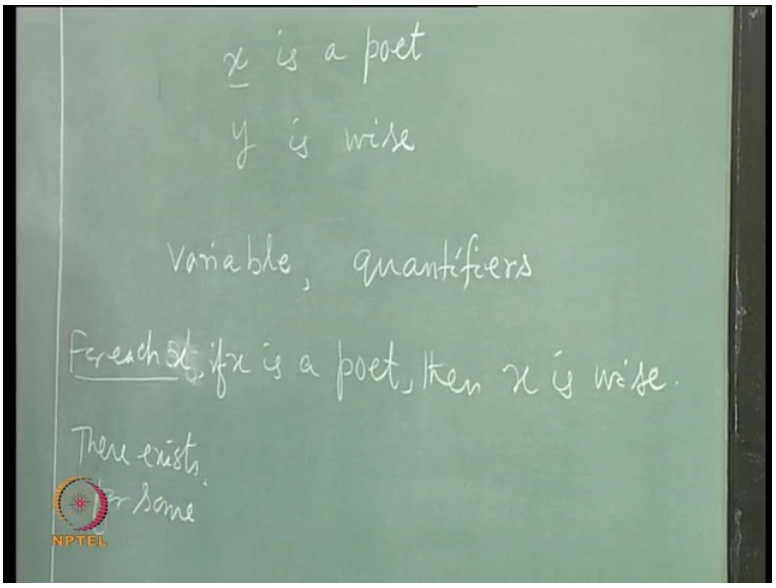
Being a poet, or 'is a poet', let us say, 'is a poet' becomes a predicate now. It is a property or relation; that property holds for Kalidasa; that may hold for Omar Khayam; that may not hold for Arindama Singh. Is it okey? It can be true or it can be false. A predicate can assert truth or falsity about some objects; that is what we see here. That means, we should have predicates, like 'is a poet'. Similarly, 'is wise', that is also another predicate, right. They are unary predicates, in the sense, one person, to one person only it applies; it has only one argument. Something or somebody is a poet, but you say, x likes y; so that 'likes' becomes a binary predicates; it is not unary now.

So, “2 lies between 1 and 3”, that becomes a ternary predicate. There are three arguments here; something lies between two other things. There are really three arguments involved; it may be true it may be false when applied to some particular objects. So, predicates can be there with different number of arguments, that is possible, we need that. Second thing here is, in the first sentence, Kalidasa is a name. It refers to some object as we understand, it refers to some person here, it is a name. You should have some name, say proper names, you just call them names, individual constants. This would also be there, right, so we need names.

Then what else we need? When you look at third sentence “Kalidasa was wise', this also requires the same thing; 'is wise is' one property or a predicate. It is a unary predicate. Then Kalidasa is a name, that is accounted for. What about the second sentence? Already we have told 'is a poet' is a predicate, 'is wise' is a predicate. This sentence connects two predicates; not a name and a predicate. Whether a predicate holds for a name or not, this sentence does not say that. It connects, now, two predicates. And it says in a particular way; it does not say all wise persons are poets, right? It says something in a particular way: all poets are wise.

So, how do we translate it? See, we think that predicates will be applicable to some objects; one particular thing, it has an argument. Now, if you say all poets are, the predicate cannot apply; predicate itself, it is not a sentence. Suppose 'is a poet'; it has vacancy so, one vacancy, we know because it is unary predicate. What is that vacancy filled in with? By Kalidasa. It becomes a sentence, right. Now, here 'is a poet', there is a vacancy, and 'is wise', there is also another vacancy. So sentences will become, once that vacancies filled in. Suppose we write it as some gap, some gap is there.

(Refer Slide Time: 09:10)



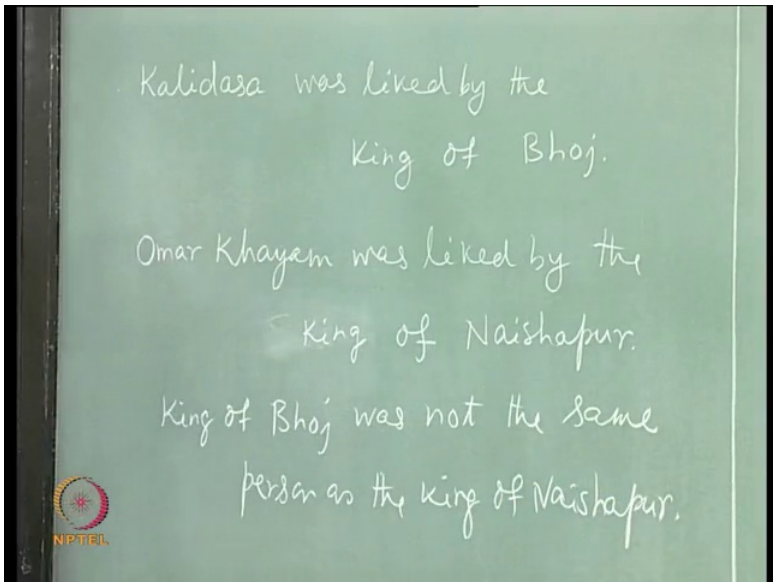
x is a poet. Now still we think of this as a gap it has not been filled in, a gap; we are just giving it some name. It is not a name really, pseudo name, it is a gap, we remember; it is a gap. Similarly, I can write “y is wise”, there is still a gap, but this gap and this gap are different. They have different names; they need not be same, unless the sentence is there, I cannot say same x can become poet and wise. They may be different completely, right. This particular thing which is a gap, but the gap has a name, right, the named gaps, we will call them as variables, just with our usual mathematical procedure. We will call this x as a variable; they are sometimes called, individual variable. So, we need variables, which are just named gaps. Once they are filled in by some particular objects we get a sentence, in a predicate.

Then how do we write this sentence? It says some implication. You are telling, right, you say, “if x is a poet then x is wise”. You do not say y here, with the same person you want to become wise, same poet should be recognized as wise. But this is not a sentence yet, because we do not know what x is; what x stands for here? It is a real number? No, all possible names. You have to write something like “for each x, if x is a poet then x is wise”. It looks alright. Now, that is what we mean by all poets are wise. You take any person or anyone now, in the poet itself, person is included; that is our interpretation. So, you take anyone or any object. If it is a poet then it has to be wise; that is what the sentence says, is that clear?

Now, look at this. We have variables, we have predicates, variables, predicates, something else we need, right. These are called the quantifiers: 'for each x' is a quantifier, for each, it is a quantifier, for each. There is another quantifier. It says, for which x we are really talking about? For all. So it quantifies over that variable. It can become something like: 'some poets are eccentric. So there is another quantifier there, it is not 'for each'; 'for some' or 'there exists at least one' or 'there is', right. You may need that also, there exists. This is another type of quantifier: there exists at least one. We will not always say 'at least one'; just say 'there exists' or 'there is', 'for some'; it has the same meaning.

In 'for each' also we sometime, say, 'for every', 'for all', but those two words are really ambiguous. Sometimes it creates problem. You say “for all men x, x has only two legs”, right. It is “all men do not have two legs”. You take all men together, count the legs; it is not two legs. It gives rise to that ambiguous all applied to everything as a whole or individual persons. In that 'for each' clarifies it. Sometimes 'every' also becomes ambiguous like that. So, 'each' is a better word. Of course, 'each' can also, you can interpret in ambiguous way. Usually it does not. 'Each' means one particular person; then look at everyone. These are the two quantifiers we will be using: 'for each' and 'there exists'. These are the quantifiers we need, fine. See, how many things we are going to need, slowly. We will take another sentence and see what happens.

(Refer Slide Time: 13:57)



Kalidasa was liked by the king of Bhoj. Now, how do you symbolize this? Or, how do you translate? A pseudo sentence like this. Kalidasa is a name, that we know. 'is liked by' , it is a ?

Student: predicate.

Predicate. What is the argument, number of arguments? Binary. Right. There are two arguments, one person liked by another person; x liked by y. So, there are two arguments. Then, next is, it is a name, 'king of Bhoj' is a name. Then 'king of Persia', that will be a different name, but they have something in common. If you want to express that there is something common in between them you have to really go a bit deeper. Right? You can take 'king of Bhoj' as a name, does not matter. It refers to a person and get away with it, right, that is okey. Suppose, there is another sentence where 'king of Persia' is also involved and now you say 'king of Bhoj' is not the same as the 'king of Persia'.

Suppose, we write another sentence; let us write this sentence. Then I will say, we want king, right, not sultan, they are same thing. Let us take king now. We say king of Bhoj was not the same person as the king of Naishapur. Now, you can take king of Bhoj as one name, it does not matter. It is a name, say, a is not equal to b that also we can do, but say, you want to go a different way. Say, 'king of' I want to express this 'king of' it is missing. That 'king of' thing, right, you may need a sentence that king is required not the jack.

Then what should you do? Is it a predicate? Is it a predicate? Suppose it is a predicate, that is why it is wrong. Suppose, you say it is predicate, 'king of' some name. You see, for example, you say it is a unary predicate. Let it be one unary predicate; here, 'is a poet'. When you say, x is a poet and you fill up that x with some particular constant, or instead of filling up you say x becomes bound to some constant, say Kalidasa, you get “Kalidasa is a poet”, which is a sentence, right.

Let us say 'king of x', there is a gap there. Suppose x is bound to Bhoj. Bhoj is an object. Then 'king of Bhoj' should be a sentence. It is not. It is again another person, right. See the problem? So, it is not a predicate; it is a different entity all together. Because once that gap is filled in you are getting an object, not a sentence. There is a difference. So, we will call them as definite descriptions; that is describing a person definitely through a

property. But that property really gives the description of the person. That person is uniquely determined by it. Right? That is why they are called definite descriptions.

Sometimes definite descriptions can look like definite descriptions, but they can be vacuous like the 'present king of France'. It is a definite description, but it gives nothing. There is nobody who is a king of France. But it is a definite description. It describes something. So, this 'describes some thing', which describes a person, when you fill in the gap you get a, get an, object or a name. Definite descriptions can be described through something having a gap, that is what we find out, like 'king of'.

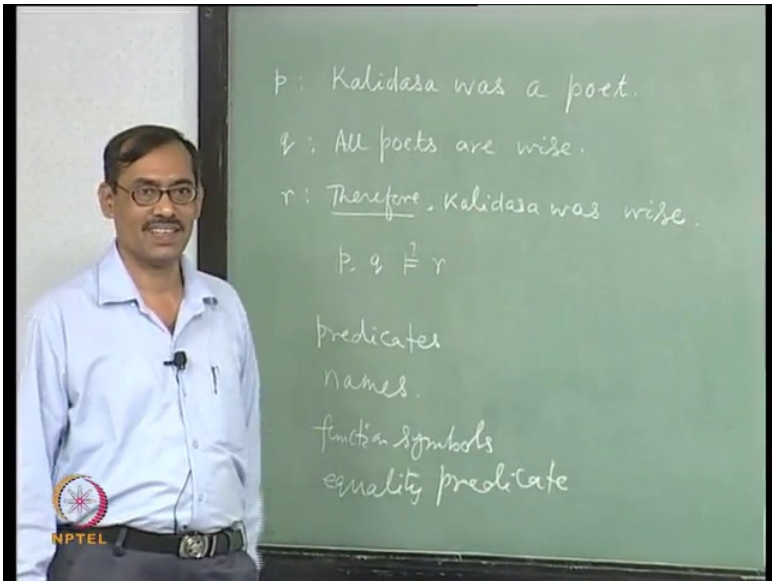
So, you will write them again through the function symbols. It is something like your function say f of x. When x is filled in by say, 1, you get another value of that say, 5, right. Suppose, it is the square function. You fill up f of x, x is filled by 2, you get 4. Again numbers, right. You will write them as function symbols, they are not functions. Because nothing is specified, no domain, no range, right. We will just tell them function symbols, just like of predicates. Similarly, if you think of function symbols as functions, you can think of predicates as relations. They relate to objects, that is why they are predicates. So, we need here function symbols.

Now, let us see another thing, say, “Kalidasa was liked by the king of Bhoj”. This 'is liked by', x is liked by y, it is a binary predicate. How do you interpret this liked by? It depends on our usual convention in English; like English words. But when you come to a formal language we will have nothing there. We will interpret, may be some other way, or explicitly tell 'liking means these ordered pairs are true'. But will have to interpret them specifically. We cannot use our ordinary common sense of liking or hating or something, but they have to be interpreted in some way or other.

When you come to the last sentence the king of Bhoj was not the same person as the king of Naishapur, 'the same as' cannot be interpreted as anything else. Otherwise there will be problem. It has to be 'that equal to'. Same as means equal to. So, 'same as' to be interpreted in one particular way, as regard to other predicates. That is how we are specifying it now itself. We have freedom to interpret it any other way also, just like any other binary predicate. But it will never give the sense; 'liked by' can be interpreted different ways.

For example, you say, author of a book. Now, 'author of' means who has actually written it, or whose name appears on the printed book, right. See, you have ways of interpreting it; whatever way you like. Like, 'father of' is physical father or social father. He might have adopted, right? So that way, we have to think of interpreting. But 'the same as', you do not want to interpret in any other way. It is particular, one, interpretation. We will preserve a different symbol for this. We will call it equality predicate, which is binary, and which has to be interpreted that way only.

(Refer Slide Time: 22:29)

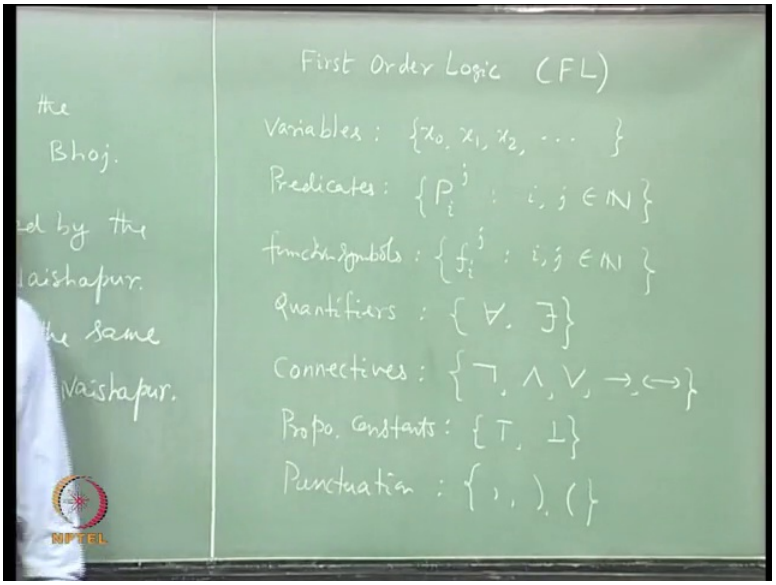


Let us fix it, identity predicate or equality predicate. See, all these things we need, now to express a simple argument like this; and some more, you want also something more. We have to get them together; we have to make them at a place, and see what are our requirements. What are the things we need? Moreover, we need also propositional logic to be inside this logic. It is a new logic we are following. It should be an extension of the propositional logic, that truth, falsity, connectives; they also should be there. Otherwise, you will be in problem, right? We cannot give this p and q implies r and so on. Let us start with that.

Then our alphabet of this, called first order logic or we will write as FL; this will have now many types of symbols, all these types of symbols; we will reserve some symbols for them. So, first we have variables which are the individual variables, say, we will put them as x0, x1, x 2, and so on. It is an infinite list, the variables, set of symbols. Then we

will have what? Predicates. We will write that Pij, why two, i and j are there? Because we also want to say what is its arity, how many arguments it can take. This superscript j will tell us how many arguments it takes.

(Refer Slide Time: 23:22)



Suppose, you want to symbolize 'is liked by' that will be one predicate with superscript 2, 'is a poet' will be one predicate written as predicate P something say 100, but superscript will be 1. It will tell how many arguments are there, right. Next, we will have the function symbols. Similarly, we will put fij, so it is there, it is the set of, so that this was, that symbol will be meaningful, set of variables, set of predicates, set of functions. Next, we need something else, say quantifiers. We have only two quantifiers: for all, there is. I read it 'for all' now because you can remember this. It is just A written different way, in this exists, E written differently; so we will read them as 'for each' and 'there exists' later. Then we have the connectives: not, and, or, implies, biconditional. And you have the propositional constants as earlier top and bottom. Anything else we need? Well. No? That depends, right? They are only named gaps, for the objects and this propositional variables are sentences, really propositions. Propositional variables are there inside it, j stands for number of arguments. Suppose there is 0 argument. So we will put a convention that they are the propositional variables, fine. If there is an argument, you substitute something in that place, it becomes a sentence. There is no argument required for it to become a sentence, right; that is the sense we are using 0 there. So, pi0s are the propositional variables, no argument is required there to become a sentence, 0. We will

make it a convention to take them as propositional variables instead of taking another class. Now, in function symbols, put j equal to 0. Again in a function symbol, if there is 1 argument, 1 vacancy is there, you fill up with a constant or name, that gives you another name, right, definite description.

So here it says nothing is required to fill in so that it becomes a name. It is a name by itself, right, so they are the constants. Function symbols with j equal to 0 are called as constants. They are called individual constants. They give you, they give you individual things, objects like your individual variables. They will be different from propositional constants, they are propositions; but individual constants are objects. One more crucial thing for unique parsing, punctuation marks, right? Punctuation marks: parenthesis, you need brackets right, so let us write punctuation. We will have 3 punctuation marks, 1 comma, sometimes f of, there are 2 things; we will write comma, right. p of two things, we will write a comma, and then we have the parenthesis, usual parenthesis; so three punctuation marks. This comma can really be removed. Sometimes we will not have the comma. Even you read it, but that requires some 'eye' to read it some 'trained eyes', fine.

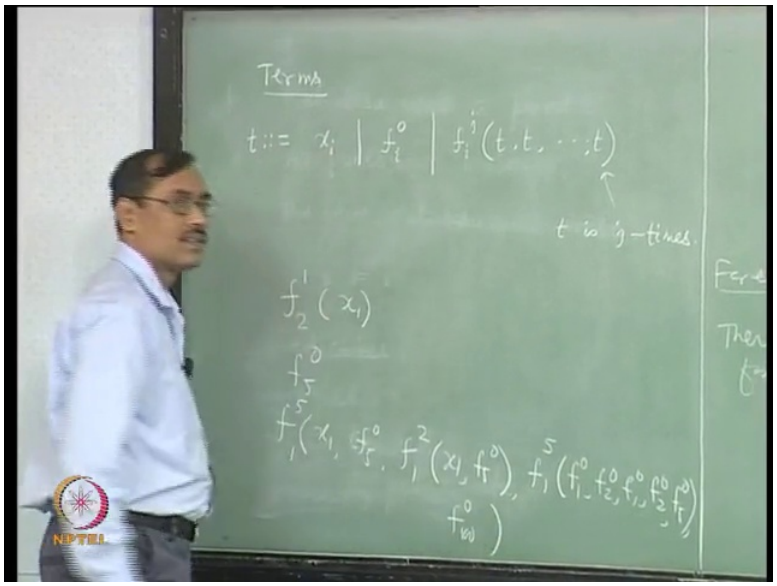
Now, you have to form the language. As you see, it will be a strictly typed language; so many types of objects are there, just like your Sanskrit or something. When you have these kind of verb or this kind of vibhakti, it will take this form, nothing else is allowed. That way it will go, like, if you have subscript, a superscript of p as 1 it will take only 1 argument, you cannot write 2. Similar things will be preserved, fine. We have to slowly develop our language from this. First thing we have to develop these kinds of things, definite descriptions. Because functions are there, so you can have a function inside a function, also as an argument of another function, right. You want to express `mother of eldest brother of Rajiv', so how do you express? 'Mother of' is a function, 'eldest brother of' is another function. Then you have to do 'mother of eldest brother of somebody'. So, a function can take another function as an argument; also it can have indefinite length, finite, but how many we do not know.

So, first you have to define that recursively. Then we go for the formulas or sentences in our language, right. Those are called the terms; we will not call them definite descriptions; their meaning will be definite descriptions. We are starting from the meaning itself how to symbolize the sentences. In our language, there is nothing, no definite description, no sentences, there is only symbols, right, expressions, strings of

symbols defined over these, union of these sets, which we have taken. Our alphabet is the union of all these, how many? Seven sets.

Then from, or, over that alphabet you take any string that is an expression. Now, this FL-expression. We have to now tell which are really ok for us, which are not ok for us, right. We have to define that; that is how we will be defining our language; just the same way as propositional logic. Connectives are there, propositions are there, punctuation mark is there, but everything is not a proposition only some certain way if it is formed, it is a proposition. Similarly, we have to define here, fine. First, we will define those, which are parallel to the definite descriptions, which will be used as definite descriptions or which will translate the definite descriptions; they are called the terms.

(Refer Slide Time: 32:05)



We define first terms. We can just give its grammar, what is a term, right. Let us write: a term can be a variable, any variable we can take, or it can be any individual constant that will look like this: with 0 on the top, or it can be any function symbol with some arguments, right, with some arguments. Means we have to again say inside that what are allowed, here is the recursion, inside that again, some function symbols can be allowed, right. We will just write t, so here t is taken j times. It is not the same t. As earlier, with our grammar convention, it can be different. They are just terms, generic terms t, or you may write t 1 to t j, where t 1 to t j are terms; again some terms, is it clear?

If you break it into rules, it will say a term can be a variable, a term can be one individual constant, or a term can be a function with arity j followed by left parenthesis, then any other term, comma, any other term, j terms like this, then another right parenthesis. That is it. Nothing else is a term. That is how we can express in 4 lines, is that clear?

For example, if you write f 2 1 of some x 1, this is a term, right? If you say f 5 0 this itself is a term; if you say f 5 1 something, now we have to give 5 variables here. So, I can take x 1, I can take f 5 0 as another, then I can take f 1 2 of say x 1, f 5 0, 2 should we need of, and then, I can have f 5 1 then f 1 0, f 2 0, f 1 0, f 2 0, f 5 0, and another; I can take f 100 0.

Now, your first job is to see that terms are really uniquely parsed. You take any term. Unambiguously, it can be found out, from which terms it has been obtained. And how it has been obtained, by using which rules, in which succession; that is what unique parsing means. And you can find out whether a term is really a term or not, a suggested term is really a term or not; so like that, if you have f 5 here, but only 4 arguments then you can say that it is not a term. So it can be parsed, only from parsing you can determinate it. If there is no unique parsing, there will be difficulty in determining it, fine. We will not prove this unique parsing. It will follow the same line as in propositional logic, proof of unique parsing in propositional logic.

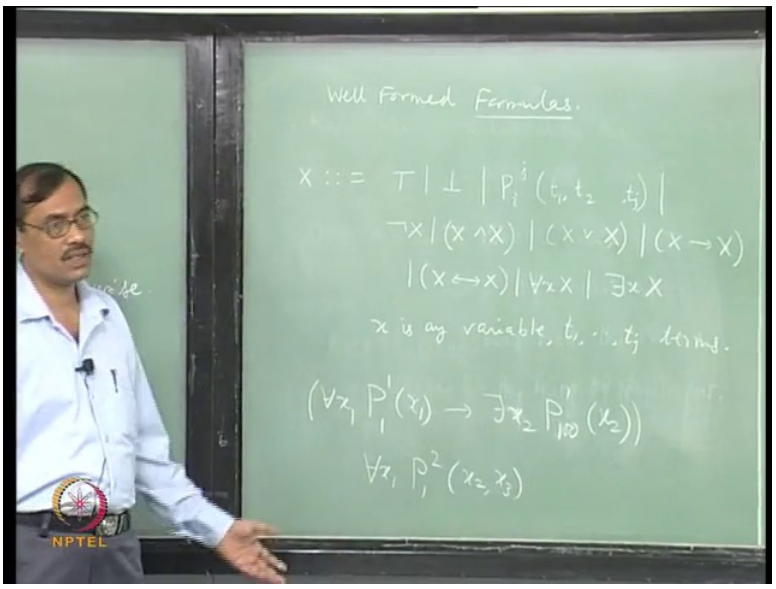
We will try to formulate our original objects, which are the sentences or how to translate the sentences into first order logic, they are called formulas instead of sentences. Even we will allow something which is not a sentence as a formula, also, because to analyze a sentence will lead some inbetween, like you have translated one sentence: for each x if x is a poet then x is wise. Now, in the absence of a quantifier it is not a sentence, it looks like x is a poet then x is a wise. It is not a sentence because x, we do not know what it is, right.

If you say f 1 0 is a poet then f 1 0 is wise. It is sentence because f 1 0 refers to one particular object, it is a constant. But if it is variable then variable can take any value, it is not a sentence, right? As usual you are accustomed with writing in your school days, x plus y whole square equal to x square plus two x y plus y square. It is not a sentence. You have to say what are these x and y's. You have to say: for every x for every y which are real numbers or something, wherever it is, you have to give this scope of a quantifier,

how much it quantifies, or say for some x for every y, something, some quantifier you need, then only it becomes a sentence, right? But then, that you need x plus y whole square equal to so and so, is meaningful; we will say them also formulas, though they are not sentences. To analyze the sentences we will need to know its meaning also, sometimes. So, we will keep them as it is; we will call them also formulas. And then we have to tell which formulas are sentences, which formulas are not sentences.

This is the approach you will be taking. That happens because, suppose you are executing a program. In the program itself you are not writing what this x is, for each x is, for each y, nothing is specified, right, in the program. But during the execution of the program the variables take some particular values; they become bound by, either by convention or explicitly in the program itself. Convention is initially if nothing is assigned, take x to be 0, depending on the language. Then inside the program it can change the values, but you see always they are some values; that is why they become meaningful, right. But before the execution, do you say that program has no meaning, no value has been assigned? They are just looking like formulas, they are not sentences, that is our stand. That means, we will take in such a way, the approach, that formulas and sentences, both of them can be taken together, and then sentences can be found out, out of the formulas which are sentences which are not.

(Refer Slide Time: 39:01)



Next one is, we will generate the formulas. They are also called well formed formulas; not malformed, or we just say formulas. We have to give a grammar for the formulas. Let us write x for that, it can be top, it can be bottom, that is the first thing. They can be taken as formulas itself or they can be as p i 0s, propositions. They are also allowed as formulas. But we take them all at a time; we finish them all at a time.

Well, now I remember one thing: inside the predicates you have the equality predicates also. We have missed it, there, so add it. Let us write a symbol, your usual equal symbol with straight line, right, is being over used; so I will use this one for the equality symbol.

Then what happens? What else can be regarded as formulas? Generalization of the sentences, which are called formulas will look like if x is a poet or if x is a poet then x is wise and so on. We start from the predicates. Let us say, P i j, inside it what is allowed, terms any terms. Now, any definite description can be a variable; will there be constants? Will be there or in general any term, right. So, there will be really j terms we can say t, t, j times, or let us write t 1, t 2 up to t j, where t 1 to t j are terms; exactly j we need. Now if j equal to 0, then nothing is there, right. So that itself is allowed as a formula. That means each proposition is also a formula. Our earlier proposition in propositional logic there, here P j 0s. They are really atomic propositions, without any connectives. Then, what else you need? Connectives we have to give; so we will write not x, not, or, x and x or y. Now, this y, we can write as x here, in the notation of the grammar, we do not need to write another; they can be different.

Anything else? They can take quantifiers, right. Let, for each x, or there is x. This x is a variable, any of the x i's. So, you have to say that x is any variable t 1 to t j terms. It is simple. There is nothing much. It says that you can form formulas by taking terms as arguments in a predicate and then use connectives, use quantifiers in the beginning, quantifiers in the beginning. But if you take quantifiers in the beginning to form first, then you can use connectives, so quantifiers need not always be appearing in the beginning. For example, you may say for each x 1 P 1 1 x 1. This is a sentence, this is a formula right. It is formed correctly because 1 is there, and 1 is there, this number can change depending on your context. Then you can say this implies there is x 2 P 100 1 x 2.

Student: variable is occurring in the formula.

Nothing is specified here, nothing is specified here. So anything you can start with. For example, this is now a formula also: for each x 1 P 1 2 x 2 x 3, this is also a formula. How you give meaning, we will see that later; now we are allowing everything. Otherwise, you have to really go into detail and see whether that is used or something else is used, not used, and so on; that will be difficult in the grammar to see, we will take care in the, when we give meaning to them, is it clear?

But if you take this as 1 it will not be a formula, right. Once you write 1, there should be 1 argument here, not 2. As it is, this is also not a formula, that is peculiarity of our grammar. We are very fussy about things now. We will not be fussy after some time. So, the thing is, there still, unique parsing holds. That is why you are doing this; just to prove unique parsing. We will again not prove it, because it is similar to the propositional logic, though it comes in two stages: now first for terms then for this.

Student: why is it called as first order?

Because, it is quantifying over the variables only; it is not quantifying over the predicates. For all properties something holds; we are not quantifying that. We are not telling that. We are telling: only for all objects something happens, which are arguments in the predicates, not about the predicate themselves; so it is called first order.

Second order logic takes care of that, where you can quantify over the predicates. For example, principle of induction. It can be properly formulated in second order logic where you have to say: for all predicates, unary predicates, this sentence. But in first order, we accept it; because that p becomes one predicate, which is generic. It becomes an induction principles scheme in our axiomatic sense. It is a scheme. Now, put any p there, it is correct; though we are not specifically telling for every p. Once you write in that form, it becomes second order. We stop here. Just we summarize. We have simply presented the syntax of first order logic. Next, we will come to semantics, that is what we want.