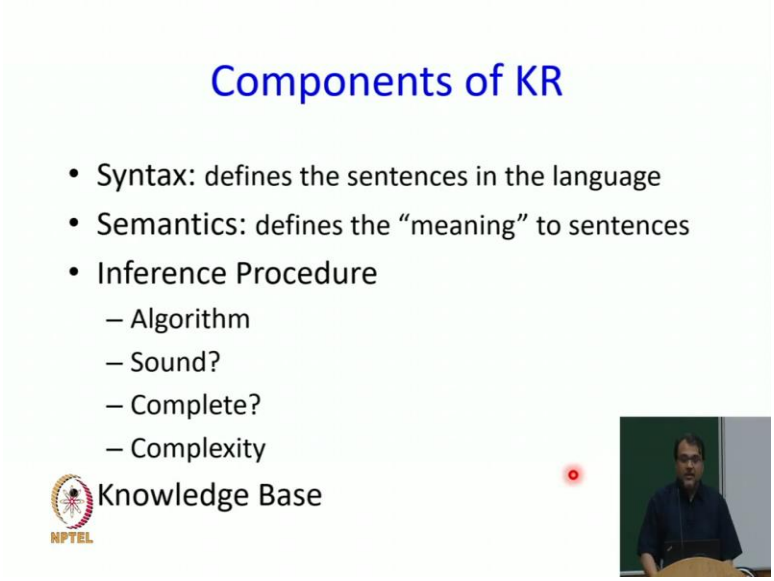
Artificial Intelligence
Prof.Masum
Department of Computer Science and Engineering
Indian Institute of Technology - Delhi

Lecture - 7
Logic In AI : Syntax, Part - 2

Every knowledge representation system like logic, probabilistic logic and so on. Have these characteristics, it has a syntax.

(Refer Slide Time: 00:31)

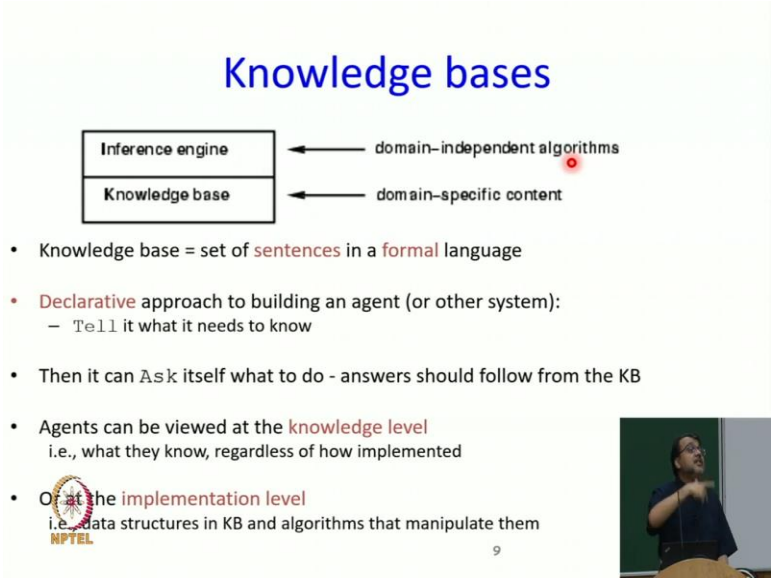


The syntax has an underlying semantics and then they all have an inference procedure which is the main reason why we are studying it we want to prove theorems we want to prove new facts we want to see which fact is in line with what you have said and so on so forth. So, they have they all have inference procedures, these are algorithms and then their; the algorithms will have some complexity.

The complexity is what is that amount of time it takes to prove a certain thing or to make a certain inference and these algorithms have 2 important properties. These properties are soundness and completeness. You would often hear the term sound and complete algorithm is a very commonly used term. So, let us understand what it means to be a sound algorithm and what it means to be complete algorithm.

We will talk about that in today is lecture. And then in addition to all these, we will have a knowledge base, this knowledge base are the set of facts associated with a particular problem or situation at hand and this will maintain all these facts, in a sentence which would be syntactically correct and will have an associated semantic meaning. So, this is how it is going to work.

(**Refer Slide Time: 01:44)**



You will have a knowledge base; this knowledge base will have a domain specific content in it. That all knowledge base will have Viswajith it is sitting in you know, chair 25 and Bianchu sitting in chair 77. Then so, therefore, you as it say the slide says knowledge base is the set of sentences written in a formal language. And where does this language comes from? It comes from the syntax of the particular underlying logical representation that you are using.

The language that you are using. This would be declarative. Declarative means that I would explicitly tell the system what it needs to know or what we know about the world so far, we would explicitly tell the system in a logical representation and all of that would be stored in its large knowledge base. Then we can ask the system for specific questions. We can say based on this, what do you know about a particular fact? And the system will be looking at its knowledge.

It will look at knowledge and make some inferences and these inferences would be made by an associated inference engine, which will have some kind of a domain independent algorithm running. It would take all these domain specific facts and run its algorithm on top of these facts

to generate new inferences based on the questions that have been asked. And so this is called the general reasoning process.

So agents can be viewed at the knowledge level, what they know regardless of how they are implemented and or at the implementation level based on the data structures in the knowledge base and the algorithms that are used to manipulate and in the previous word, there used to be expert systems, you may have heard this term, it was the rage. Once upon a time, people used to say that expert system is going to solve AI and what would they do?

They would have the experts write knowledge in a logical representation and give it to the system to make inferences once a logical representation would be about symptoms and diseases and treatments and tests. A set of doctors would say the symptom 1 is seen with disease 3, symptom 5 is seen with disease 4, disease 3 is treated by treatment 4, disease 3 is also treated by treatment 7, disease 3 is diagnosed by test 7, and so on and so forth.

Somebody will write this whole knowledge of medicine in this expert system. And then I will give a new situation to the machine. I will say that this person has symptom 3, symptom 7, symptom 8 we have done test 6 that is positive bla bla tell me, what disease does the system has this patient has and what medicine do we have to be? This was called expert systems. Everything was being done in some of the presentation of logic.

There was a question, what the question is what does domain independent mean? Good question, I did not explain this. Domain independent is a very important term. Suppose I give you this medical diagnosis system what is my domain? And what is my representation? Those are 2 different things. What is my representation, some representation of logic? What is my domain? Medicine, my domain is medicine.

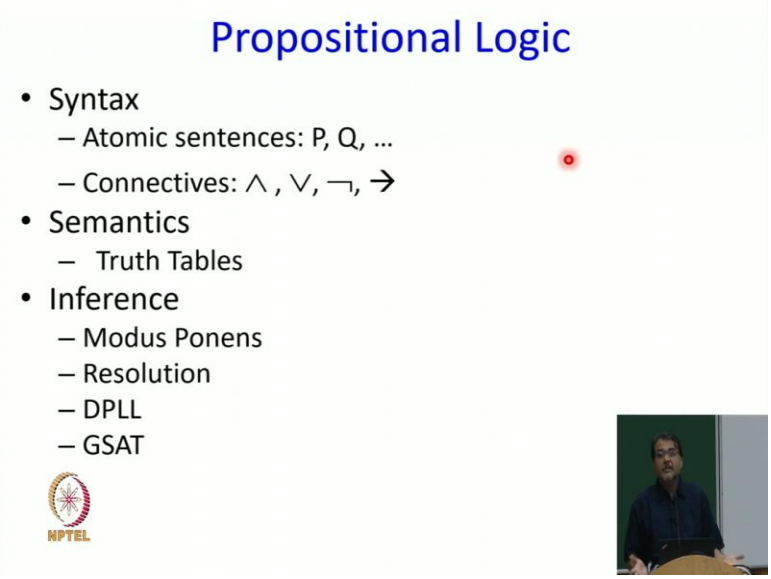
But eventually, I am only looking at it in terms of objects and relations and you know inference algorithms on top of it. Now, my domain would have been education. I have let us say mathematics, I have to teach them addition I have to teach them subtraction, but I cannot teach

them subtraction before I have taught them addition. I will teach them multiplication, but before I have taught them addition and subtraction.

I cannot use the multiplication and I have this graph of the various concepts I have which concepts should be taught earlier which concepts should be taught later then I have a new student which has already taught been taught all these things, I have a goal that I need to teach them these things and then I ask the system which topic do I teach them next. Again all of this can be written down in a logical representation.

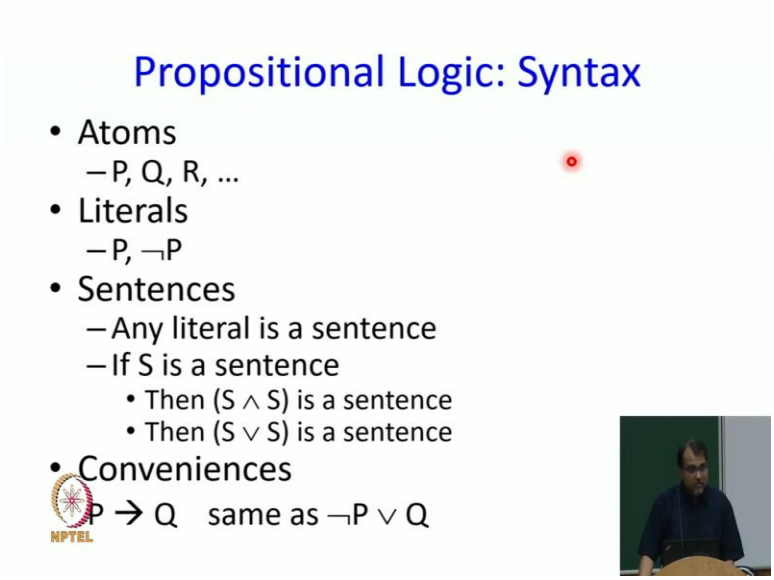
And then I can do inference to figure out which is the next thing they should be taught etcetera. So, my domains are the application domains, the particular domains of interest and my representation is a representation of logic and my inference algorithm is not about domains, it general purpose it gives it facts from medicine, it will give you medical diagnosis, you give it facts from education, it will give you education, diagnosis and so on.

(Refer Slide Time: 06:48)



Propositional logic, so, now we have talked about syntax, semantics and inference algorithms of propositional logic. This is sort of what we are going to do for the rest of these lecture slides. Let us talk about syntax first syntax is easy. So, what is the atomic syntax? atomic syntax is just facts ABCPQXYZ These are my atomic sentences.

(Refer Slide Time: 07:14)

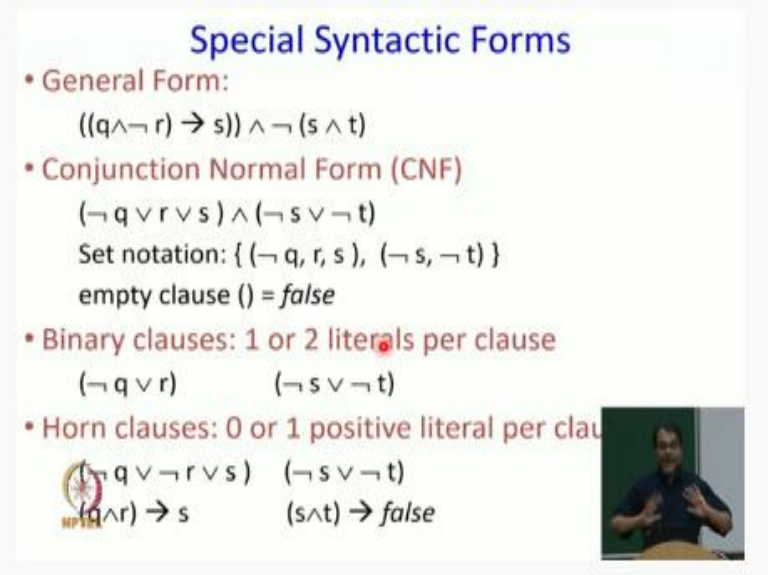


And then with each atomic sentence I have literals 2 literals, which each atom I have 2 literals. So, with P I have P and not P as my literals, P stands for P is true and not P stands for P is false. Then I have sentences and P is a sentence or if S is a sentence then S 1 has 2 sentences than S 1 and S 2 are sentences. And if as S 1 as S 2 are sentences and S 1 or S 2 are sentences, and I can do this recursively.

So, I can create P and Q not P or not Q and R, and so on and so on and so forth. These are all possible syntactically grammatically accurate sentences; I can create in propositional logic. But if I create sentences like not and all P that would be called that will litul syntactic errors syntax error. You can think of it as C++ right you have syntax X equal to equal to Y. And if you miss an equal to it has a different meaning that is semantics.

But if you put in 2 semicolons if you forget a semicolon, then that is a syntax error. So, basically what syntax is what is the space of allowed sentences I can speak in this language. And then in the case of logic, I have observed syntax syntactic sugar, like P arrow Q also means not P or Q.

(Refer Slide Time: 08:48)



Now, I have defined the syntax, but as in graphs we have special graphs, trees. In logic, we have special syntactic forms, so we have a general form like this, q and not r implies s and not r or whatever. But then there is a specific kind of forms. One is called the conjunctive normal form, or the conjunction normal form as CNF. Now, this is very important, because we know that it is a canonical form is one in which any general form can be converted into.

I mean, any sentences, any form can be converted into this canonical form at CNF notation. Everybody knows the CNF notation, so this is very important rule of logic. Have you heard this joke? Would everybody like a beer? There are 3 logician sitting in the first logician says, I do not know. The second logician says, I do not know. A third logician says yes. If you got it to the logician if you did not get it will happily move on.

So when you ask a universal quantification question, anybody who does not know has to sort of say, I do not know otherwise the answer is I mean, the answer is no otherwise the answer is yes. So who does not know what the CNF form is? Somebody does not know. So I need to explain it. So in CNF form is a form in which it said conjunction of disjunctions. Now what is a conjunction, a conjunction is and operator. What is a disjunction? It is the all operators.

So, when I have conjunction of disjunction, so whenever my statement says that many things are true, and the things that are true, are inside it, many things are disjunction. This is true or that is

true, that is true. That is called a CNF form. I also have a DNF form. disjunctive normal form, which is a disjunction of conjunctions. In whenever in 11 and 12, we used to study an SOP or a POS, SOP was sum of product.

And POS was product of sum. And we always like the sum of product. What is the sum of product in our notation? DNF why? Because at the top level I have plus, which is equivalent to disjunction here. So it is a disjunction of conjunctions whereas a POS notation is CNF. Now, we will be using CNF notation heavily. And because of that, we will define a simple set way of representing it.

And the set way of representing it is inside the set I have parentheses each parentheses is one disjunction and I will represent them by commas and then between to the parentheses I will have a comma and each parentheses is a disjunction and then all the parentheses have a conjunction between them. So, a formula not qrs and not s or not st or not t becomes a set of parentheses not q ,r, s and parentheses not s ,not t.

In fact each of these parentheses or each search disjunction is called a clause is just a term. So, we have 2 clauses in this conjunctive normal form, 1 is not q r s 1 is not s and not t not s, not t so I have 2 classes. Now, another very important question what does an empty clause mean? And what does an empty formula mean? What is empty mean? So, empty is like 0 what is the 0; for addition? What number when added does not change the value? 0.

What is the 0 of multiplication? 1. What number when multiplied to something leads to the same thing? So this is the notion of emptiness 0. This is a set theoretic phenomenon. Have you done groups, group theoretic have you done groups, rings and fields? Probably not. But if you study algebra, then you will realize that everything has a 0 every operator often has a 0 and you have to figure out 0 and the meaning of 0 is X operator 0 should be equal to X.

So, let us talk about empty clauses. Now, a clause is a disjunction what is the 0 of the disjunction of operator true or false? False. On the other hand, what is the 0 of a conjunction operator true or false? True. So, an empty clause therefore is empty clause therefore is of clauses at disjunction,

so an empty clause is false on the other hand a formula which has no clause and empty formula is true.

This will take some time to absorb. So, you know I understand these are new things for you. So, if I give you no formula, then my statement is vacuously true. But if I give you a clause but there is nothing inside the clause, then there is no way I can make this clause true and therefore, this clauses vacuously false and the whole formulas vacuously false it will take you some time to absorb this.

We will keep going however, and we will come back to this point when we use it, we will use it binary clauses those where there are only 1 or 2 literals per clause and horn clauses are very important when you have exactly 0 or 1 positive literals per clause. So, that means not q or not r or s is a horn clause, not s or not p is a horn clause, but a or b is not a horn clause because there are 2 positive clauses.

And why are they important because they actually represent a very important kind of implication relation. So, actually not q and not r or s is equivalent to q and r implies s. You can do the logic conversion q and r implies s which basically is not of q and r or s and not of q and r when you do the distribution becomes not q or not r. So, these are actually really important to study and so we will study them so this is syntax.