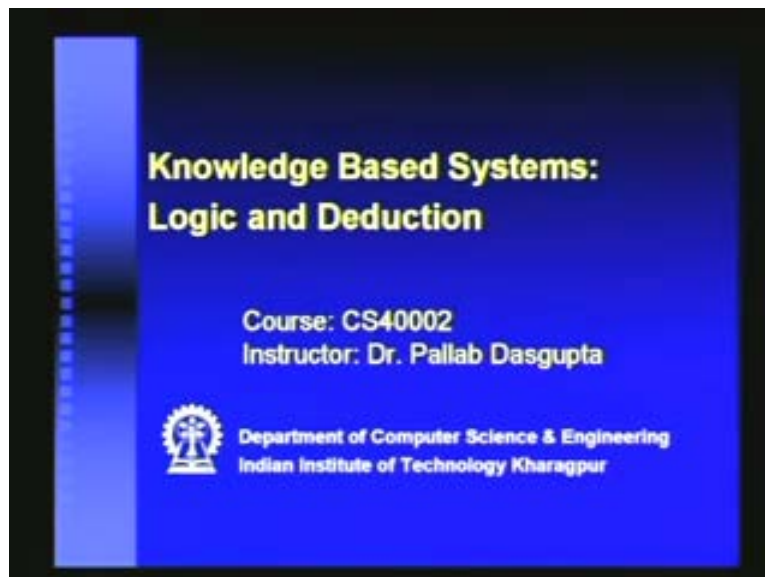


Artificial Intelligence
Prof. P. Dasgupta
Department of Computer Science & Engineering
Indian Institute of Technology, Kharagpur

Knowledge Based Systems: Logic and Deduction
Lecture - 8

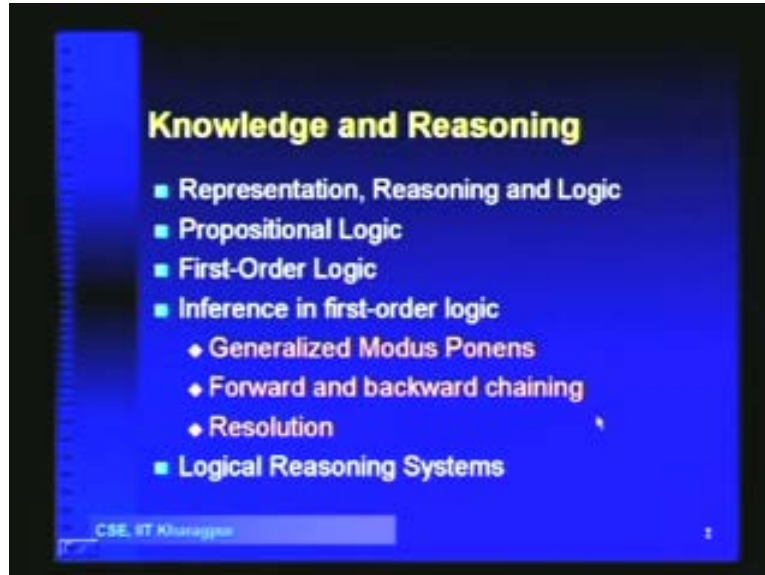
Right, so, we will now move into a completely new topic. I think we have done a lot of search. Now, we will move into a chapter on knowledge based systems: logic and deduction. This is a new thing, which we will start working at, from this thing. And we will see, that it is the combination of search and deduction, which will make up for almost **the en** the majority of the different kinds of activities or computational requirements that we have in earth.

(Refer Slide Time: 00:01:30)



So, in this, we are going to study bunch of different techniques for representing problems in logic, and also techniques for deduction, new facts and new rules, from existing ones.

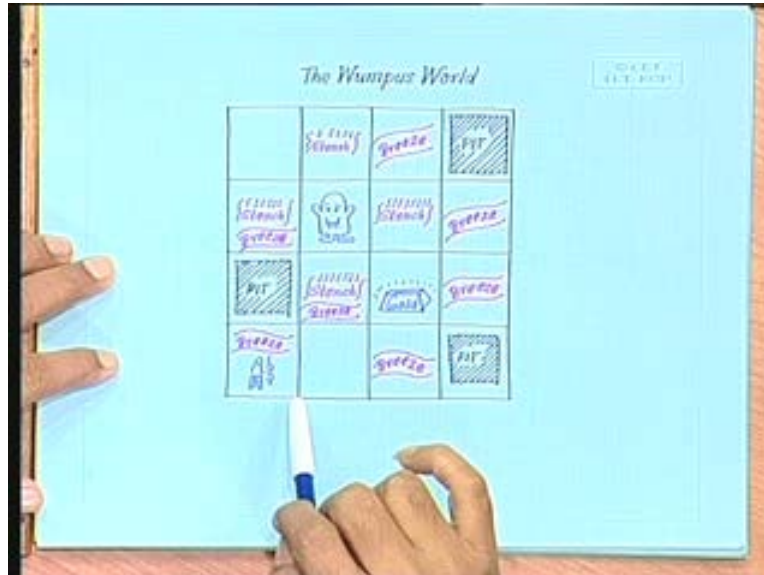
(Refer Slide Time: 00:03:43)



So, the important things are representation of real world scenarios and reasoning with logic. We will study, firstly, to start with a logic called propositional logic; then, we will see, add further things into the propositional logic, namely quantifiers, to get into first order logic; then, we will study inference mechanisms in propositional logic, as well as in first order logic. And we will see that depending on the explicability of your logic, the cost that we have to pay is in terms of the computational complexity of reasoning in those logics. For example, while finding out the satisfiability of a propositional logic formula will be shown to be- and to complete the question of first, the satisfiability in first order logic, or the validity of a formula in first order logic, will be semi-decidable.

There will be cases where the problem will be unsolvable, like you have studied the halting problem of Turing machine; there was another problem that you came across which is unsolvable, right? We will see- the inference in first order logic is also semi-decidable, in the sense that, in one direction, we can say yes, but the other direction- no answer can be un-decidable, right? So, we will look into those things. We will study some classical techniques called generalized modus ponens, for reasoning in first order logic. We will study some kind of forward and backward chaining mechanisms for inferencing; these are popularly used in all the deduction systems. And we will also study a technique called resolution, and give you an overview of logical reasoning systems that are currently used in practice. So, there, we will start with an example- we will start with a very well known classical example called the wumpus world environment.

(Refer Slide Time: 00:04:13)



So, in that environment, we have the following kind of thing: we have a grid representing a very dark cave; this cave has many chambers, and in some of these chambers, we have pits; if you fall into a pit, then that is the end of you, right? So, these are very dark, so you cannot see from outside the chamber, whether the chamber has a pit or not. But, you do have some information about the position of the pits, in the sense that you sense a breeze when you are in a pit; when you are in a chamber which is adjacent to a pit. So, for an example, we have a pit here, so you can send some breeze here, and some breeze here, right? You have a pit here, and you have some breeze in these. Diagonally adjacent ones are not adjacent; by adjacent, we just mean the ones which here, common wall, with the other chamber.

Now, these chambers also have walls, so it is not the case, that all that, you can move from any one of this chambers; from any one of them, there could be walls on some of them. For example, if there is a wall like this, then if we are in this chamber, you cannot walk directly into this chamber, right? Also, in one, somewhere in this cave, there is a wumpus- some kind of nasty creature, and if you are in the same chamber as the wumpus, then the wumpus leaves a new door. The wumpus can be on any of these chambers; it can also be on a chamber which has a pit, because the wumpus is so large that it does not fall into the pit, right? In addition, the wumpus has a stench, so if you are in any of this squares, which is adjacent to the wumpus, you will get a stench, and you will also get a stench **in the in this** in the chamber in which the wumpus exists, but if you have already entered that chamber, **then** then there is no point actually getting or not getting in that stench.

And all of this stench, breeze etc., can be sensed only when you are in the chamber where you have the stench or where you have the breeze, right? Now, the only reason for our being in this in this horrible cave, is that there is goal somewhere inside it, right? And the objective is to find the goal without falling into a pit, and without being consumed by the

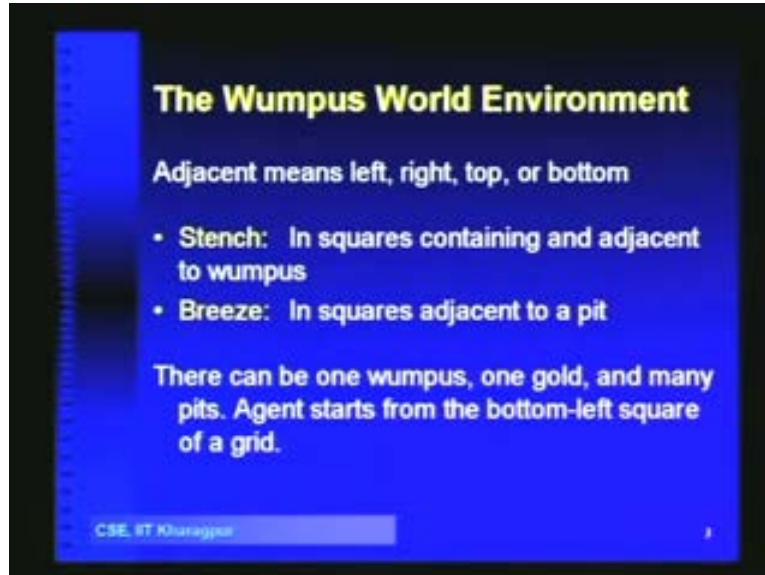
wumpus. Fortunately, in this initial formulation, the wumpus does not move around- it does not change chambers. So, wherever it is, it is going to remain there throughout the game, and we have some remedy also. We have one arrow, and we can shoot that arrow along a straight line- say, either this way, or this way; not diagonally- we can shoot it along a straight line, and if there is no wall that is blocking it, and if you find the wumpus **in line of the** in the direction in which we are shooting the arrow, then the wumpus will get killed.

And you will know that it has got killed, because it will emit terrible scream, right? So, you will know that it has got killed. So, this is the wumpus world and our objective is to find out a mechanism of getting the gold. You can shoot the wumpus, you can go from **one cave to ano** one chamber to another and you cannot see the walls, so if you bump into a wall, you will feel a bump, so, you can backtrack and try some other path. As you can see, it is also a kind of search that we want to do, but except, there is some inferencing that you have to do. If you just blindly search, then you will fall into a pit or you will go into a chamber which has a wumpus. So, as you are searching- as you are moving from chamber to chamber- your knowledge is growing.

So, you will be able to deduce that okay, in this chamber, I have a breeze, so, in some or neighboring chamber, there is a pit. Say, at this point of time, it senses a breeze here. Now, here is a problem- it knows that one of these adjacent ones has a pit, but it does not know which one. For this particular kind of example, there is no full proof- we are finding the goal, right, because it has to take one risk, at least, and if it takes the wrong move, that is the end of you, right? In many cases, the wumpus world is very unfair, but there are also certain configurations, where, as you move from square to square, you will build up your knowledge and you will start deducing that no, I found breeze there, and I found a stench there, so the wumpus could be in that square, but not in this square, so, it is safe to move from this square to that square, right?

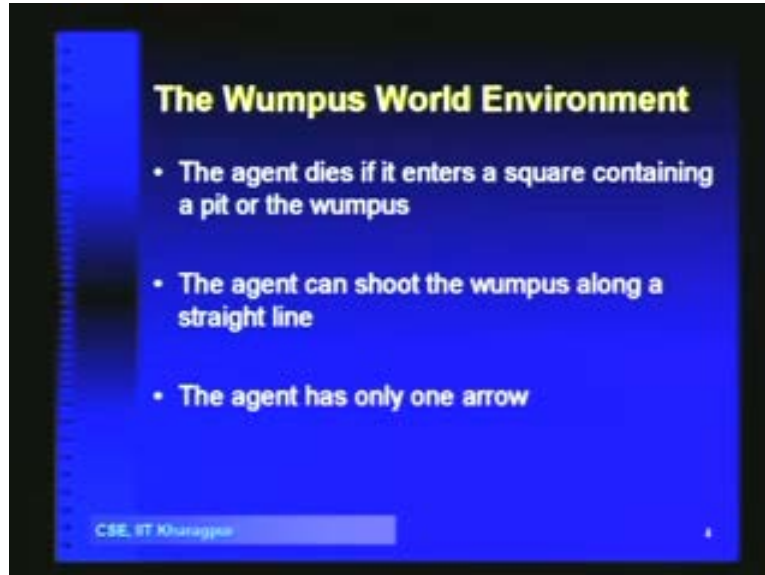
These are the deductions that will come up, so, we will have to see how we can formulate all these criterion, and all these possible information that we have, just now, told in form of a logic, right? And how we can use deduction to enhance our knowledge and eventually deduce mechanism of finding the goal? We will come back to this example later.

(Refer Slide Time: 00:10:00)

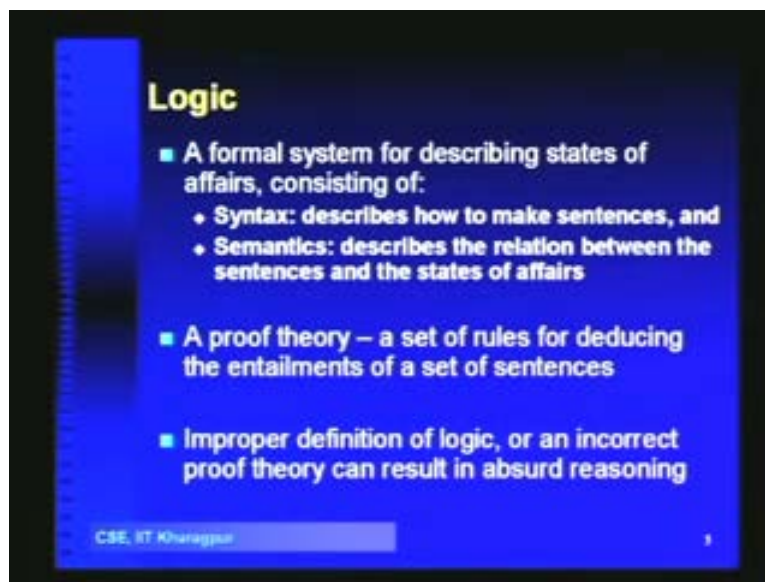


This is just what I told you just now; that adjacent means left, right, top or bottom, and then, we have discussed what is a stench, what is a breeze. There can be one wumpus, one gold and many pits. Agent starts from the bottom left square of the grid. Again, I was already mentioned all this. Agent dies, if it enters a square containing a pit, or the wumpus. Agent can shoot the wumpus along a straight line, but it has only one arrow. So, let us come to logic. What is logic? It is a formal system for describing states of affairs, and it typically- a logic will consist of 2 things: one is the syntax, which describes how to write sentences in this logic, and the semantics, which tells us what those sentences will actually mean. And then, we will have a proof theory for the logic, which will be a set of rules, which will deduce which will help us in deducing a set of sentences from another set of sentences, right? So, we will study some proof theories. If you have an improper definition of logic or an incorrect proof theory, then, you can actually have all kinds of absurd reasoning.

(Refer Slide Time: 00:10:24)

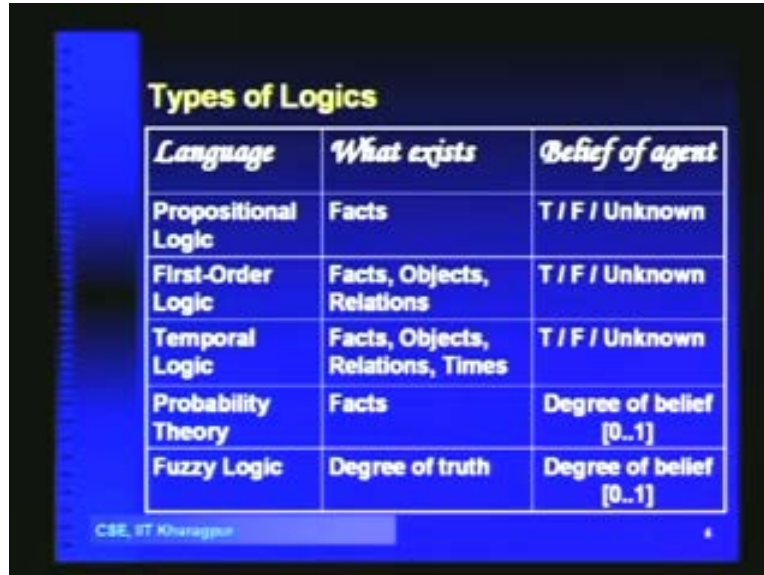


(Refer Slide Time: 00:11:21)



For example, **if you say that if** if you say that a is a cook, **right ok** **this is this um this this thing** about this thing, that in Bengali, Tagore means a chef, so, if somebody says that Rabindranath Tagore was a Tagore and Tagore means a chef, so Rabindranath Tagore was a chef, you know, that is the kind of reasoning which can give you very absurd results. **o their** You have to actually make a difference between the domains of what a Tagore can be and what a chef can be; there is an intersection, but it is not that one implies the other, right? So, we will see that for every logic, there can be certain paradoxes, and then we will see that how we can get around those paradoxes, if at all.

(Refer Slide Time: 00:12:45)



<i>Language</i>	<i>What exists</i>	<i>Belief of agent</i>
Propositional Logic	Facts	T / F / Unknown
First-Order Logic	Facts, Objects, Relations	T / F / Unknown
Temporal Logic	Facts, Objects, Relations, Times	T / F / Unknown
Probability Theory	Facts	Degree of belief [0..1]
Fuzzy Logic	Degree of truth	Degree of belief [0..1]

CSE, IIT Kharagpur

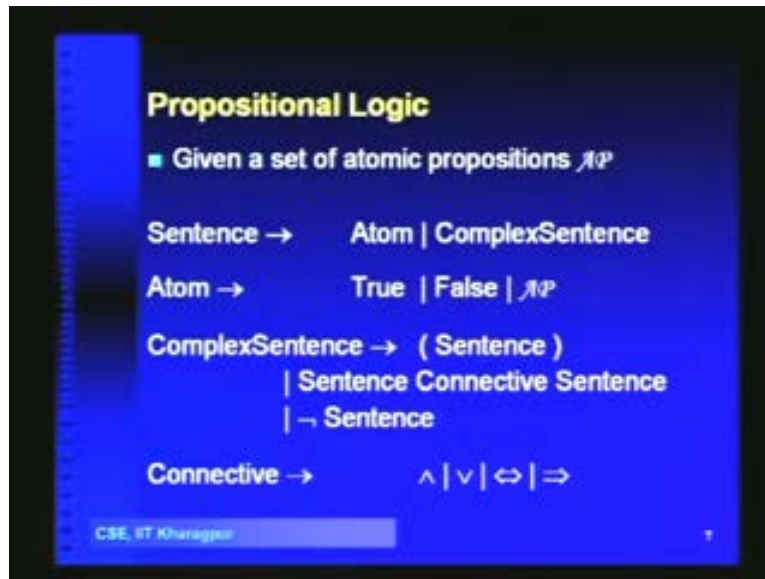
We will study several different types of logic and these logics are characterized in terms of what exists and what is the belief of the agent. So, in propositional logic, we will have facts and the agent either knows that the fact is true, either it knows that that it is false. So, proposition can be true, false or unknown. In first order logic, there can be facts, objects and relations; again, we have true, false or unknown. In temporal logic, we have notion of time, so, in addition to what we have in first order logic, we will have a special attribute called time; again, we will have true, false or unknown.

In probability theory, we will have facts, but we will have an associated probability with those facts. So, we will have degree of belief- what is the probability that this proposition is true? What is the probability that this proposition is false? And then, we have fuzzy logic, where we have degree of truth, right? And the degree of belief is again, **between** something between 0 and 1. Now, **there is a** there is a difference between probabilistic reasoning, probabilistic logic and fuzzy logic. When we are saying that- what is the probability that our new manager will be fat?- so, that is a probability, right?

We do not know whether that person is fat or not, right, we just trying to associate some probability based on the age of that person and other criteria. But in fuzzy logic, **we will the** it is not a question of whether that person is fat or not, it is the question of how fat, right? So, it is the degree of the fatness that we are talking about. There is a difference between the probability of a person being fat, **and the** and the extent to which that person is fat. So, in fuzzy logic, we will **do** deal with that kind of thing. We will say that, okay, this person is tall. How tall? So, tall is not just a Boolean, it has a gradation, right? Whereas, in propositional logic, and with logic **which** where we will deal with Bayesian probabilities, we will talk about scenarios where we do not know about the value of the proposition and associate a probability of it being true or false.

So, it is a probability of being true and probability of being false, whereas in fuzzy logic, it is how true and how false. This is our first look at proportional logic. I will first define the syntax, and then we will work out several examples to get the hang of the thing.

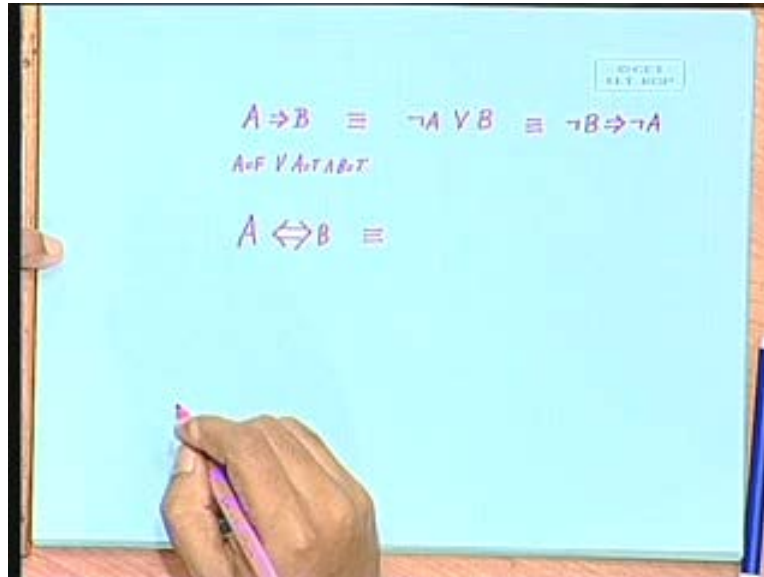
(Refer Slide Time: 00:15:42)



So, what we will have is a collection of sentences. A sentence can be of 2 types: it can be an atom, where an atom is either true or false, or a member of the set of atomic propositions. The set of atomic propositions are the basic propositions that will be true or false in our logic. For example, if we are talking about a person has passed or failed; passed is a Boolean, so passed, yes, or passed, no, right? That is a member of the set of the atomic propositions. Then, we will have sentences which can be atoms, like this, and we can also have complex sentences. Complex sentences can be sentences connected together with connectives, where connectives can be and/or, both wise implication one side implication.

If I say a implies b, it means if a, then b; whenever a is true, b has to be true, right? Whenever a is false, the sentence a implies b is backwardly true. Both wise connective means, if and only if, and we have the negation also, so you have complex sentence can be negation of another sentence. Before we go in to modus ponens, just let us look at one thing here- that when we are talking about a implies b, can you represent this in terms of the basic Boolean operators? This is equivalent to not a or b.

(Refer Slide Time: 00:19:56)



Now, please try to understand the equivalents between these 2. This is true when either a is false or a is true and b is true. What does this say- either a is false or b is true, right? In addition, if a is true, so, if a is false, then this part makes it true, right, but if a is true, then, we have to depend on this, so **it is** b is true. So, if you just construct a truth table of these 2 sides, you will see that these 2 are equivalent. How do we represent this? Not a or b, and not b or a, right, that is going to give you the x not of the 2; also, you can write the implication in another form.

So, this is the same as writing not b implies not a, right? Because this says, if a, then b, so if it is b, it must be the case that **it is not a** if it is not b, then it must be the case it is not a, because if it is not b and it is still a, then it will refute this, right? So, these are all equivalent; this is just- you know, these are some of the things which will help us later on in deducing different kinds of things. Now, let us look at the basic inference tools that we have in propositional logic.

(Refer Slide Time: 00:20:32)

The slide is titled "Inference Rules" in yellow text on a blue background. It lists two inference rules:

- Modus Ponens or Implication Elimination:
$$\frac{\alpha \Rightarrow \beta, \alpha}{\beta}$$
- Unit Resolution:
$$\frac{\alpha \vee \beta, \neg \beta}{\alpha}$$

At the bottom left, there is a small text "CSE, IIT Kharagpur" and a small number "8" at the bottom right.

We will then use this to consider some examples of deduction. The first one- okay, this is how we write an inference rule. As I was saying, **that** in a logic, you have the sentences, you have the syntax, semantics, and you also have certain inference rules, which tells you how to deduce new things from the existing one. **So on in the** What we write above this line is the premises and what we write below the line is what we can infer from that. So, **if we have a** if we have alpha implies beta and we also have alpha, then, we can deduce beta from them, right? So, alpha implies beta is a rule, and alpha is fact; that is given to us, then we can deduce beta.

Unit resolution rule is, if you have alpha or beta, and you have given not beta, then you can deduce alpha. Actually, this is written as $\alpha \vee \beta$ - please read this as alpha- it is a mistake, so you deduce alpha. Now, this and this are actually, you can think of them as similar, because, when you have alpha or beta, you can write this, actually, as not beta implies alpha, right? If you have not beta implies alpha and not beta, then, you can use modus ponens and actually arrive at alpha, right, so it is the same thing, but these are different rules, which I have come out through reason. These rules are really old; these rules have come from the time when initial logic and deduction was proposed at the time of Aristotle, right? So, as you can see, modus ponens does seem like a Latin word, right, or a Greek. **this thing** So, these are really old model, okay?

(Refer Slide Time: 00:22:31)

The slide is titled "Inference Rules" and lists "Resolution" as a key rule. It shows two equivalent forms of the resolution rule. The first form is a fraction where the top part is $\alpha, \beta, \neg\beta, \gamma$ and the bottom part is α, γ . The second form is a fraction where the top part is $\neg\alpha \rightarrow \beta, \beta \rightarrow \gamma$ and the bottom part is $\neg\alpha \rightarrow \gamma$. The two forms are separated by the word "or". Below the rules, it says "... and several other rules". At the bottom left, there is a small text "CSE, IIT Kharagpur".

When you have resolution, we have alpha or beta and not beta or gamma, then, from that, you can deduce alpha or gamma. Yes? So, given that either alpha or beta has to be true, and also given, that either gamma or not beta is always true, we can deduce that alpha or gamma is true. How? See, either beta is true or not beta is true, right? Either beta is true or not beta is true; if not beta is true, then beta is false, so alpha is to be true. On the other hand, if beta is true, then not beta is false, so gamma is to be true. In both of these cases, either alpha is to be true or gamma is to be true, right?

And these deduction is actually easy- what you do is, you take this premise and this premise, **take the and right and** then you use Boolean algebra and you will eliminate beta and not beta and you get alpha or gamma, right? If you take alpha or beta and not beta or gamma, then if you minimize it, you will get alpha or gamma; beta will get eliminated. You can write also in implication forms, where you write for alpha or beta, you write not alpha implies beta, and for not beta or gamma, you write beta implies gamma, so that is the same by using modus ponens. You can actually use transitivity here to obtain not alpha implies gamma, right? And not alpha implies gamma is the same as alpha or gamma, right? There are many other rules which we will discover, as we go further into deduction. Let us take an example on deduction.

(Refer Slide Time: 00:25:50)

The slide has a blue background with white text. The title 'Automated Reasoning' is at the top in a bold, yellow font. Below it are three bullet points, each starting with a small blue square. The first bullet point says 'If the unicorn is mythical, then it is immortal, but if it is not mythical, then it is a mortal mammal.' The second says 'If the unicorn is either immortal or a mammal, then it is horned.' The third says 'The unicorn is magical if it is horned'. Below the bullet points is a question: 'Can we prove that the unicorn is mythical? Magical? Horned?'. At the bottom left, there is a small white box containing the text 'CSE, IIT Kharagpur'. At the bottom right, there is a small white box containing the number '18'.

Automated Reasoning

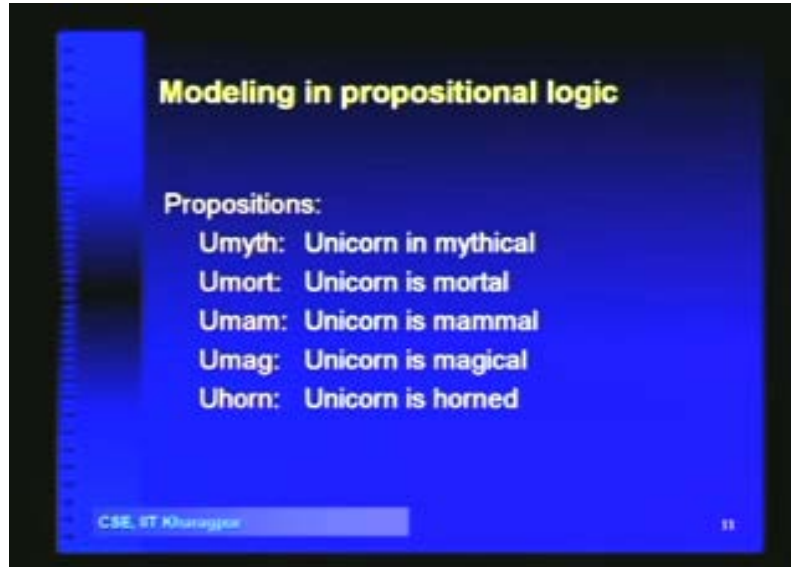
- If the unicorn is mythical, then it is immortal, but if it is not mythical, then it is a mortal mammal.
- If the unicorn is either immortal or a mammal, then it is horned.
- The unicorn is magical if it is horned

Can we prove that the unicorn is mythical?
Magical? Horned?

CSE, IIT Kharagpur 18

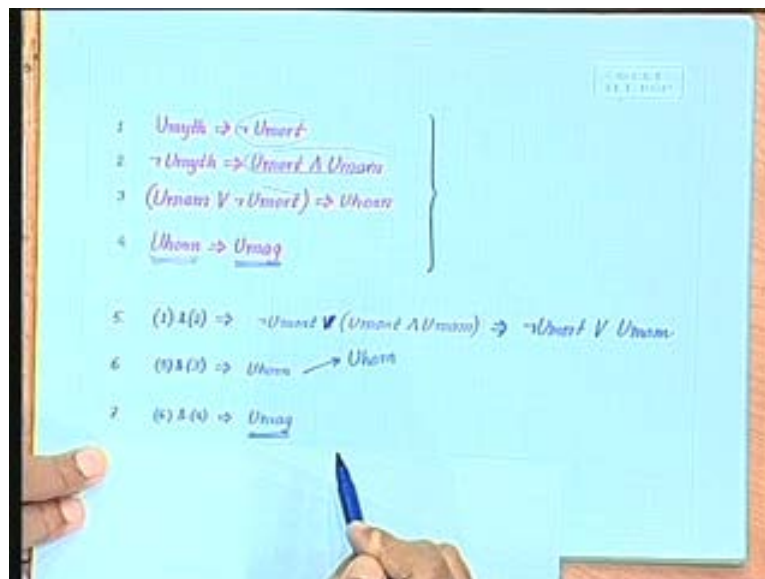
So, here is one example that we will consider- these are the basic premises that are given to us: we are given that if the unicorn is mythical, then it is immortal; but if it is not mythical, then it is a mortal mammal, right? If the unicorn is either immortal or a mammal, then it is horned. The unicorn is magical if it is horned. So, these are all the premises that are given to us. And then, can we prove that the unicorn is mythical **can we prove that it is magical** and can we prove that it is horned? The question is that, these 3 sentences that has been given to us- from that, we want to deduce whether the unicorn is mythical, magical, or horned, right? We will formulate this in propositional logic and show how we can deduce whether it is mythical, magical or horned, right? Okay. Now, the first thing that we are going to do is, that we are going to represent each of these as propositions. What are the set of proportions that **we have** we will come back to this slide? **we will come back to this slide**. Let us look at the basic proposition that we have.

(Refer Slide Time: 00:26:39)



The propositions that we have are the following: we will say umyth is unicorn is mythical umort unicorn is mortal umam unicorn is mammal, then, likewise, umag and uhorn, right? These are the 5 different propositions, along which we will yield our logic. Let us go back to each of those lines. Let us start with the first point.

(Refer Slide Time: 00:31:30)



If the unicorn is mythical, then it is immortal. We will write that as follows: if the unicorn is mythical, so, umyth implies, then, it is immortal, so, it is not mortal. So, umyth implies not umort, right? This says that if a unicorn is a mythical, then it is immortal. The second one says that if the unicorn is not mythical, then it is a mortal and it is mammal. Clear?

These 2 together is the first one- slides please- these 2 together, gives us the first thing, that if the unicorn is mythical, then it is immortal; if it is not mythical, then it is a mortal mammal.

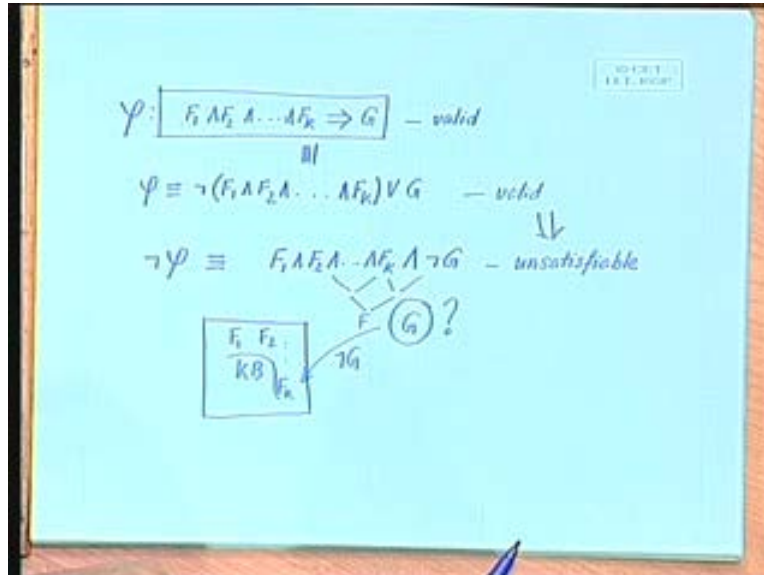
Then, when we look at the second one, it says that the if the unicorn is either immortal or a mammal, then it is horned. Let us come here and see if it is a mammal, and it is immortal or it is immortal; if it is either immortal or it is a mammal, then it is horned. And the final statement says that if it is horned, then it is magical, so, uhorn implies umag, right? This is the formulation of the set of sentences that we had written down in propositional logic. The key thing here is first, choosing the set of atomic propositions and secondly, writing them down in terms of propositional logic formulas. Then, what we do here is, we deduce new things from here.

Now, let us see what we can deduce from here. Let us look at this one first. What we get here is- from one and 2, we find that if it is mythical, then it is immortal; if it is not mythical, then it is mortal mammal. Now, one of this is always going to be true, right? So, I can say that always either this or this will be true, so either this is going to be true, or this is going to be true, right? Here, we can write from these 2, that it is either immortal or it is a mortal mammal, right? This is where we use resolution. Then, when you look at this 5, and if you look at 3, then, what we have here- **this is** this says that if it is a mammal and immortal, then it is horned.

Now, look at this thing: this says that it is either immortal or it is a mammal. So, this actually also implies that it is either umort or umam. Because we have this and we have this, so, we can deduce uhorn, right? So, we have uhorn; we have deduced uhorn. So, we have deduced that the unicorn is horned, and then if you apply this one, then, now that we know that it is horned, always, we can always we can say that it is magical. We have deduced that the unicorn is magical, so we have been able to deduce that the unicorn is horned and we have been able to deduce that the unicorn is magical.

But, we have not been able to deduce whether the unicorn is mythical or not, and as it turns out, that in this case, you will not be able to deduce whether the unicorn is mythical. Now, this analysis that we have done, has been done in a somewhat adhoc manner, so we have to bring some formalism into the procedure. If you look at this- that each of these sentences that we wrote is a Boolean formula; it is nothing but a Boolean formula over a set of Boolean variables- each of this propositions is a Boolean variable, right? So, what we are trying to find out is, if you are given a set of Boolean formulas F_1 and F_2 and f_k .

(Refer Slide Time: 00:38:06)



These are- these Boolean formulas that I have given to us, and of2 Boolean formulas, is yet another Boolean formula. This is one big Boolean formula, and we are trying to deduce whether the goal is implied by this. We are trying to deduce, that if all this Boolean formulas are given to us, and we are told that these Boolean formulas are all true, they are all valid formulas, then can we deduce the goal which is another Boolean formula? From this, can we deduce the goal from this? Can we show that this is true? We are trying to see whether this is valid, right. We will reduce this to the Boolean satisfiability problem, and how are we going to reduce this to the Boolean satisfiability problem?

If this is valid- if this whole thing has to be valid, then the negation of this has to be- it has to be unsatisfiable, right? The negation of that has to be unsatisfiable. Now, what is valid? It means that no matter what values you assign to these Boolean variables, you will always get true. A valid formula is one where whatever values you assign to the Boolean variables, this formula will evaluate to true. Satisfiable means, that there exist some assignment to the Boolean variables, which makes this formula true. There is a difference difference between satisfiability and validity. Here, we are interested in deduction, so we want to know whether, always, if these holds, then g also holds.

We are trying to deduce the validity of this thing. If we take the negation of this, then we will look for satisfiability. If we take the negation of this, remember that this is going to look like not of f1 and f2, and dot dot fk or g- this and this are the same, right? If you take the negation of this, what are we going to have? If this whole thing is what we want to prove, so, not of varphi is going to be f1 and f2, fk, and not of g, right? And if this is valid, if varphi is valid, then not of varphi is unsatisfiable, right? Then this is unsatisfiable. Now, is that clear? This is valid implies this is unsatisfiable. Clear? This gives us a nice way of systematically doing the deduction.

What we are going to do is, we are going to have a knowledge base, a knowledge base which going to contain all these f_1, f_2, \dots, f_k . These are the things that I am going to be present in this knowledge base. These are the things which are given to us; these are the facts and rules- everything expressed in terms of Boolean formulas- given to us, right? Now, you have given me a goal, and you ask that is this goal implied from this knowledge base? Can I deduce this goal from the knowledge base?

What I am going to do is, I am going to push not of g into this, and check whether this thing becomes unsatisfiable. I am going to push not of g into this; I will push not g into this and try to see whether I can prove that, then this thing will become inconsistent or unsatisfiable, right? And how can I do this? See, we have all of this in conjunctions, so, if, from any of these, by any combination, you are able to deduce false, then false ended with all of this is going to make everything false. What we will try to do basically- essentially, what we are trying to do is, we are pushing not g in and trying to arrive at a contradiction.

We are trying to take facts and rules and try to deduce a contradiction which will lead us to false, and the moment that we get false, we have shown that not g along with this going to become unsatisfiable, and therefore, by this reasoning, that we have shown here, g can be deduced from the set of fact. If this becomes unsatisfiable, then this is valid, so g can be deduced from this, clear? Right. Now, in case of **in case of** propositional logic, it is both ways. If you find that **if** you are not able to derive false from this, that means that this thing is satisfiable; if this is satisfiable, then the negation of this cannot be valid.

Now, what we have seen so far is that we can maintain a consistent knowledge base and whenever we want to deduce something, we can push the negation of that goal in, and then just call a sat solver. There are many very good sat solvers that are available, which takes a Boolean formula and deduces whether the formula is satisfiable or not. So, **you can now create** you can have a sat solver and you can create a complete reasoning system which will act as a front end. In that front end, you will be able to key in all your rules and facts, then you push the goal in, you translate this whole thing into a Boolean formula, feed it to the sat solver at the back end; if it says unsatisfiable, then you say yes, your goal has been deduced. If it says satisfiable, then you say that the goal has not been deduced, right? Okay.

But all this is based on what we call monotonic reasoning. You have to ensure that at no point of time the truth of your propositions change. Later on, we will see cases of non monotonic reasoning, where the truth of this propositions can change with new facts being introducing, right? So, that is going to make things more complex; we are not going to look at that right now. So, now, to some results.

(Refer Slide Time: 00:41:35)

Automated Reasoning

- In general, the inference problem is NP-complete (Cook's Theorem)
- If we restrict ourselves to Horn sentences, then repeated use of Modus Ponens gives us a polytime procedure. Horn sentences are of the form:

$$P_1 \wedge P_2 \wedge \dots \wedge P_n \Rightarrow Q$$

CSE, IIT Kharagpur 13

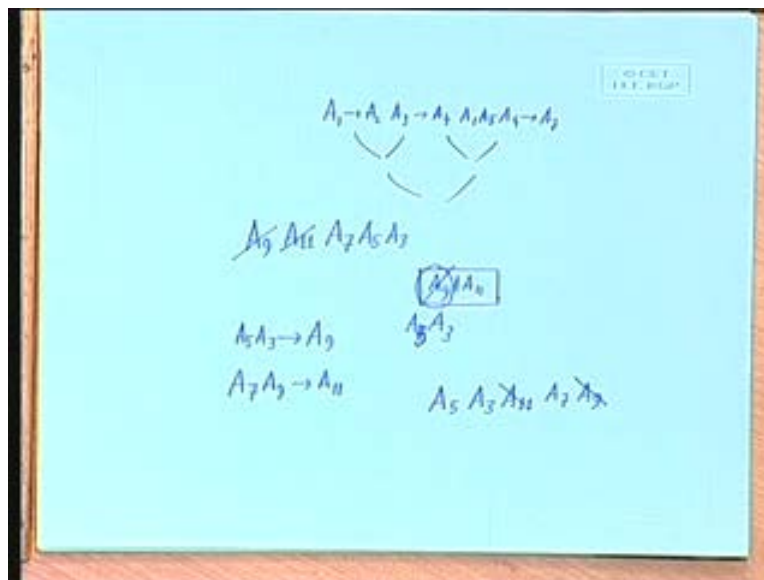
So, in general, the inference problem is np complete cooks theorem- we have all studied that in algorithms, isn't it? Yes or no? So, you are familiar with what is np complete? If not, then, please brush it up. So, Boolean satisfiability is an np complete problem. But in some cases, we can have polynomial time procedures for doing the deduction. So, if we restrict ourselves to horn sentences, where horn sentences are of the form like this, then we can use modus ponens repeatedly, to give us a polynomial time procedure. Now, what is missing here? There are 2 things that we are not allowing in horn sentences, one of them is- we are not allowing or's is here, and we are not allowing or's here- no or's in the left hand side, no or's in the right hand side, right?

It is the It is the ors that will create problems. Now, this is a- just think of this as an exercise to find out, that why is it the case, that when we have horn sentences, it makes life easier for us? Pleasure is when we have sentences where you can have ors also on the left hand side. If I also allow p one or p 2 implies q, then it suddenly becomes difficult. Why so? Yes, so you will not be able to directly use modus ponens to give you a definite answer to whether something is true or false. For example, if you have this thing implies q or r; if you have something like this thing implies q or r here, then, when you have this thing in your knowledge base, you still do not know q is there or r is there, right?

Intuitively, that is what brings in the problem. But, note that it is not the case, that this thing does not have or at all in in the Boolean sense, because if you represent this in the implication, if you eliminate the implication, then this is going to be not of all this or q, so r is there, but it is there in such a way that that you can still use modus ponens repeatedly to derive that. Now, there are 2 ways of broadly of doing this: one is that we start from the set of facts and rules that we have; we start from the set of facts that and rules that we have, and suppose that all these rules are given to us in the form of modus ponens. So, we can see whether the left hand side is satisfied by the knowledge base and if so, then the right hand side is deduced in the modus ponens.

If an in an implication, the left hand side is satisfied by the knowledge base, then the right hand side can be deduced. So, you add the right hand side to the knowledge base, and then now check whether any of the other rules can be applied, right? If a new rule now becomes applicable, then again, deduce the right hand side of that, put it into the fact base and keep on doing this, until at some point of time, you have reached the goal, right? This is called forward chaining; this is called forward chaining, where I am starting from the existing set of rules and I am trying to move forward. So, what I am I am look trying to explicitly deduce the goal. So, I have all my implicational rules, right so which says A1 implies A2 and A3 implies A4, and so on, and A1, A5, A4, implies A7, and so on. And then, I am just using these to deduce newer and newer facts and putting it in the knowledge base.

(Refer Slide Time: 00:47:55)



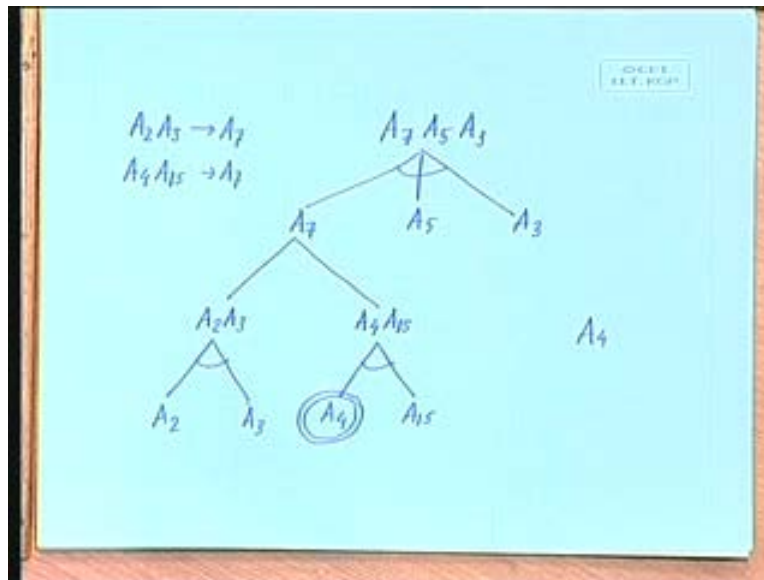
And I am not looking that I want to deduce, let us say, A9 and A11; I am not starting from this and working backwards; I am starting from these and working forward, so it is forward chaining. The other option is to do backward chaining, where we start from A9 and A11, and then examine the rules and see which of the rules has A9 and A11 on the right side, which are the rules that can give us A9 and A11 on the right side.

So, let us say that we find, that there is a rule which says A5, A3, gives me A9 and A7, and A9 gives me A11, right? Now, here, we try to see whether any of these are being implied by this. So, we find that yes, A9 is coming from here, right? So now, by backward chaining, I can replace this goal; I can replace this goal by A7 and A5 and A3A5 and A3. My new goal becomes A5, A3, A11; then, I look at A11- okay, I can get A11 if I have A7, a 9, right? So, I replace A11 by A7, a 9, then, again, I can apply this rule to eliminate A9 and I have A5, A3, right? So, I can eliminate this. Now see, this is an extra step that I have to do. If I had actually In the initials configuration, when I had a 9, A11; if I had applied this one first, then this would have got reduced to A7 and

A9 is already there, and then, by another application of this rule, I would have got A7, A5, A3, right?

And then, we can go on backwards like this, until we arrive at the set of facts that are there in the knowledge base. And when we have the set of facts in the knowledge base, then we say that that is solved, right? Now, do you see some similarity, of what is happening here, with problem deduction search? This is what we have to get, so this is actually **an and of** A7, A5, A3, so I have to get A7, I have to get A5, I have to get A3. In order to get A7, some rule might tell me that you might have 2 rules for A7.

(Refer Slide Time: 00:50:05)



So, suppose I have to get A7, A5, A3- so, this is an end of A7, A5, A3, that I have to get. And then, there are 2 rules: one which says that A2 and A3 gives me A7, and another rule, which says that A4 and a fifteen gives me A7. Now, here, I have an or node: one for this, one for this; if I am here, then I have to- if I choose this rule, then I will have to solve A2, A3. If I choose this rule, then I will have to solve A4, A15. What do I mean by solve? I have to see whether these can be deduced from the knowledge base. So, again, this is an and of A2, A3, and this is an and of A4 and A15, and this is actually a graph because you see this A3 and this A3 are the same node, right? And then, again, we see what rules can give us A3.

We go backwards, until we hit upon the set of basic facts that are given to us. Suppose it is given to us, that A4 is a fact, A4 is always true; that means that this is a node which is solved, right? **So there are** So, we now have- as discussed, we have 2 strategies: one is forward chaining, where we start from the beginning and start generating new facts and rules; new facts, rather, and putting it in knowledge base and keep on applying the set of rules, and, we are backward chaining, which does the thing from the goal, backwards, right? Now, try to think of the scenarios where forward chaining will be useful and where backward chaining will be useful.

There are deduction systems which apply forward chaining; there are also the deductions systems which apply backward chaining. Think in terms of the branching that is there and also the depth of the deduction that you may have to do. Yes. So, **the gain the actual search so** you see, this deduction is actually- it involves search, it involves searching in this rule space to determine which rules to apply in what sequence, to be able to derive the thing, right? So, **we are going to um** we are not going to explicitly go into the exact search procedures for the deduction, because you already now have some background on search. So, you can figure out, that what kind of search **will be able** will be able to do deduction.

But interestingly, there are actually platforms for programming, where this kind of backward chaining is built in. So, you have studied programming languages c pl, c plus plus, etc., which are procedural programming languages. In this course, we will study a language called prolog, which is a logic programming language. In this logic programming language, you are just specifying your facts and rules, in the form that we mentioned in implicational form, right? And this mechanism of backward chaining is build into the interpreter of the language. So, you will see that computation will be viewed in an entirely different perspective, when you talk about logic programming .

And, from a programming point of view, it is very interesting to see how to take program which you normally implement in procedural language, and see how you can solve that in logic programming. It is not the case, that logic programming is going to give you any additional computational advantage; it is only the representational advantage. You can write the program for 8 queens in 5 lines; we can write the program for TSP in 5- 6 lines. You know that is where the advantage lies- you can write it in a very crisp form and this mechanism of computation in backward chaining is built into the interpreter. When you study prolog, we will go into that.

So, we conclude this lecture here- we will start off in the next lecture on first order logic; that is where we will talk about quantification and see what are the advantages there.