

Artificial Intelligence

Clause Form and The Resolution Rule

Prof. Deepak Khemani

Department of Computer Science and Engineering

Indian Institute of Technology, Madras

Module - 07

Lecture - 03

Okay so we are looking at resolution method. And we had said that we came to the resolution method because of the fact that forward chaining and backward chaining were not complete. Now a resolution rule if you keep applying repeatedly is not complete in general. To give you a simple example so what is the notion of completeness that if something is entailed then they must be derived. So to give you an example if you have some random database in which there is only one statement P it could be first order logic it could be in propositional logic it doesn't matter and this entails the statement $S \text{ OR } \text{NOT } S$ which is not surprising because $S \text{ OR } \text{NOT } S$ is a tautology in any language so it must be true. Infact you don't even need that P to be true. But as you can see that there is no way of deriving $S \text{ OR } \text{NOT } S$ from P in the sense that you cannot derive.

Or in other words from this statement P you cannot derive $S \text{ OR } \text{NOT } S$. in that sense it is not complete but resolution refutation is complete. And so what do we mean by this that if the input is unsatisfiable then one can drive the empty clause. Which in our language we can say if a set of formulas S entails an empty clause it means it is unsatisfiable. Anticlauses stands for false if you can derive false that means your original set must be unsatisfiable or contradiction. So if this then you can and this was shown by Robinson when he actually proposed the method the refutation is unsatisfiable that if you want to show that something is the refutation is complete that if you want to show that a set of formulas is unsatisfiable then there would always exist a proof that you can find. So you can derive the null clause.

(Refer Slide Time 4:16)

Resolution - not complete in general $\{P\} \not\vdash (S \vee \neg S)$ but $\{P\} \not\vdash (S \vee \neg S)$
but $\{P\} \not\vdash (S \vee \neg S)$
Resolution Refutation is complete
↳ if the input is UNSATISFIABLE
then one can derive the EMPTY CLAUSE
If $S \not\vdash []$ then $S \not\vdash []$


NPTEL

Now of course this doesn't create any particular difficulty for us because in general if a set of clauses S entails a formula α and which is what you want to show that α is a consequence of a set of clauses S then if you take the union of S and the negation of α then this is empty. So remember that entailment basically stand for a tautology of some kind that S implies α is true so a simple way to make it unsatisfiable is to take the negation of what you want to show which is of course we know it as proof by contradiction add it to the set of clauses and then entailment follows that the empty clause is listed. And because we have said that this is true this is the definition of completeness. We have already spoken about soundness so in fact you can replace this if then statement with an if and only if statement that you can derive the null clause if and only if the original set is a contradiction.

So its not a problem to work with refutation method you can easily adapt that. So we are talking of FOL in general so resolution refutation so we already see that the choice of your proof methods determines if it is complete or not. so with resolution refutation it is sound and complete. There is also a third property we are interested in that will our algorithms terminate that's called decidability and it turns out that for all kinds of methods FOL is it is semi decidable. What do we mean by this. That if the input is unsatisfiable we can derive the null clause, so for example one could use something like breadth first search but if the input is satisfiable could loop forever. Let me give you an example.

(Refer Slide Time: 8:32)

Resolution - not complete in general $\{P\} \models (S \vee S)$ but $\{P\} \not\models (S \vee S)$
but $\{P\} \not\models (S \vee S)$ but cannot derive $(S \vee S)$
Resolution Refutation is complete
↳ if the input is UNSATISFIABLE then one can derive the EMPTY CLAUSE
If $S \models C$ then $S \vdash C$
————— COMPLETENESS
In general If $S \models C$ then $S \cup \{\neg C\} \models \perp$
FOL - with Resolution Refutation - Sound & Complete
it is only semi-decidable
If input is unsatisfiable we can derive the null clause
one could use Breadth First Search
Satisfiable \rightarrow could loop for ever.



And we have seen in backward chaining for example that you could write rules which could go into infinite loops. So just look at this rule this says that if let me first write it in logic. That if the successor of s remember s is a successor function. If the successor of s is less than y then s is less than y and we can accept that this is a true statement. So which if we write in clause form we will write it as less than let me use question mark here. I convert P implies Q into not P or Q and write it as a clause. Now supposing that our goal is that is zero less than zero.

So first of all when we say that's the goal the basic question we are asking is is this entailed by the knowledge base and our knowledge base has only one sentence. And you can see that this will not entail. So we negate it this is our strategy for refutation and add the negation of the clause which is. You can see that if I resolve these two clauses I will cancel this not less 0 0 by less than x y by saying x is equal to 0 and y is equal 0 and I will get a new clause successor of 0 which I will write as 1 or lets just write it as 0 to avoid confusion. Now we can match this with the same clauses we have only two clauses to start with and we can produce a new formula. And this process can continue indefinitely.

So with just these two statements we can get into an infinite loop we can keep resolving this whole process to generate new clauses which can be resolved with the first clause again and again. So this goes into a loop.

(Refer Slide Time: 14:03)

Semi decidability of FOL

Example: $\forall x \forall y \text{ LessThan}(s(x), y) \supset \text{LessThan}(x, y)$ *Successor*

$\text{LessThan}(?x, ?y), \neg \text{LessThan}(s(?x), ?y)$ *Goal: LessThan(0,0)?*
↓ negate

$\neg \text{LessThan}(0, 0)$ ←

$\neg \text{LessThan}(s(0), 0)$

$\neg \text{LessThan}(s(s(0)), 0)$ *goes into a loop!*

$\neg \text{LessThan}(s(s(s(0))), 0)$

NPTEL

Okay but there are still some nice things we can do we see that so it's semidecidable in the sense that what you are trying to show is indeed a consequence of your knowledge base then you will terminate in a finite amount of steps and then we have also mentioned that we can for existential queries we can do something like answer extraction. So we had seen for example if you know that all men are mortal if you ask a query is there someone who is mortal then you could have come up with answer that yes depending on what you database has whoever is listed as a man you can infer is mortal.

So some people have suggested that we can use an answer predicate to explicitly extract an answer so let me take that example of the plus thing that we had developed earlier. So remember that the clauses for addition one of them was that if x plus y gives you z then the successor of x added to y gives you the successor of z and this is something we have seen. Ofcourse we had the base clause we will see that in a moment. Now supposing our goal is to ask whether this formula is true $\exists R$ and if you just follow the procedure you negate it and add its negation here which is negation of plus 2 1 and we add a clause which we call as answer and we put in the variable that we are interested in there. So this is the answer.

And we ofcourse need to modify our termination criteria that if earlier we used to terminate when we derived the null clause now we will terminate if our clause contains only the answer.

So we will go through our process again to resolve this with this. And what do we get not plus this time I will just use natural numbers 3 R1 or. Remember that the unifier for this will have R is equal to successor of R1 and ofcourse x is equal to 2 and y is equal to 3. Sorry x is equal to 1 and y is equal to 3. From this and again from this we will get not plus 0 3 R1 OR. Now this will resolve with the second statement given in our knowledge base which says that plus 0 z z . okay so we wont get the null clause this time but what we will get is only this answer successor successor I will just write 3 here.

(Refer Slide Time: 19:48)

Existential Query - Answer extraction. use an answer predicate

Example (addition)

$\neg \text{Plus}(\text{?X}, \text{?Y}, \text{?Z}), \text{Plus}(\text{S}(\text{?X}), \text{?Y}, \text{S}(\text{?Z}))$
 $\text{Plus}(0, \text{?Z}, \text{?Z})$

$\text{?R} = \text{S}(\text{?R})$
 $\neg \text{Plus}(2, 3, \text{?R})$ Goal Plus(2, 3, ?R)?
 $\vee \text{Answer}(\text{?R})$ 'negate' - answer predicate

$\neg \text{Plus}(1, 3, \text{?R1}) \vee \text{Answer}(\text{S}(\text{?R1}))$

$\neg \text{Plus}(0, 3, \text{?R2}) \vee \text{Answer}(\text{S}(\text{S}(\text{?R2})))$

$\text{Answer}(\text{S}(\text{S}(3)))$

NPTEL

so we will not terminate here. So at termination the answer predicate tells you what is the answer. What was the question we asked we asked what is 2 plus 3 and this answer predicate says that the answer is 5. So successor of successor of 3 is 5.

Remember the example Answer we had considered in which our KB was the following onTable a onTable b green a or green b. so we have these four clauses. And if our query is does there exist something which is on the table and green and we had shown that resolution method will allow you to do this. Now what will we do we will convert this into clauses so ontable We will negate and convert it into clause form. so what will we get now. So what are the clauses this is one clause. Then ontable a is another clause ontable b is third clause and green a or green b Is fourth clause. Andwe have shown that from these four clauses this is the negation of the

goal that we have we can derive the null clause. What do you expect you will terminate with. What will be the answer to this question. So what is the question the question is that a is on the table b is on the table and atleast one of them is green then we are asking is there something on the table which is green.

The answer is yes that we have shown already we can derive the null clause but now we have added the answer predicate to this. What will that contain. I will leave this as an exercise for you but you should verify that at termination you will have the following.

(Refer Slide Time: 23:30)

Answer predicate

$$KB = \{ onTable(a), onTable(b), green(a) \vee green(b) \}$$

Query : $\exists x [onTable(x) \wedge green(x)]$

↓ negate

$$\neg onTable(?x) \vee \neg green(x) \vee Answer(x)$$

$onTable(a)$
 $onTable(b)$
 $green(a) \vee green(b)$

Exercise

?

$Answer(a) \vee Answer(b)$

NPTEL

That if you did the same resolution steps and carried forward the answer predicate you will get the answer that obviously you can see from the knowledge base you cannot say whether a is the answer or b is the answer because one of them is the answer. And infact the resolution method will to this answer predicate that either a is the answer or b is the answer. Sometimes you have to be a little careful. So let me take yet another example lets say that we are analyzing some detective story or something like that and whats given to us is the following set of clauses. I will just write the clauses and I will ask you to reinterpret them as first order logic statements. So I will use the predicate A predicate C to stand for culprit because I want to keep writing it over and over again I don't want to spend too much time writing that.

So this is given to us so without using real names we will use the names a and b. so this statements says that either a is a culprit or b is a culprit that is known to us that is given to us. You could have written this statement as follows. For all x you could have written this as well and basically two of them are equivalent. And we have seen that one of the problems in first order logic is how to express things. So this is one question that one should look at. And then we have some more clauses there is a clause that says that I will just use uppercase in prolog style to stand for the variable here. So if you read it in first order logic you should really read it as culprits wear red shirts. Or for all x if x is a culprit then the color of x's shirt is red.

So remember we had talked about properties and there are different ways of writing color and so on so I will chosen this particular set of predicate. As an exercise you should convert to triple. But I will leave that as an exercise for you. There are different ways of expressing the same fact, so anyway we are told that culprits wear red shirts and we are also told that one of the two culprits that we have a was actually wearing a blue shirt. And then we have some general knowledge about colors that colors are different which we can express as follows.

So if you go and sort this a little bit carefully what I am saying is that either color c1 is equal to color c2 or they are different you cannot have u for some object shirt lets say shirt wearing both red shirt and blue shirt at same time. One of them must be atleast false. So this is basically saying that colors are distinct and I have in my knowledge base certain statements like red is not equal to green and blue is not equal to red and all kinds of such statements. I just need blue is not equal to red so I am not writing the others here. And this is the knowledge base given to us this is one clause this is another clause this is the third clause which says that a was wearing a blue color. This is the clause which says that colors are different that you cannot wear the color of your whatever you are wearing is unique. And this is the clause which says red is not equal to green and blue is not equal to red.

(Refer Slide Time: 29:25)

Example


Culprit $C(a) \vee C(b)$ $\exists x \text{ Culprit}(x) \supset (x=a) \vee (x=b)$

$\neg C(x) \vee \text{Color}(x, \text{shirt}, \text{red})$
 Exercise: Convert to triples

$\text{Color}(a, \text{shirt}, \text{blue})$

$\neg \text{Color}(U, V, C1) \vee C1=C2 \vee \neg \text{Color}(U, V, C2)$

$\text{red} \neq \text{green}$ $\text{blue} \neq \text{red}$



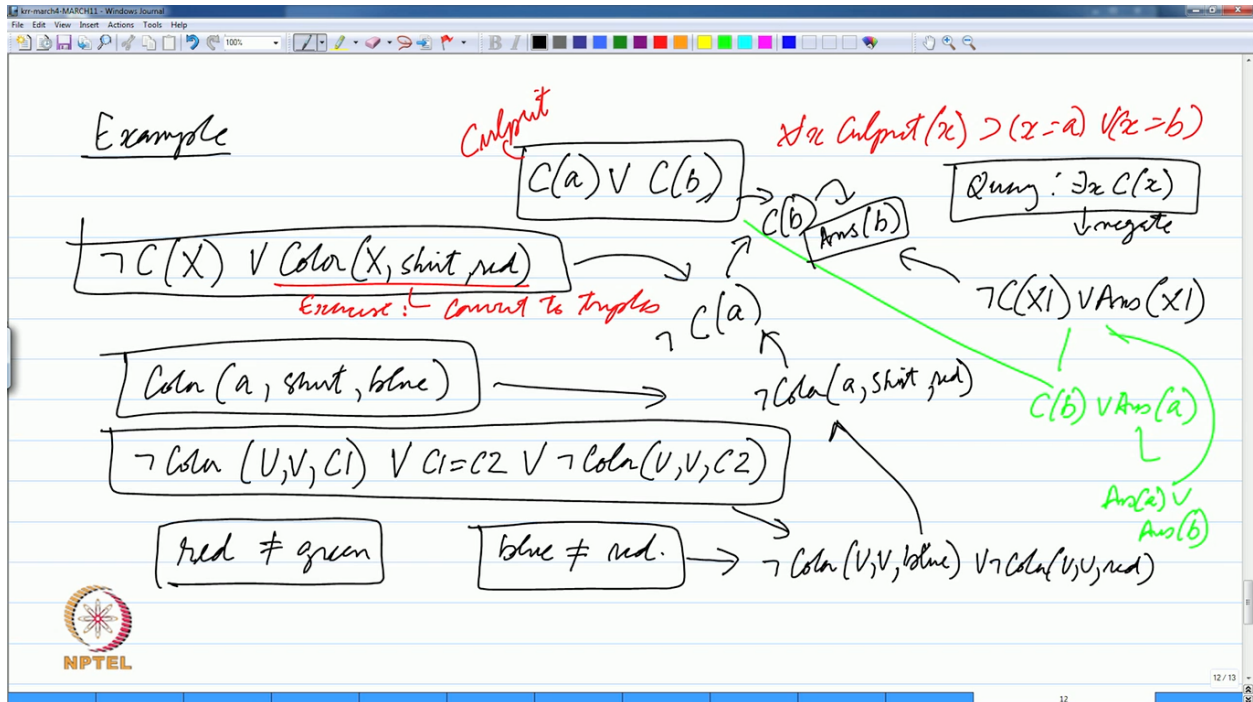
12 / 12

Now we are given this query that there is a culprit. Okay which again we negate and replace it by not C so let me use a different variable name x1 here or answer x1. I am using the answer predicate here. So one thing you can observe about the resolution method here is that basically the method that shows that something is unsatisfiable. I have got this set of clauses I have said that a or b is the culprit. Culprit is wearing red shirt. A is wearing the blue shirt and I am asking who is the culprit or is there a culprit. Now obviously there is a culprit because you have said that there is a a or b is a culprit.

Now one thing is that you can actually make a derivation which works as follows. That you derive from this something what do you derive from this. So let's say you put x1 is equal to a so you will get C b or answer a and then from this and this you can derive answer a or answer b. Of course this is not telling us anything. We started off by saying that a is a culprit or b is a culprit and now this is telling us that a is the answer or b is the answer. But in this particular example as you can see we have more information we can in fact identify the culprit. So what I am trying to highlight here is the fact that the resolution method may terminate like this green derivation that yes there is a culprit indeed without identifying who that culprit is stating that either a is the answer or b is the answer. But we can have a derivation so for example if I use this fact then I can get not color u v so C1 is blue here or not color u v red. Then I can take this one and I can take this one and by saying u is equal to a and v is equal to shirt I will get not color a shirt red because I said that a shirt is blue and this one says that it must be either red or blue so either we get the clause that a shirt is red and from that and this I will get not culprit a. Okay so we know that a is not wearing a red shirt and from this and the fact that culprits wear red shirt we can infer that a is not the culprit here.

Then we can resolve the fact that there is a culprit with this one to give us the fact that b is the culprit. Infact by now we have arrived at that answer but since we are looking at the resolution refutation method you see that between this clause and the goal clause you will terminate with answer.

(Refer Slide Time: 33:59)



So as we saw that there were two derivations one which is in green did not pinpoint the culprit for us but we have an alternate derivation which can exploit the information we have in the knowledge base that culprits wear red shirts and since we know that a is not wearing red shirt we could figure out that a is not the culprit and b is the culprit and that shows up in answer. So if the resolution method had taken this as the derivation then the answer would have been answer b. notice that both answers are logically correct. If you say b is the culprit it is correct if you say a is the culprit or b is the culprit even that is correct so there is nothing wrong which is going on.

Its just that we have not been able to identify what is . so in the next class we will take a closer look at equality and we will see that equality is a special predicate its not like any other predicate like father mother loves and so on because it carries with it a certain amount of mathematical knowledge which we will need to make explicit if we are to exploit the fact that equality has certain properties. So we will do that in the next class.

