

COMP210: Artificial Intelligence

Lecture 24. First-order resolution

Trevor Bench-Capon

<http://www.csc.liv.ac.uk/~tbc/COMP210/>

Dealing with Quantifiers

- Existential quantifiers

$\exists x \cdot m(x)$ is rewritten as $m(a)$

Informally *somebody is the murderer—call this person a*, *a* is a Skolem constant.

- Note any remaining variables are thought to be universally quantified.

$\exists y \forall x \cdot p(x) \Rightarrow q(x, y)$

is rewritten as

$\neg p(x) \vee q(x, a)$

where *a* is a skolem constant

Overview

- The last lecture we looked at resolution based proof for propositional logic.
- In this lecture we will give an overview to resolution in first-order logics and illustrate how knowledge representation and deduction can be carried out in first order logic.
- We also look at the connection between Prolog, logic and resolution.

Variable Free Resolution

- If a set of clauses contain no variables resolution can be applied similarly to the propositional case.
- Example: show

$$\left. \begin{array}{l} cat(Katy) \\ cat(Katy) \Rightarrow mammal(Katy) \end{array} \right\} \models mammal(Katy)$$

i.e. show

$$\left(\begin{array}{l} cat(Katy) \\ \wedge \\ (cat(Katy) \Rightarrow mammal(Katy)) \end{array} \right) \wedge \neg mammal(Katy)$$

is unsatisfiable.

Resolution Method

The method involves:-

- translation to a normal form (CNF);
- At each step, a new clause is derived from two clauses you already have
- Proof steps all use the same *resolution* rule;
- repeat until false is derived or no new formulae can be derived.

To Normal Form

- in conjunctive normal form:

$$\begin{array}{l} cat(Katy) \\ \neg cat(Katy) \vee mammal(Katy) \\ \neg mammal(Katy) \end{array}$$

Normal Form for Predicate Logic

- To write into normal form we must be able to deal with the removal of quantifiers (uses a technique known as Skolemisation).
- This is quite complex. We just give some examples.

Resolution

- Applying the resolution rule

- $cat(Katy)$ [given]
- $\neg cat(Katy) \vee mammal(Katy)$ [given]
- $\neg mammal(Katy)$ [given]
- $mammal(Katy)$ [1, 2]
- false** [3, 4]

- Thus $mammal(Katy)$ is a logical conclusion of $cat(Katy)$ and $cat(Katy) \Rightarrow mammal(Katy)$.

Resolution with Variables

- Show

$$\forall x \cdot \text{cat}(x) \Rightarrow \text{mammal}(x) \} \models \text{mammal}(\text{Katy})$$

- i.e. show the following is unsatisfiable.

$$\left(\begin{array}{c} \text{cat}(\text{Katy}) \\ \wedge \\ (\forall x \cdot \text{cat}(x) \Rightarrow \text{mammal}(x)) \end{array} \right) \wedge \neg \text{mammal}(\text{Katy})$$

Teveer Bench-Capron

COMP210: Artificial Intelligence, Lecture 24, First-order resolution—d. 930

Example of Non Termination

- Assume we have the following pair of clauses derived from a formula that is satisfiable. We try to show then unsatisfiable (but are in fact satisfiable).

- $q(y) \vee \neg q(g(y))$
- $\neg q(x) \vee \neg p(x)$

- The proof continues as follows.

- $\neg q(g(x)) \vee \neg p(x)$ [1, 2, $\{y \mapsto x\}$]
- $\neg q(g(g(x))) \vee \neg p(x)$ [1, 3, $\{y \mapsto g(x)\}$]
- $\neg q(g(g(g(x)))) \vee \neg p(x)$ [1, 4, $\{y \mapsto g(g(x))\}$]

...

etc

Teveer Bench-Capron

COMP210: Artificial Intelligence, Lecture 24, First-order resolution—d. 1330

To Normal Form

- in conjunctive normal form:

$$\begin{array}{l} \text{cat}(\text{Katy}) \\ \neg \text{cat}(x) \vee \text{mammal}(x) \\ \neg \text{mammal}(\text{Katy}). \end{array}$$

Teveer Bench-Capron

COMP210: Artificial Intelligence, Lecture 24, First-order resolution—d. 1030

Rule Base Example

- R1: IF animal has hair
THEN animal is a mammal
- R5: IF animal eats meat
THEN animal is carnivore
- R9: IF animal is mammal
AND animal is carnivore
AND animal has tawney colour
AND animal has dark spots
THEN animal is cheetah

Teveer Bench-Capron

COMP210: Artificial Intelligence, Lecture 24, First-order resolution—d. 1430

Resolution

- Now to resolve

$$\text{cat}(\text{Katy}) \text{ and } \neg \text{cat}(x) \vee \text{mammal}(x)$$

we must look for a way to replace x in $\neg \text{cat}(x)$ in clause 2 so that it matches with $\text{cat}(\text{Katy})$ in clause 1.

- We do this by applying the *substitution* $\{x \mapsto \text{Katy}\}$.
- The process of generating these substitutions is known as *unification*. We substitute the Most General Unifier: i.e make the fewest commitments needed to give a match.
- Clause 2 becomes $\neg \text{cat}(\text{Katy}) \vee \text{mammal}(\text{Katy})$ and now the proof continues as before.

Teveer Bench-Capron

COMP210: Artificial Intelligence, Lecture 24, First-order resolution—d. 1130

In FO Logic

- We can write the above rules in first-order logic as follows (there are other ways).

- $\forall x \cdot \text{has_hair}(x) \Rightarrow \text{mammal}(x)$
- $\forall x \cdot \text{eats}(x, \text{meat}) \Rightarrow \text{carnivore}(x)$
- $\forall x \cdot (\text{mammal}(x) \wedge \text{carnivore}(x) \wedge \text{colour}(x, \text{tawney}) \wedge \text{dark_spots}(x)) \Rightarrow \text{cheetah}(x)$

- Similarly for the other rules.

Teveer Bench-Capron

COMP210: Artificial Intelligence, Lecture 24, First-order resolution—d. 1530

Theoretical Considerations

- The transformation to normal form is satisfiability preserving. That is if there is a model for A then there is a model for the transformation of A into CNF.
- Soundness** If **false** is derived from applying the resolution method to a set of clauses S then S is unsatisfiable.
- Completeness** If S is an unsatisfiable set of clauses then a contradiction can be derived by applying the resolution method.
- Decidability** As we've already mentioned first-order logic is undecidable. Resolution is *semi-decidable*, i.e. given an unsatisfiable set of formulae it is guaranteed to derive false, however given a satisfiable set it may never terminate.

Teveer Bench-Capron

COMP210: Artificial Intelligence, Lecture 24, First-order resolution—d. 1230

Working Memory

- Assume that we have the following information in working memory.

cyril has hair,
cyril eats meat,
cyril has tawney colour,
cyril has dark spots

- This can be written in first-order logic as follows.

- $\text{has_hair}(\text{cyril})$
- $\text{eats}(\text{cyril}, \text{meat})$
- $\text{colour}(\text{cyril}, \text{tawney})$
- $\text{dark_spots}(\text{cyril})$

Teveer Bench-Capron

COMP210: Artificial Intelligence, Lecture 24, First-order resolution—d. 1630

Goal

- Assume we want to show that cyril is a cheetah
- This can be written in first-order logic as

$cheetah(cyril)$

Refinements

- Many ways at refining resolution for example restricting which clauses can be resolved.
- Such restrictions may affect completeness.

Reasoning

- To show that $cheetah(cyril)$ follows from the above first-order formula we must show

$L1, L5, L9, F1, F2, F3, F4 \models cheetah(cyril)$

- We show $L1 \wedge L5 \wedge L9 \wedge F1 \wedge F2 \wedge F3 \wedge F4 \wedge \neg cheetah(cyril)$ is unsatisfiable. We abbreviate cyril by c

Example

- All birds have feathers. (1) $Fx \vee \neg Bx$
- All penguins are not able to fly. (2) $\neg Ax \vee \neg Px$
- Some birds fly. (3) Ba (4) Aa
- Show, some birds are not penguins. $\neg(Bx \rightarrow Px)$ i.e. $\neg(Bx \wedge \neg Px)$
- Add negation: (5) $(\neg Bx \wedge Px)$
- Resolve (2) and (4) to get (7) $\neg Pa$
- Resolve (7) and (5) to get (8) $\neg Ba$
- Resolve (3) and (8) to get empty clause

Proof

- $\neg has_hair(x) \vee mammal(x)$
- $\neg eats(y, meat) \vee carnivore(y)$
- $\neg mammal(z) \vee \neg carnivore(z) \vee \neg colour(z, tauney) \vee \neg dark_spots(z) \vee cheetah(z)$
- $has_hair(c)$
- $eats(c, meat)$
- $colour(c, tauney)$
- $dark_spots(c)$
- $\neg cheetah(c)$
- $\neg mammal(c) \vee \neg carnivore(c) \vee \neg colour(c, tauney) \vee \neg dark_spots(c)$ [3, 8, { $z \mapsto c$ }]
- $\neg mammal(c) \vee \neg carnivore(c) \vee \neg colour(c, tauney)$ [7, 9]
- $\neg mammal(c) \vee \neg carnivore(c)$ [6, 10]
- $\neg mammal(c) \vee \neg eats(c, meat)$ [2, 11, { $y \mapsto c$ }]
- $\neg mammal(c)$ [5, 12]
- $\neg has_hair(c)$ [1, 13, { $x \mapsto c$ }]
- false** [4, 14]

Prolog and First-Order Logic

- Prolog programs are really first-order logic formulae where variables are assumed to be universally quantified.

Search

- Deciding which clauses to resolve together to obtain a proof is similar to the search problems we looked at earlier in the course.
- To show p follows from some database D , i.e.

$D \models p$

we apply resolution to

$D \wedge \neg p.$

- If we resolve first with clauses derived from $\neg p$, and then the newly derived clauses we have a *backward chaining system*.

Example

- Earlier in the course we had the following Prolog program.
- ```
parent(cathy, ian).
parent(pete, ian).

female(cathy).
male(pete).

mother(X, Y) :- parent(X, Y), female(X).
```

## In FO Logic

- Writing this in FOL we obtain the following.

$$\begin{aligned} & \text{parent}(\text{cathy}, \text{ian}) \wedge \\ & \text{parent}(\text{pete}, \text{ian}) \wedge \\ & \text{female}(\text{cathy}) \wedge \\ & \text{male}(\text{pete}) \wedge \\ & \forall x \forall y \cdot (\text{parent}(x, y) \wedge \text{female}(x)) \Rightarrow \text{mother}(x, y) \end{aligned}$$

Teveer Bandh-Capran

COMP210: Artificial Intelligence, Lecture 24, First-order resolution - d. 29/30

## Unification vs Matching

- Note performing unification on  $p(x)$  and  $p(f(x))$  fails (known as the *occurs check* as we are not allowed to unify  $x$  with  $f(x)$ ).
- However many implementations of Prolog will attempt to unify these two terms causing a loop in execution.
  - It was believed that implementing the occurs check would make matching inefficient.
- Try the program  
$$p(X) :- p(f(X)).$$
with the query  $p(a)$ .

Teveer Bandh-Capran

COMP210: Artificial Intelligence, Lecture 24, First-order resolution - d. 29/30

## Facts, Rules and Queries

- Facts (eg  $\text{male}(\text{pete})$ ) in Prolog programs are atomic sentences in FOL.
- Rules in Prolog programs such as  
$$p(X, Y, Z) :- q(X), r(Y, Z)$$
are universally quantified FOL formulae.  
$$\forall x, \forall y, \forall z \cdot q(x) \wedge r(y, z) \Rightarrow p(x, y, z)$$
- Queries in Prolog such as  $\text{mother}(\text{cathy}, \text{ian})$  are dealt with by testing whether  $\text{mother}(\text{cathy}, \text{ian})$  follows from the FOL formula representing facts and rules of the Prolog program.

Teveer Bandh-Capran

COMP210: Artificial Intelligence, Lecture 24, First-order resolution - d. 29/30

## Summary I

- We've shown how to apply resolution to propositional and first-order logics.
- If a rule based system is written in FOL we can use resolution to show whether a particular fact follows from the facts (in working memory) and the rules base.
- Although resolution is sound and complete it is semi-decidable, i.e. applying resolution to a satisfiable formula may lead to non-termination.

Teveer Bandh-Capran

COMP210: Artificial Intelligence, Lecture 24, First-order resolution - d. 30/30

## Horn Clauses

- Here is our example written into clausal form.
  1.  $\text{parent}(\text{cathy}, \text{ian})$
  2.  $\text{parent}(\text{pete}, \text{ian})$
  3.  $\text{female}(\text{cathy})$
  4.  $\text{male}(\text{pete})$
  5.  $\neg \text{parent}(x, y) \vee \neg \text{female}(x) \vee \text{mother}(x, y)$
- Here the clauses 1-4 contain only one positive predicate and clause 5 contains two negative predicates and one positive.
  - *Horn Clauses*
- Dealing with Horn Clauses can be very efficient.

Teveer Bandh-Capran

COMP210: Artificial Intelligence, Lecture 24, First-order resolution - d. 27/30

## Summary II

- Logic is useful for knowledge representation as it has syntax, a well defined semantics (we know what formulae mean), and proof methods eg resolution allowing us to show a formula is a logical consequence of others.
- Prolog is known as a logic programming language. The language of Prolog is a restricted version of first-order logic and inference is by a form of resolution.

Teveer Bandh-Capran

COMP210: Artificial Intelligence, Lecture 24, First-order resolution - d. 31/30

## Inference

- Prolog answers queries by using a special form of resolution known as *SLD resolution*.
- That is asking the query  $\text{mother}(\text{cathy}, \text{ian})$  of the Prolog program given earlier is similar to applying resolution to the FOL formula of the program conjoined with  $\neg \text{mother}(\text{cathy}, \text{ian})$ .
- Matching in Prolog corresponds to *unification* in resolution.

Teveer Bandh-Capran

COMP210: Artificial Intelligence, Lecture 24, First-order resolution - d. 28/30