Comp24412 Symbolic AI
Lecture 7: First-Order Logic and Prolog

Ian Pratt-Hartmann

Room KB2.38: email: ipratt@cs.man.ac.uk

2016–17
Some formulas in the predicate calculus ...

\[ \text{man}(\text{socrates}) \]
\[ \forall x (\text{man}(x) \rightarrow \text{mortal}(x)) \]
\[ \text{parent}(\text{pam}, \text{bob}) \]
\[ \forall x \text{ancestor}(x, x) \]
\[ \forall x \forall y \forall z ((\text{parent}(x, y) \land \text{ancestor}(y, z)) \rightarrow \text{ancestor}(x, z)) \]

... and the corresponding Prolog assertions

\[ \text{man}(\text{socrates}) \]
\[ \text{mortal}(X) :- \text{man}(X) \]
\[ \text{parent}(\text{pam}, \text{bob}) \]
\[ \text{ancestor}(X,X) \]
\[ \text{ancestor}(X,Z) :- \text{parent}(X,Y), \text{ancestor}(Y,Z) \]
In fact, Prolog maps systematically onto logic as follows:

- Facts map to atomic formulas:
  \[ \text{man(socrates)} \mapsto \text{man(socrates)} \]

- Variables are treated as universally quantified
  \[ \text{ancestor(X,X)} \mapsto \forall x(\text{ancestor}(x,x)) \]

- And rules map to conditional statements
  \[
  \text{mortal(X):- man(X)} \mapsto \\
  \forall x(\text{man}(x) \rightarrow \text{mortal}(x))
  \]
• So all Prolog assertions correspond to formulas in the predicate calculus.
• But what about the other way ’round?
• Given a formula, e.g.

$$\forall x \exists y (\neg a(x, y) \rightarrow (b(x) \lor a(y)))$$

can we translate it into Prolog?
• The answer is: sort of
• The important notion here is that of a normal form
• A literal is an atomic proposition or an atomic proposition letter prefixed by $\neg$.
• Thus, $p$, $q(x)$, $\neg r(x, f(x, y))$ are all literals; $s(x, y) \lor s(x, y)$, $s(x, y) \land p$ and $\neg \neg s(x, y)$ are not literals.
• A clause is a (possibly empty) collection of literals all joined by $\lor$.
• Thus $\neg p(x)$, $p(x) \lor q$, $\neg p(x) \lor \neg q \lor r$ and $\bot$ are all clauses; $p(x) \land q$, $p(x) \rightarrow q$ and $\neg \neg p(x)$ are not.
• The following result can easily be established:
• Let $\varphi$ be a quantifier-free formula. Then there exist clauses $c_1, \ldots, c_n$, such that

$$\models \varphi \iff (c_1 \land c_2 \land \ldots \land c_n).$$

• In other words, any quantifier-free formula can be transformed into a logically equivalent to a conjunction of clauses. We call this process **putting the formula in clause form**.

• **Oh, I almost forgot:** $\varphi \iff \psi$ abbreviates $(\varphi \to \psi) \land (\psi \to \varphi)$
• See Blackburn and Bos pp. 175 ff. for an explanation of how to perform the conversion.
• A formula in the predicate calculus in which all the quantifiers are at the front is said to be in prenex form.

• The formulas

\[ \forall x (\text{man}(x) \rightarrow \text{mortal}(x)) \]
\[ \forall x \exists y \text{loves}(x, y) \]

are in prenex form

• \[ \forall x (\text{boy}(x) \rightarrow \exists y (\text{girl}(y) \land \text{loves}(x, y))) \]
\[ \forall x ((\text{boy}(x) \land \exists y (\text{girl}(y) \land \text{loves}(x, y)))) \rightarrow \text{happy}(x)) \]

are not.
• The following result can easily be established:
• Let $\varphi$ be a formula. Then there exists a prenex form formula $\psi$ such that
  \[
  \models \varphi \leftrightarrow \psi.
  \]
• In other words, any formula is logically equivalent to a formula with all the quantifiers at the front.
• Example
\[ \forall x (\text{boy}(x) \rightarrow \exists y (\text{girl}(y) \land \text{loves}(x, y))) \]
is equivalent to
\[ \forall x \exists y (\text{boy}(x) \rightarrow (\text{girl}(y) \land \text{loves}(x, y))) \]

• Example
\[ \forall x ((\text{boy}(x) \land \exists y (\text{girl}(y) \land \text{loves}(x, y))) \rightarrow \text{happy}(x)) \]
is equivalent to
\[ \forall x \forall y ((\text{boy}(x) \land \text{girl}(y) \land \text{loves}(x, y)) \rightarrow \text{happy}(x)) \]

Why does the existential quantifier get changed into a universal quantifier?
• Given a prenex formula, any existential quantifiers can be eliminated

• Consider

\[ \exists x (\text{man}(x) \land \text{philosopher}(x)) \]

• This is **equisatisfiable** with

\[ \text{man}(a) \land \text{philosopher}(a). \]

(This name must be *new*: i.e. not occurring in any other formulas.)

• Similarly

\[ \exists y \forall x (\text{loves}(x, y)) \]

is equisatisfiable with

\[ \forall x (\text{loves}(x, b)). \]
• Things are different however, for:

\[ \forall x \exists y (\text{loves}(x, y)). \]

This formula is equisatisfiable with:

\[ \forall x (\text{loves}(x, f(x))) \]

where \( f \) is a function (called a Skolem function).

• Note the use of a function (rather than a constant): the formula says that everyone loves someone or other – but not necessarily the same person.

• We need a function whenever we want to eliminate an existential quantifier which is to the right of some universal quantifiers.
• Suppose then, we put a formula in prenex form, and eliminate all the existential quantifiers.
• The result will be
\[ \forall x_1 \ldots \forall x_n \chi \]
where \( \chi \) is quantifier-free.
• But the universal quantifiers convey no information any more, so we might as well write:
\[ \chi \]
• Now we can put \( \chi \) (the quantifier-free rump) into clause form!
• So we can massage any formula of the predicate calculus into a collection of clause form expressions (containing variables) of the form

\[ e_1 \lor \ldots \lor e_N \]

where the \( e_i \) are literals

• Example:

\[
\forall x \forall z (\exists y (\text{parent}(x, y) \land \text{ancestor}(y, z)) \rightarrow \text{ancestor}(x, z))
\]

\[
\Rightarrow \\
\neg \text{parent}(x, y) \lor \neg \text{ancestor}(y, z) \lor \text{ancestor}(x, z)
\]
• Suppose that exactly one of these literals is positive and the rest are negative
• Then we have something like:

\[ p_1 \lor \neg p_2 \lor \ldots \neg p_N \]

• Which we know to be equivalent to the Prolog rule
\[ p_1 :- \ p_2, \ldots \ p_N \]
• Example:

\[ \text{ancestor}(x, z) \lor \neg \text{parent}(x, y) \lor \neg \text{ancestor}(y, z) \]

\[ \text{ancestor}(X,Z):- \ \text{parent}(X,Y), \ \text{ancestor}(Y,Z) \]
• If the formula consists of exactly one positive literal \( p \), then this corresponds to the Prolog fact \( p \).
• So we have answered our question:
• The formulas that are expressible as Prolog are those whose normal forms are conjunctions of clause-form formulas having exactly one positive literal (and any number of negative literals).
• Clauses in which there is at most one positive literal are called Horn clauses.
• The difference between exactly and at most one is not really essential.
• Revise Chapter 1 of *Representation and inference for natural Language*
• Read Chapter 7 of *Learn Prolog Now!* for next lecture