COMP30026 Models of Computation

Predicate Logic: Clausal Form

Bach Le / Anna Kalenkova

Lecture Week 4 Part 2 (Zoom)

Semester 2, 2021

This Lecture is Being Recorded

Resolution for Predicate Logic

The resolution technique generalises to first-order predicate logic.

As for propositional logic, it assumes clausal form, that is, having a formula presented in conjunctive normal form, without any quantifiers.

Again, it consists on generating logical consequences (resolvents), trying to derive an empty clause, thereby proving the original formula unsatisfiable.

Existential quantifiers are eliminated in a process called Skolemization.

Eliminating Existential Quantifiers

Consider the formula $F = \exists x \forall y \; P(x, y)$ under some interpretation $\mathcal{I}$.

F is satisfiable iff some valuation σ makes $\forall y \; P(x, y)$ true. Say that σ, with $\sigma(x) = d_0$, makes $\forall y \; P(x, y)$ true.

Now consider a fresh constant a and the formula $\boxed{\forall y \; P(a, y)}$.

This formula is satisfiable iff F is. If $\mathcal{I}$ satisfies F then we simply add to $\mathcal{I}$ the mapping of a to d_0—this extended interpretation satisfies $\forall y \; P(a, y)$.

If $\forall y \; P(x, y)$ is unsatisfiable, there is no valuation that will make $\forall y \; P(x, y)$ true. Hence no interpretation will make $\forall y \; P(a, y)$ true.

Skolem Constants and Functions

Now consider the formula $G = \forall y \exists x\ P(x, y)$.

We cannot conclude that $\forall y\ P(a, y)$ is satisfiable iff G is.

Since $\exists x$ is within the scope of $\forall y$, the value of x for which $P(x, y)$ holds may differ given different values of y. The value of x is a function of the value of y. Replacing x by a constant will not do.

But then we can generate the formula $\forall y\ P(f(y), y)$, choosing a fresh function symbol f.

For reasons similar to those outlined on slide 4, this formula is satisfiable iff G is.

Skolemization

We call a (on slide 4) a Skolem constant, and f (on slide 5) a Skolem function.

Skolem functions can be of arbitrary arity. To eliminate $\exists y$ in $\forall x_1, x_2, x_3\ \exists y[\ldots]$ we replace each occurrence of y in its scope by $f(x_1, x_2, x_3)$.

Namely, y may depend on all three xs.

Each introduced Skolem constant or function must be fresh.

Recall also our convention: We use letters from the start of the alphabet (a, b, c, ...) for constants, and letters from the end of the alphabet (u, v, x, y ...) for variables.

Skolemization Example

This formula has three existential quantifiers—we remove them one by one:

$$\exists u \forall v \exists x \forall y \exists z$$
$$((\neg P(u, f(v), x, b) \vee R(g(x, y), u)) \wedge S(y, g(a, z)))$$

Skolemization Example

This formula has three existential quantifiers—we remove them one by one:

$$\exists u \forall v \exists x \forall y \exists z$$
$$((\neg P(u, f(v), x, b) \vee R(g(x, y), u)) \wedge S(y, g(a, z)))$$
$$\rightsquigarrow \quad \forall v \exists x \forall y \exists z$$
$$((\neg P(c, f(v), x, b) \vee R(g(x, y), c)) \wedge S(y, g(a, z)))$$

Skolemization Example

This formula has three existential quantifiers—we remove them one by one:

$$\exists u \forall v \exists x \forall y \exists z$$
$$((\neg P(u, f(v), x, b) \vee R(g(x, y), u)) \wedge S(y, g(a, z)))$$

$$\rightsquigarrow \quad \forall v \exists x \forall y \exists z$$
$$((\neg P(c, f(v), x, b) \vee R(g(x, y), c)) \wedge S(y, g(a, z)))$$

$$\rightsquigarrow \quad \forall v \forall y \exists z$$
$$((\neg P(c, f(v), h(v), b) \vee R(g(h(v), y), c)) \wedge S(y, g(a, z)))$$

Skolemization Example

This formula has three existential quantifiers—we remove them one by one:

$$
\begin{array}{ll}
 & \exists u \forall v \exists x \forall y \exists z \\
 & \quad ((\neg P(u, f(v), x, b) \lor R(g(x, y), u)) \land S(y, g(a, z))) \\
\rightsquigarrow & \forall v \exists x \forall y \exists z \\
 & \quad ((\neg P(c, f(v), x, b) \lor R(g(x, y), c)) \land S(y, g(a, z))) \\
\rightsquigarrow & \forall v \forall y \exists z \\
 & \quad ((\neg P(c, f(v), h(v), b) \lor R(g(h(v), y), c)) \land S(y, g(a, z))) \\
\rightsquigarrow & \forall v \forall y \\
 & \quad ((\neg P(c, f(v), h(v), b) \lor R(g(h(v), y), c)) \land S(y, g(a, j(v, y))))
\end{array}
$$

Skolemization Example

This formula has three existential quantifiers—we remove them one by one:

$$
\begin{array}{ll}
 & \exists u \forall v \exists x \forall y \exists z \\
 & \quad ((\neg P(u, f(v), x, b) \lor R(g(x, y), u)) \land S(y, g(a, z))) \\
\rightsquigarrow & \forall v \exists x \forall y \exists z \\
 & \quad ((\neg P(c, f(v), x, b) \lor R(g(x, y), c)) \land S(y, g(a, z))) \\
\rightsquigarrow & \forall v \forall y \exists z \\
 & \quad ((\neg P(c, f(v), h(v), b) \lor R(g(h(v), y), c)) \land S(y, g(a, z))) \\
\rightsquigarrow & \forall v \forall y \\
 & \quad ((\neg P(c, f(v), h(v), b) \lor R(g(h(v), y), c)) \land S(y, g(a, j(v, y))))
\end{array}
$$

Instead of $j(v, y)$ we could have chosen $k(v, y)$, or even $j(y, v)$—as long as we replace each occurrence of z by the same term, of course.

From Predicate Logic Formulas to Clausal Form

The process is similar to what we did with propositional formulas:

1. Replace occurrences of $\oplus$, $\Leftrightarrow$, and $\Rightarrow$.
2. Drive negation in.
3. Standardise bound variables apart.
4. Eliminate existential quantifiers (Skolemize).
5. Eliminate universal quantifiers (just remove them).
6. Bring to CNF (using the distributive laws).

Clausal Form: Step 1—Use Just $\vee$, $\wedge$, $\neg$

Let us use this running example:

$$\forall x\, (P(x) \Leftrightarrow \exists y\, (R(x,y) \wedge \forall z\, R(z,y)))$$

First use the usual translations to eliminate $\Leftrightarrow$ (and $\Rightarrow$ and $\oplus$):

$$\forall x \left(\begin{array}{l} (P(x) \Rightarrow \exists y\, (R(x,y) \wedge \forall z\, R(z,y))) \wedge \\ \qquad (\exists y\, (R(x,y) \wedge \forall z\, R(z,y)) \Rightarrow P(x)) \end{array} \right)$$

which then becomes:

$$\forall x \left(\begin{array}{l} (\neg P(x) \vee \exists y\, (R(x,y) \wedge \forall z\, R(z,y))) \wedge \\ \qquad (\neg \exists y\, (R(x,y) \wedge \forall z\, R(z,y)) \vee P(x)) \end{array} \right)$$

Clausal Form: Step 2—Push Negation

Next drive negation in.

$$\forall x \left(\begin{array}{l} (\neg P(x) \vee \exists y\ (R(x,y) \wedge \forall z\ R(z,y))) \wedge \\ \qquad (\neg \exists y\ (R(x,y) \wedge \forall z\ R(z,y)) \vee P(x)) \end{array} \right)$$

then becomes

$$\forall x \left(\begin{array}{l} (\neg P(x) \vee \exists y\ (R(x,y) \wedge \forall z\ R(z,y))) \wedge \\ \qquad (\forall y\ (\neg R(x,y) \vee \exists z\ \neg R(z,y)) \vee P(x)) \end{array} \right)$$

Now rename variables so that no two quantifiers use the same variable name. With that,

$$\forall x \left(\begin{array}{c} (\neg P(x) \lor \exists y\ (R(x,y) \land \forall z\ R(z,y))) \land \\ (\forall y\ (\neg R(x,y) \lor \exists z\ \neg R(z,y)) \lor P(x)) \end{array} \right)$$

turns into, say

$$\forall x \left(\begin{array}{c} (\neg P(x) \lor \exists y\ (R(x,y) \land \forall z\ R(z,y))) \land \\ (\forall u\ (\neg R(x,u) \lor \exists v\ \neg R(v,u)) \lor P(x)) \end{array} \right)$$

Clausal Form: Step 4—Skolemize

Let us highlight the existentially quantified variables:

$$\forall x \left(\begin{array}{c} (\neg P(x) \lor \exists y\ (R(x,y) \land \forall z\ R(z,y))) \land \\ (\forall u\ (\neg R(x,u) \lor \exists v\ \neg R(v,u)) \lor P(x)) \end{array} \right)$$

The existentially quantified y is in the scope of $\forall x$—so replace it by $f(x)$.

Let us highlight the existentially quantified variables:

$$\forall x \left(\begin{array}{l} (\neg P(x) \vee \exists y\ (R(x, y) \wedge \forall z\ R(z, y))) \wedge \\ \quad (\forall u\ (\neg R(x, u) \vee \exists v\ \neg R(v, u)) \vee P(x)) \end{array} \right)$$

The existentially quantified y is in the scope of $\forall x$—so replace it by $f(x)$.

The existentially quantified v is in the scope of $\forall x$, as well as of $\forall u$. So we replace it by $g(u, x)$.

$$\forall x \left(\begin{array}{l} (\neg P(x) \vee (R(x, f(x)) \wedge \forall z\ R(z, f(x)))) \wedge \\ \quad (\forall u\ (\neg R(x, u) \vee \neg R(g(u, x), u)) \vee P(x)) \end{array} \right)$$

Clausal Form: Step 5—Drop Universal Quantifiers

Eliminating universal quantifiers is easy:

$$\forall x \left(\begin{array}{c} (\neg P(x) \vee (R(x, f(x)) \wedge \forall z\ R(z, f(x)))) \wedge \\ (\forall u\ (\neg R(x, u) \vee \neg R(g(u, x), u)) \vee P(x)) \end{array} \right)$$

becomes

$$(\neg P(x) \vee (R(x, f(x)) \wedge R(z, f(x)))) \wedge (\neg R(x, u) \vee \neg R(g(u, x), u) \vee P(x))$$

It is understood that all variables are now universally quantified. If you prefer, you can think of all the universal quantifiers as sitting in front of the formula.

Clausal Form: Step 6—Convert to CNF

$$(\neg P(x) \vee (R(x, f(x)) \wedge R(z, f(x)))) \wedge (\neg R(x, u) \vee \neg R(g(u, x), u) \vee P(x))$$

becomes, using distribution:

$$\begin{array}{l} (\neg P(x) \vee R(x, f(x))) \wedge \\ (\neg P(x) \vee R(z, f(x))) \wedge \\ (\neg R(x, u) \vee \neg R(g(u, x), u) \vee P(x)) \end{array}$$

or, written as a set of sets of literals:

$$\left\{ \begin{array}{c} \{\neg P(x), R(x, f(x))\}, \\ \{(\neg P(x), R(z, f(x))\}, \\ \{(\neg R(x, u), \neg R(g(u, x), u), P(x))\} \end{array} \right\}$$

Justifying Skolemization

Note that Skolemization of a formula does not produce a logically equivalent formula.

For example, $\forall x \exists y\ P(x, y)$ turns into $\forall x\ P(x, f(x))$.

If we interpret these in the domain $\mathbb{Z}$ of integers, interpreting f as the "successor" function $(+1)$, and P as $>$, then the original formula is satisfied, but the second is not.

However, Skolemization does produce an equisatisfiable formula—one that is satisfiable iff the original was—and this is all we care about for the purposes of resolution proofs.

A First Look at Resolution for Predicate Logic

We wish to develop the resolution principle for predicate logic with function symbols.

However, now we will not be resolving on simple literals, but on atomic formulas containing variables, constants, and function symbols.

Simple cases seem easy enough, for example, from

$$\neg B(x) \vee M(x) \quad \text{and} \quad \neg M(c)$$

we would like to conclude $\neg B(c)$. ("Every borogove is mimsy" and "Colin is not mimsy" entails "Colin is not a borogove.")

Resolution for Predicate Logic

Note that all variables in

$$\neg B(x) \vee M(x) \quad \text{and} \quad \neg M(c)$$

are universally quantified.

In particular, we could instantiate $\neg B(x) \vee M(x)$ to $\neg B(c) \vee M(c)$; then we will be resolving the two clauses on $M(c)$ and its negation, just as happened in the propositional case.

The resolvent then comes out as $\neg B(c)$, as we hoped.

Next we will develop this idea and define resolution deduction for arbitrary sets of clauses.

Next Week

How resolution can be extended to predicate logic.

Plus: Induction principles.