Week 6 - Resolution for Predicate Logic

Billy Price

Goal Make resolution work for predicate logic.

Prop: $P, Q, \neg R$

Pred: $P(x,a), Q(y), \neg R(y,z)$

Problem How do we deal with quantifiers?

Solution Transform formulas to remove quantifiers, but preserve satisfiability

"Why does this preserve satisfiability?"

Definition An interpretation I models a formula φ if for all valuations, v, I makes φ true (using v).

SKOLEMIZATION

OLD φ	NEW φ'
$\exists x\, P(x)$	$P(a)$ ← fresh constant
$\forall x\, P(x)$	$P(x)$
$\forall x\, \forall y\, \exists z [P(x) \wedge Q(y) \wedge R(z)]$	$P(x) \wedge Q(y) \wedge R(f(x,y))$ ← fresh function symbol

φ satisfiable in lies φ' satisfiable

Definition A valuation $v : \text{vars} \to D$ is a function assigning the free variables of a formula to objects, and, by extension, compiles all terms to objects

"What's a valuation again?"

Note: Quantifiers $\forall$ and $\exists$ modify the v when evaluating the inner formula of the quantifier

CONVERSION TO CLAUSAL FORM

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).

Do not pass this line until there are no "$\oplus, \Leftrightarrow, \Rightarrow$," and there are no "$\neg$" outside any quantifiers

Two contradictory formulas

$\forall x\, P(x)$ ⇝ $\{\{P(x)\}\}$

$\exists y\, \neg P(y)$ ⇝ $\{\{\neg P(a)\}\}$

$x \mapsto a$

$\{\}$

Now we can resolve clauses as normal, we just need to specialize the variables in pairs of clauses, so they are talking about the same objects (via constants/functions)

In general, we just need to spot a $P(\rightsquigarrow)$ in one clause and a $\neg P(\rightsquigarrow)$ another, and then try to UNIFY the argument terms.

Note We can only really specialize variables e.g. $P(a)$ and $\neg P(b)$ do not contradict

UNIFICATION

1. $F(s_1, \ldots, s_n) = F(t_1, \ldots, t_n)$:
   - Replace the equation by the n equations $s_1 = t_1, \ldots, s_n = t_n$.
2. $F(s_1, \ldots, s_n) = G(t_1, \ldots, t_m)$ where $F \neq G$ or $n \neq m$:
   - Halt, returning 'failure'.
3. $x = x$:
   - Delete the equation.
4. $t = x$ where t is not a variable:
   - Replace the equation by $x = t$.
5. $x = t$ where t is not x but x occurs in t:
   - Halt, returning 'failure'.
6. $x = t$ where t contains no x but x occurs in other equations:
   - Replace x by t in those other equations.