Logik für Informatiker
Logic for computer scientists

First-order resolution

Till Mossakowski

WiSe 2009/10

First-order resolution

- generalises propositional resolution to first-order logic
- is a proof system that is well-suited for efficient implementation
- many automated first-order provers are based on resolution: SPASS, Prover9, Vampire
- also interactive provers for higher-order logic are based on resolution: Isabelle, HOL, HOL-light

Satisfiability and logical consequence

Logical consequence can be reduced to (un)satisfiability:

> *The logical consequence* $\mathcal{T} \models S$ *holds*
> *if and only if*
> $\mathcal{T} \cup \{\neg S\}$ *is unsatisfiable.*

Note: Resolution is about satisfiability.

Skolemization

The sentence

$$\forall x \exists y Neighbor(x, y)$$

is logically equivalent to the second-order sentence

$$\exists f \forall x Neighbor(x, f(x))$$

In first-order logic, we have the *Skolem normal form*

$$\forall x Neighbor(x, f(x))$$

Theorem about Skolem normal form

Theorem

A sentence $S \equiv \forall x \exists y P(x, y)$ is satisfiable iff its Skolem normal form $\forall x P(x, f(x))$ is.

Every structure satisfying the Skolem normal form also satisfies S. Moreover, every structure satisfying S can be turned into one satisfying the Skolem normal form. This is done by interpreting f by a function which picks out, for any object b in the domain, some object c such that they satisfy $P(x, y)$.

$$\{P(f(a)), \forall x\ \neg P(f(g(x)))\}$$

is satisfiable, but

$$\{P(f(g(a))), \forall x\ \neg P(f(x))\}$$

is not. This can be seen with *unification*.

Terms $t_1, \ldots, t_n$ are *unifiable*, if there is a substitution of terms for some or all the variables in $t_1, \ldots, t_n$ such that the terms that result from the substitution are syntactically identical terms.

Example

$$f(g(z), x), \quad f(y, x), \quad f(y, h(a))$$

are unifiable by substituting $h(a)$ for x and $g(z)$ for y.

Prenex Normal Form

Goal: shift all quantifiers to the top-level

$$(\forall x P) \wedge Q \leadsto \forall x(P \wedge Q) \qquad (\exists x P) \wedge Q \leadsto \exists x(P \wedge Q)$$
$$P \wedge (\forall x Q) \leadsto \forall x(P \wedge Q) \qquad P \wedge (\exists x Q) \leadsto \exists x(P \wedge Q)$$
$$(\forall x P) \vee Q \leadsto \forall x(P \vee Q) \qquad (\exists x P) \vee Q \leadsto \exists x(P \vee Q)$$
$$P \vee (\forall x Q) \leadsto \forall x(P \vee Q) \qquad P \vee (\exists x Q) \leadsto \exists x(P \vee Q)$$
$$\neg \forall x P \leadsto \exists x(\neg P) \qquad \neg \exists x P \leadsto \forall x(\neg P)$$
$$(\forall x P) \rightarrow Q \leadsto \exists x(P \rightarrow Q) \qquad (\exists x P) \rightarrow Q \leadsto \forall x(P \rightarrow Q)$$
$$P \rightarrow (\forall x Q) \leadsto \forall x(P \rightarrow Q) \qquad P \rightarrow (\exists x Q) \leadsto \exists x(P \rightarrow Q)$$

$$P \leftrightarrow Q \leadsto (P \rightarrow Q) \wedge (Q \rightarrow P)$$

Alpha-renaming (change of bound variables)

The Prenex normal form algorithm assumes that all variables in a formula are distinct. This can be achieved by α-renaming:

$\forall x P(x) \leadsto \forall y P(y)$

$\exists x P(x) \leadsto \exists y P(y)$

Suppose that we have a set $\mathcal{T}$ of sentences an want to show that they are not simultaneously first-order satisfiable.

1. Put each sentence in $\mathcal{T}$ into prenex form, say
$$\forall x_1 \exists y_1 \forall x_2 \exists y_2 \ldots P(x_1, y_1, x_2, y_2, \ldots)$$
2. Skolemize each of the resulting sentences, say
$$\forall x_1 \forall x_2 \ldots P(x_1, f_1(x_1), x_2, f_2(x_1, x_2), \ldots)$$
using different Skolem functions for different sentences.
3. Put each quantifier free matrix P into conjunctive normal form, say
$$P_1 \wedge P_2 \wedge \ldots \wedge P_n$$
where each P_i is a disjunction of literals.
4. Distribute the universal quantifiers in each sentence across the conjunctions and drop the conjunction signs, ending with a set of sentences of the form
$$\forall x_1 \forall x_2 \ldots P_i$$

5. Change the bound variables in each of the resulting sentences so that no variable appears in two of them.

6. Turn each of the resulting sentences into a set of literals by dropping the universal quantifiers and disjunction signs. In this way we end up with a set of resolution clauses.

7. Use resolution and unification to resolve this set of clauses

$$\frac{\{C_1, \ldots, C_m\},\ \{\neg D_1, \ldots, D_n\}}{\{C_2\theta, \ldots C_m\theta, D_2\theta, \ldots, D_n\theta\}}$$

if $C_1\theta = D_1\theta$ (θ is a unifier of C_1 and D_1)