CS 228 tut 6

16.10

Minimizing arguments to Skolem functions

$Q_1 x_1\ Q_2 x_2 \ldots\ Q_k x_k$

Q_i: $\exists$, $\forall$

No. of arguments to f_i ← Skolem fn if $Q_i = \exists$

= No. of $\forall$ upto i

Idea: Have minimal $\forall$ before any $\exists$

1) Rename apart ← this is important to allow us to move quantifiers around

2) NNF

3) Prenex form: Actually playing with quantifier scope.

4) Skolemization: Optimising for this step.

Because of rename apart, x's & y's disjoint

$$\underbrace{Q_1 x_1\ Q_2 x_2 \ldots Q_i x_i}_{\text{ind of } y_1}\ F \underset{\vee,\ \wedge}{\circ}\ \underbrace{R_1 y_1\ R_2 y_2 \ldots R_j y_j}_{\text{ind. of } x_1}\ G$$

The equivalence.

y independent of x

$$Q x.\ F(x) \circ G \equiv Q x.(F(x) \circ G)$$

$\circ$: $\vee, \wedge$

$\equiv Q_1 x_1 (Q_2 x_2 \ldots Q_i x_i\, F \circ R_1 y_1 \ldots R_j y_j\, G)$

$\equiv Q_1 x_1 (R_1 y_1 (\ldots))$ ← recurse

$\equiv R_1 y_1 (Q_1 x_1 \ldots Q_i x_i\, F \circ R_2 y_2 \ldots R_j y_j\, G)$

What do we take out?

$\exists$: as few $\forall$ ahead! ✓$\exists, \forall$ / $\forall, \forall$

little bit tricky

Q : $\cancel{\exists}\ \cancel{\forall}\ \exists \exists \forall$ $\qquad$ R: $\cancel{\forall}\ \cancel{\exists}\ \cancel{\exists}\ \forall \exists$

$\forall \forall \exists$ $\qquad$ $\forall \exists \exists$

$\forall \exists \exists \forall \forall \exists$ (1 1 3) | $\forall \forall \exists \forall \exists \exists$ (2 3 3)

Will get back.

We need some interleaving of Q & R, will update once I figure out how to prove what is optimal

16.12 Set theory axioms.

$\forall x, y, z\ (z \in x \Leftrightarrow z \in y)\quad x \simeq y$ — set equality, binary predicate

$\in$: inclusion of element

$\forall x, y\ (x \subseteq y \Leftrightarrow \forall z.(z \in x \Rightarrow z \in y))$

$\subseteq$: subset

$\forall x, y, z\ (z \in x \setminus y \Leftrightarrow (z \in x \wedge z \notin y))$

binary func.
set difference

$$\bigwedge \Sigma \wedge \neg \forall x, y. (x \subseteq y \Rightarrow \exists z. (y - z \approx x))$$

1) Rename apart: $x_i, y_i, z_i \quad i \in \{1,2,3,4\}$
2) NNF: push $\neg$ inside. Replace $\Leftrightarrow$ by $\wedge$ of $\Rightarrow$, by $\neg(\ )\vee(\ )$
3) prenex form (hopefully we'll figure out 16.10 and try it here later)
4) Skolemize*, to CNF. like propositional case.
5) Remove $\forall$

* the stuff inside the qf. will be ~ to propositional logic, only instead of atoms, you have predicates.

17.6

Substitutions and unifiers are SYNTACTIC concepts.

It takes some introspection to relate the English meaning of generality to substitutions, and the formal def'n.

$\sigma_1 \geq \sigma_2$ if $\exists \tau$
$$\sigma_2 = \sigma_1 \tau$$

general × specific.

Specific substitution?

Map each variable to a constant!

$\sigma_0 : \quad x_i \mapsto c_i$ ← const.

You can't compose σ_0 with anything

$t\sigma_0\tau \equiv (t\sigma_0)\tau$ ← acts only on variables

but everything here is a constant

Any subst. that maps ALL variables to an expression that involves only constants is a minimal general subst.

General just means that you don't commit to any information.

So intuitively, a most general subst. will just rename variables.

To prove σ is most general,

Take σ' (arbit). Find τ such that

$\sigma' = \sigma\tau$ $\qquad\qquad \sigma = id$

$\qquad\qquad\qquad \tau = \sigma'$

Prove: Maximum general substitutions are equiv. upto renaming

i.e. if σ_1, σ_2 most general

$\sigma_1 = \sigma_2 \tau_2 \qquad \tau_i$ are

$\sigma \ldots$

$\sigma_2 = \sigma_1 \tau_1$ 'RENAMINGS

Very similar to proof of Thm 17.1 earlier in slides.

For every term t

$$t\sigma_1 = t\sigma_1\tau_1\tau_2$$

$\sigma_1 = (\sigma_2)\tau_2$
$= \sigma_1\tau_1\tau_2$

For any subst. σ

$$|t| \leq |t\sigma|$$

lengths as strings

Var $\mapsto$ Var
Const
func(. .)

this is the only case length gets bigger

τ_1, τ_2 map vars to either vars or consts.

σ_1: Vars $\mapsto$ Terms

Y_1: Free variables in the image of σ_1

doing this in the context of CNF quantifiers have been eliminated

Γ: Vars $\setminus Y_1$

can assume τ_1 acts identically on Γ

$y \in Y_1$ $y = x\sigma_1$ for some x

$$y\tau_1\tau_2 = y$$

$$y\tau_1 \in \text{Vars}$$

$$y\tau_1 = z$$

1) substitution preserves string length const/vars

2) But if const, cannot map further

there is no $y' \in Y_1$ such that

$$y'\tau_1 = z$$

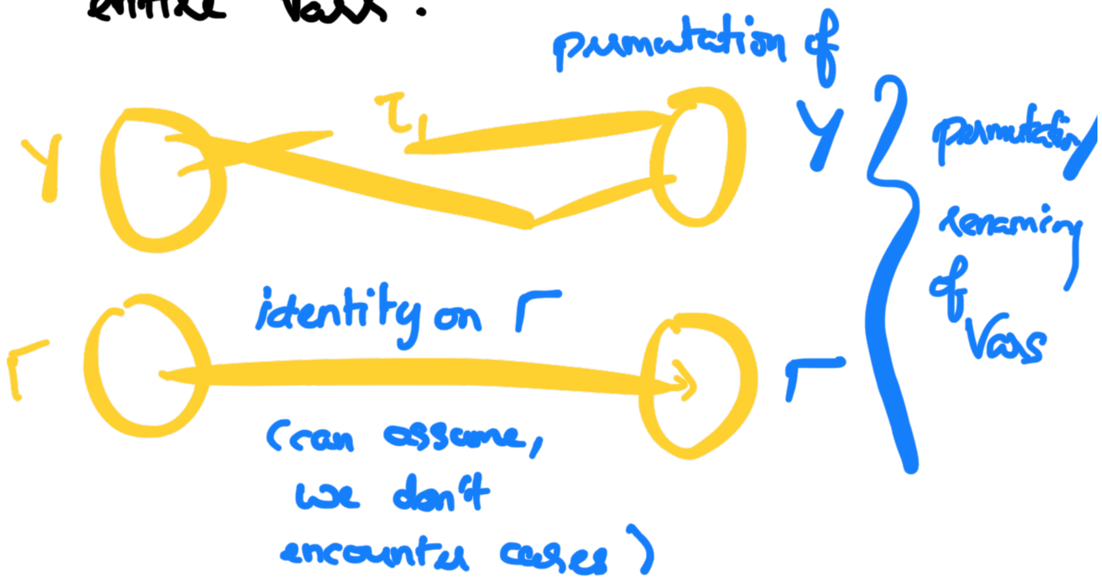
otherwise $y\tau_1\tau_2 = y'\tau_1\tau_2$

but $y\tau_1\tau_2 = y$, $y'\tau_1\tau_2 = y'$

τ_1 is a permutation of Y_1.
Similarly τ_2 is a permutation of Y_2

$$(\sigma_2 = \sigma_2\tau_2\tau_1)$$

The identity is trivially a max. substitution
so here $Y_1 =$ Vars if $\sigma_1 = id$
for τ_1, it is a renaming of
entire Vars.



17.7

Multiple unification

σ such that

$$t_1\sigma = t_2\sigma = \ldots = t_n\sigma$$

unification is SYNTACTIC

make n-ary function f

$$f(t_1, t_2, \ldots t_n) \quad f(t_2, t_3, \ldots, t_n, t_1)$$

F_1 F_2

find mgu σ of these terms

$$F_1\sigma = F_2\sigma$$

by defn

$$f(t_1\sigma, t_2\sigma, \dots t_n\sigma) = f(t_2\sigma, \dots t_n\sigma, t_1\sigma)$$

$$t_1\sigma = t_2\sigma = t_3\sigma = \dots = t_n\sigma$$

$$=$$

17.8 Concurrent unification

$f(t_1, \dots, t_n)$ $\quad$ $f(u_1, \dots u_n)$

σ

$$t_1\sigma = u_1\sigma$$

17.7

18.2 Formal proof

1. even(sum(twosq, b))
2. twosq = four
3. $\neg$zero(x) $\vee$ diff(four, x) = sum(four, x)
4. zero(b)

Assumpt

5. $\neg even(diff(twosq, b))$

RES $\dfrac{\neg A \vee C \quad B \vee D}{(C \vee D)\sigma}$ $\sigma = mgu(A, B)$

$\sigma: \{x \mapsto b\}$

6. $diff(four, b) = sum(four, b)$ Res 3,4 $\sigma: \{x \mapsto b\}$

In my set of clauses, I have a bunch of equalities, and I want to exploit equalities to make substitutions

Paramodulation!

PARA MOD $\dfrac{(s = t) \vee C \quad D(u)}{(C \vee D(t))\sigma}$ $\sigma = mgu(s, u)$

7. $even(sum(four, b))$ Paramod 2,1 $\sigma = id$

8. $\neg even(diff(four, b))$ Paramod 2,5 $\sigma = id$

9. $even(diff(four, b))$ Paramod 6,7 $\sigma = id$

10. $\perp$ Res 8,9

18.3

1. $P(f(a))$ 2. $a = c$ 3. $\neg Q(x, x) \vee (f(c) = f(d))$

4. $Q(b, b)$ 5. $\neg P(f(d))$

6. $f(c) = f(d)$ Res. 3, 4 $\sigma : \{x \mapsto b\}$

7. $P(f(c))$ Paramod 2, 1 σ: id

8. $P(f(d))$ oops Paramod 6, 7

9. $\bot$

$\forall\forall\exists$ $\underline{\forall}\,\underset{\sim}{\exists\exists}$