3.4 Resolution in Predicate Logic and Unification

Monday, 27 April, 2015 8:30

Ex 341 Up to now: before performing resolution, we have to instantiate variables by ground terms. This instantiation does not only have to enable the next resolution step, but one has to guess the right instantiation which also allows all needed future resolution steps.

We need an inst. for X and Y such that

$p(X)$ and $p(f(Y))$ become equal.

(i.e. we have to unify $p(X)$ and $p(f(Y))$)

But this instantiation should also allow future resolution steps (e.g., between $q(\dots)$ and $\neg q(\dots)$).

Solution: do not instantiate variables by ground terms, but also allow instantiations by arbitrary terms. Only look for most general unifiers, i.e., only instantiate them in such a way that the next resolution step is possible)

In the example: Finally one uses $\{Y/a\}$ to derive $\Box$.

Def 342 (Unification)

A clause $K = \{L_1, \dots, L_n\}$ is unifiable iff there exists a substitution σ such that $\sigma(L_1) = \dots = \sigma(L_n)$ (i.e.,

$|\underbrace{\sigma(K)}_{\{\sigma(L_1),\dots,\sigma(L_n)\}}| = 1$). Such a subst. σ is a <u>unifier</u> of K.

A unifier σ is a <u>most general unifier</u> (mgu) iff for any unifier σ' there exists a subst. δ such that

$$\sigma'(X) = \delta(\sigma(X)) \quad \text{for all } X \in \mathcal{V}.$$

In the example: $K = \{p(X), p(f(Y))\}$

mgu $\sigma = \{X/f(Y)\}$ $\qquad |\sigma(K)| = |\{\sigma(p(X)), \sigma(p(f(Y)))\}| = |\{p(f(Y))\}| = 1$

alternative unifier $\sigma' = \{X/f(a), Y/a\}$

We have $\sigma' = \delta \circ \sigma$

for $\delta = \{Y/a\}$

<u>Observations</u>:

- If a clause is unifiable, then it also has an mgu.
- The mgu is unique up to variable renaming.

  Ex: $\{p(X), p(Y)\}$

  mgu $\sigma = \{X/Y\}$ or $\sigma' = \{Y/X\}$
- It is decidable whether a clause is unifiable and the mgu is computable.
- First unification algorithm by J. Robinson (1965).

  ↑ inventor of resolution

<u>Ex 343</u>

• Try to unify $\{q(f(X,Y)),\ q(g(X,Y))\}$

$\sigma = \emptyset$

<u>clash failure</u>

• Try to unify $\{q(X),\ q(h(X))\}$

$\sigma = \emptyset$

<u>occur failure</u>

• Try to unify $\{p(f(Z, g(a,Y)), h(Z)),\ p(f(f(U,V),W), h(f(a,Y)))\}$

$\sigma = \emptyset$

$\sigma = \{Z/f(U,V)\}$

$\{p(f(f(U,V), g(a,Y)), h(f(U,V))),\ p(f(f(U,V), W), h(f(a,Y)))\}$

$\sigma = \{W/g(a,Y)\} \circ \{Z/f(U,V)\} = \{W/g(a,Y),\ Z/f(U,V)\}$

$\{p(f(f(U,V), g(a,Y)), h(f(U,V))),\ p(f(f(U,V), g(a,Y)), h(f(a,Y)))\}$

$\sigma = \{U/a\} \circ \{W/g(a,Y),\ Z/f(U,V)\} = \{U/a,\ W/g(a,Y),\ Z/f(a,V)\}$

$\{p(f(f(a,V), g(a,Y)), h(f(a,V))),\ p(f(f(a,V), g(a,Y)), h(f(a,Y)))\}$

$\sigma = \{Y/V\} \circ \ldots = \{Y/V,\ U/a,\ W/g(a,V),\ Z/f(a,V)\}$

$\{p(\ldots),\ p(\ldots)\}$

are the same now

Friday (May 8): 2 lectures (lecture instead of exerc. course)

Monday (May 11): ex. course instead of lecture

Thm 3.44 (Termination + Soundness of Unif. Alg.)

The unif. alg. terminates for every clause K and it is sound, i.e., it returns an mgu for K iff K is unifiable.

Proof: The alg. terminates because the number of variables in the clause decreases in each loop iteration.

If the alg. returns a subst. σ, then σ is a unifier of K (since it checks $|\sigma(K)| = 1$ in step 2).

Thus: if K is not unifiable

$\leadsto$ alg can't return a subst. σ

$\leadsto$ alg stops with failure. (since alg. terminates).

It remains to prove:

If K is unifiable, then alg. returns a mgu σ.

Let $m \geq 0$ be the number of loop iterations of the alg. for the input K. For every $0 \leq i \leq m$, let σ_i be the value of σ after the i-th loop iteration.

We prove the following for all $0 \le i \le m$:

For every unifier σ' of K, we have $\sigma' = \sigma' \circ \sigma_i$. $(\forall)$

This implies the soundness of the alg. if K is unifiable:

- If the alg would stop with failure in the $(m+1)$-th loop iteration, then $\sigma_m(K)$ would not be unifiable.
  But $(\forall)$ implies: $\sigma' = \sigma' \circ \sigma_m$ and there exists a unifier σ' of K.

  $$|\sigma'(K)| = 1$$
  $$\curvearrowright |\sigma'(\sigma_m(K))| = 1$$
  $\curvearrowright$ σ' is a unifier of $\sigma_m(K)$. $\lightning$
- So the alg. has to stop with success
  $\curvearrowright |\sigma_m(K)| = 1$, i.e., σ_m is a unifier.
  Now $(\forall)$ implies that for every unifier σ' there exists a subst δ (viz. $\delta = \sigma'$) such that:

  $$\sigma' = \underbrace{\delta}_{\sigma'} \circ \sigma_m$$

  $\curvearrowright$ σ_m is mgu of K.

Now we prove the following for all $0 \le i \le m$ by induction on i:

For every unifier σ', we have $\sigma' = \sigma' \circ \sigma_i$ $(\forall)$

<u>Ind Base</u>: $i = 0$

$\sigma_0 = \emptyset$ $\curvearrowright$ $\sigma' = \sigma' \circ \sigma_0$ holds for all substitutions σ'. ✓

<u>Ind Step</u>: $i > 0$

Ind. Hypothesis: $\sigma' = \sigma' \circ \sigma_{i-1}$

To unify $\sigma_{i-1}(K)$ one has to replace a var. X by a term t

in Step 6. Thus: $\sigma_i = \{X/t\} \circ \sigma_{i-1}$.

We have:

$$\begin{aligned}
&\sigma' \circ \sigma_i \\
= \;& \underbrace{\sigma' \circ \{X/t\}}_{\sigma'} \circ \sigma_{i-1} \qquad (\text{by def. of } \sigma_i) \\
= \;& \sigma' \circ \sigma_{i-1} \\
= \;& \sigma' \qquad (\text{by the ind. hyp.})
\end{aligned}$$

Reason for $\sigma' = \sigma' \circ \{X/t\}$:

For $Y \neq X$: $\sigma'(Y) = (\sigma' \circ \{X/t\})(Y)$

For X: $(\sigma' \circ \{X/t\})(X) = \sigma'(t) = \sigma'(X)$

Reason: σ' is also a unifier of $\sigma_{i-1}(K)$

(since $|\sigma'(\sigma_{i-1}(K))| \underset{\text{by ind. hyp}}{=} |\sigma'(K)| = 1$)

Therefore, σ' must make X and t equal. $\boxtimes$

<u>Def 345</u> (Resolution in Predicate Logic)

Let K_1, K_2 be clauses. Then a clause R is <u>resolvent</u> of K_1 and K_2 iff:

- There exist variable renamings ν_1, ν_2 such that $\nu_1(K_1)$ and $\nu_2(K_2)$ have no common variables.

  ok, since clause stands for universally quantified disjunction of its literals (i.e.: renaming of bound variables)

- There exist literals $L_1, \ldots, L_m \in \nu_1(K_1)$ and $L_1', \ldots, L_n' \in \nu_2(K_2)$ with

$L'_1, \dots, L'_n \in \nu_2(K_2)$ with

(i.e.: renaming of bound variables)

$n, m \geq 1$ such that

$\{\overline{L_1}, \dots, \overline{L_m}, L'_1, \dots, L'_n\}$ are unifiable with mgu σ

- $R = \sigma\Big(\big(\nu_1(K_1) \setminus \{L_1, \dots, L_m\}\big) \cup \big(\nu_2(K_2) \setminus \{L'_1, \dots, L'_n\}\big)\Big)$

As before, we define the following for a clause set $\mathcal{K}$:

$$Res(\mathcal{K}) = \mathcal{K} \cup \{R \mid R \text{ is resolvent of 2 clauses in } \mathcal{K}\}$$

$$Res^0(\mathcal{K}) = \mathcal{K}$$

$$Res^{n+1}(\mathcal{K}) = Res(Res^n(\mathcal{K})) \quad \text{for all } n \geq 0$$

$$Res^*(\mathcal{K}) = \bigcup_{n \geq 0} Res^n(\mathcal{K})$$

Clearly, propositional resolution is a special case of this form of resolution.

<u>Ex 346</u>

K_1: $\{p(f(X)), \neg q(Z), p(Z)\}$

K_2: $\{\neg p(X), r(g(X))\}$

$\nu_2(K_2)$: $\{\neg p(U), r(g(U))\}$

$\{\neg q(f(X)), r(g(f(X)))\}$

Variable renaming:

$\nu_1 = \cancel{\emptyset}$

$\nu_2 = \{X/U, U/X\}$ (must be injective)

$L_1 = p(f(X))$ $\qquad$ $L'_1 = \neg p(U)$

$L_2 = p(Z)$

$\sigma = mgu(\{\neg p(f(X)), \neg p(Z), \neg p(U)\}) = \{Z/f(X), U/f(X)\}$

Resolution in pred. logic is also sound + complete, i.e.: $\mathcal{K}$ is unsatisfiable iff $\square \in Res^*(\mathcal{K})$

Soundness can be shown in a similar way as in prop. logic.

Lemma 3.47 (Resolution Lemma for Pred. Logic)

Let $\mathcal{K}$ be a clause set. If $K_1, K_2 \in \mathcal{K}$ and R is a resolvent of K_1 and K_2, then $\mathcal{K}$ and $\mathcal{K} \cup \{R\}$ are equivalent.

Proof: similar to the propositional resolution lemma (Lemma 3.3.6) $\boxed{\varepsilon}$

This implies soundness of resolution:

$$\square \in Res^*(\mathcal{K})$$

$$\curvearrowright \quad \square \in Res^n(\mathcal{K}) \text{ for some } n \geq 0$$

$$\curvearrowright \quad Res^n(\mathcal{K}) \text{ is unsatisfiable}$$

$$\curvearrowright \quad \mathcal{K} \text{ is unsatisfiable}$$

(since the resolution lemma implies that $\mathcal{K}$ and $Res(\mathcal{K})$ are equivalent. Thus, by induction on n one can show that $\mathcal{K}$ and $Res^n(\mathcal{K})$ are equivalent.)

Now: Show completeness of resolution in pred. logic.

Goal: Use completeness of prop. resolution to show completeness of full resolution.

Solution: Lifting Lemma
shows how to "lift" a resolution proof from propositional to predicate logic.

Lemma 348 (Lifting Lemma)
Let K_1, K_2 be clauses with ground instances K_1', K_2'.
If R' is a (propositional) resolvent of K_1' and K_2', then there exists a resolvent R of K_1 and K_2 such that R' is a ground instance of R.

$$\begin{tikzcd}
K_1 \arrow[d, "\text{ground instance}"] & & K_2 \arrow[d] \\
K_1' \arrow[dr, dash] & & K_2' \arrow[dl, dash] \\
& R' &
\end{tikzcd}
\quad \rightsquigarrow \quad
\begin{tikzcd}
K_1 \arrow[dr, dash] & & K_2 \arrow[dl, dash] \\
& R \arrow[d, "\text{ground instance}"] & \\
& R' &
\end{tikzcd}$$

Ex 349 to illustrate the lifting lemma

K_1 – $\{p(f(X)), \neg q(Z), p(Z)\}$ $\qquad$ K_2 – $\{\neg p(X), r(g(X))\}$

$\downarrow$ ground subst $\{X/a, Z/f(a)\}$ $\qquad$ $\downarrow$ $\{X/f(a)\}$

K_1' – $\{p(f(a)), \neg q(f(a))\}$ $\qquad$ K_2' – $\{\neg p(f(a)), r(g(f(a)))\}$

$\{\neg q(f(a)), r(g(f(a)))\}$ – R'

The lifting lemma states that one could instead perform resolution on K_1, K_2 and obtain a resolvent R such that R' is a ground instance of R:

$\{p(f(X)), \neg q(Z), p(Z)\}$ – K_1 $\qquad$ $\{\neg p(X), r(g(X))\}$ – K_2

$\{\neg q(f(X)), \ldots\}$ – R

$\{\neg q(f(X)), r(g(f(X)))\}$ – K

$\downarrow$ ground inst. with $\{X/a\}$

$\{\neg q(f(a)), r(g(f(a)))\}$ – R'

Proof of the Lifting Lemma:

$$\begin{tikzcd}
K_1 \arrow[d] & & K_2 \arrow[d] \\
K_1' \arrow[dr, dash] & & K_2' \arrow[dl, dash] \\
& R' &
\end{tikzcd}$$

Let ν_1, ν_2 be var. renamings such that $\nu_1(K_1)$ and $\nu_2(K_2)$ are variable-disjoint.

Then K_1', K_2' are also ground instances of $\nu_1(K_1)$ and $\nu_2(K_2)$.

Since $\nu_1(K_1)$ and $\nu_2(K_2)$ are variable-disjoint, one can use the same ground subst. σ:

$$\sigma(\nu_1(K_1)) = K_1' \quad \text{and} \quad \sigma(\nu_2(K_2)) = K_2'$$

Since R' is resolvent of K_1' and K_2', there is a literal

$L \in K_1'$ with $\overline{L} \in K_2'$ and

$$R' = (K_1' \setminus \{L\}) \cup (K_2' \setminus \{\overline{L}\})$$

Let $L_1, \dots, L_m$ be all literals from $\nu_1(K_1)$ that are mapped to L by σ (i.e., $\sigma(L_1) = \dots = \sigma(L_m) = L$).

Let $L_1', \dots, L_n'$ be all literals from $\nu_2(K_2)$ that are mapped to $\overline{L}$ by σ (i.e., $\sigma(L_1') = \dots = \sigma(L_n') = \overline{L}$).

We must have $m, n \geq 1$.

Since σ is a unifier of $\{\overline{L_1}, \dots, \overline{L_m}, L_1', \dots, L_n'\}$, there also exists a mgu σ'.

Therefore, K_1 and K_2 have a resolvent

$$R = \sigma'\Big(\big(\nu_1(K_1) \setminus \{L_1,\dots,L_m\}\big) \cup \big(\nu_2(K_2) \setminus \{L_1',\dots,L_n'\}\big) \Big).$$

Thus:

$$\begin{tikzcd}[column sep=small]
K_1 \arrow[d, "\sigma"] & & K_2 \arrow[d, "\sigma"] \\
K_1' \arrow[dr] & & K_2' \arrow[dl] \\
& R' &
\end{tikzcd}
\quad \rightsquigarrow \quad
\begin{tikzcd}[column sep=small]
K_1 \arrow[dr, dash] & & K_2 \arrow[dl, dash] \\
& R \arrow[d, "\delta"] & \\
& R' &
\end{tikzcd}$$

This still has to be shown.

Since σ is a unifier of $\{\overline{L}_1,\dots,\overline{L}_m, L_1',\dots,L_n'\}$ and σ' is a mgu, there exists a substitution δ such that

$$\sigma = \delta \circ \sigma'$$

We now show that R' is indeed an instance of R:

$$\begin{aligned}
R' &= (K_1' \setminus \{L\}) \cup (K_2' \setminus \{\overline{L}\}) \\
&= \big(\sigma(\nu_1(K_1)) \setminus \{L\}\big) \cup \big(\sigma(\nu_2(K_2)) \setminus \{\overline{L}\}\big) \\
&= \sigma\Big(\big(\nu_1(K_1) \setminus \{L_1,\dots,L_m\}\big) \cup \big(\nu_2(K_2) \setminus \{L_1',\dots,L_n'\}\big)\Big)
\end{aligned}$$

(since $L_1,\dots,L_m$ are all literals from $\nu_1(K_1)$ that are mapped to L by σ)

$$\begin{aligned}
&= \delta\Big(\underbrace{\sigma'\big((\nu_1(K_1) \setminus \{L_1,\dots,L_m\}) \cup (\nu_2(K_2) \setminus \{L_1',\dots,L_n'\})\big)}_{R}\Big) \\
&= \delta(R).
\end{aligned}$$

$\boxtimes$

Thm 3.4.10 (Soundness + Completeness of Resolution in Pred. Logic)

Let $\mathcal{K}$ be a finite clause set. Then

$\mathcal{K}$ is unsatisfiable iff $\square \in Res^*(\mathcal{K})$.

<u>Proof</u>: "$\Leftarrow$" (Soundness): direct consequence of the resolution lemma 3.47.

"$\Rightarrow$" (Completeness):

$\mathcal{K}$ is unsatisfiable

$\Leftrightarrow$ by Thm 3.27: $\underbrace{\text{Herbrand-expansion of } \mathcal{K}}$ is also unsatisfiable

The set of clauses containing all ground instances of clauses from $\mathcal{K}$, i.e.,

$\{ \sigma(K) \mid K \in \mathcal{K}, \sigma(K) \text{ contains no variables} \}$

$\Leftrightarrow$ by Thm 3.37 (completeness of resolution in propositional logic):

One can deduce $\square$ from the Herbrand-exp. of $\mathcal{K}$, i.e.,

$\square \in Res^*(\{\sigma(K) \mid K \in \mathcal{K}, \sigma(K) \text{ has no variables}\})$.

$\Leftrightarrow$ There exists a sequence of ground clauses

$K_1', \dots, K_m'$ with $K_m' = \square$ and

for all $1 \le i \le m$:

- K_i' is a ground instance of a clause from $\mathcal{K}$ or
- K_i' is resolvent of K_j' and K_k' for $j, k < i$

Now we "lift" this resolution proof to predicate logic, i.e., we create a sequence $K_1, \dots, K_m$ where each

- K_i' is an instance of K_i and
- $K_i \in \mathrm{Res}^*(\mathcal{K})$

Definition of K_i:

- if K_i' is a ground instance of some $K \in \mathcal{K}$, then choose $K_i := K$
- otherwise, K_i' is a resolvent of K_j' and K_k' for $j, k < i$. We already defined K_j, K_k such that K_j' and K_k' are instances of K_j and K_k resp., and $K_j, K_k \in \mathrm{Res}^*(\mathcal{K})$.

$$\begin{tikzcd}[column sep=small]
K_j \arrow[d] & & K_k \arrow[d] \\
K_j' \arrow[dr, dash] & & K_k' \arrow[dl, dash] \\
& K_i' &
\end{tikzcd}
\qquad
\begin{array}{c} \curvearrowright \\ \text{Lifting} \\ \text{Lemma} \\ 3.4.8 \end{array}
\qquad
\begin{tikzcd}[column sep=small]
K_j \arrow[dr, dash] & & K_k \arrow[dl, dash] \\
& K_i \arrow[d] & \\
& K_i' &
\end{tikzcd}$$

Lifting lemma states that there exists a resolvent of K_j and K_k such that K_i' is an instance of this resolvent. Choose K_i to be this resolvent.

$\Rightarrow K_i \in \mathrm{Res}^*(\mathcal{K})$

Thus: K_m' is an instance of K_m and $K_m \in \mathrm{Res}^*(\mathcal{K})$

$$\begin{array}{c} K_m' \\ | \\ \Box \end{array}$$

$\Rightarrow K_m = \Box$ and $\Box \in \mathrm{Res}^*(\mathcal{K})$. $\blacksquare$

Now we can improve our algorithm to check entailment by using resolution in pred. logic.

Alg is a semi-decision procedure:

If $\{\varphi_1, \ldots, \varphi_n\} \models \varphi$, then the alg. terminates and returns "true".

If $\{\varphi_1, \ldots, \varphi_n\} \not\models \varphi$, then the alg may not terminate. But if it terminates, then it returns "false".

Alg. is feasible in practice for small problems, but still too inefficient in general.

Problem: one has to generate all resolvents repeatedly (one also has to resolve clauses that were created by earlier resolution steps).

Goal: Restrict resolution (i.e., do not create all possible resolvents) without losing completeness