Lecture 12

Resolution for predicate logic

Unification, resolution

Dr Christoph Haase
University of Oxford
(with small changes by Javier Esparza)

Drawbacks of ground resolution

- Ground resolution good for showing semi-decidability, bad for practical purposes
- Requires "looking ahead" to see which ground terms will be needed
- Want to instantiate ground terms "by need"

Drawbacks of ground resolution

- Ground resolution good for showing semi-decidability, bad for practical purposes
- Requires "looking ahead" to see which ground terms will be needed
- Want to instantiate ground terms "by need"

Today:

- Predicate-logic version of resolution
- Forms basis of programming language Prolog

Substitution

Key concept **substitution**:

- Used to replace variables by σ-terms
- More general: substitution is function θ mapping σ-terms to σ-terms such that

$$c\theta = c$$
$$f(t_1, \ldots, t_k)\theta = f(t_1\theta, \ldots, t_k\theta)$$

Substitution

Key concept **substitution**:

- Used to replace variables by σ-terms
- More general: substitution is function θ mapping σ-terms to σ-terms such that

$$c\theta = c$$
$$f(t_1, \ldots, t_k)\theta = f(t_1\theta, \ldots, t_k\theta)$$

- Extends canonically to arbitrary formulas, e.g. $P(x, c)\theta = P(x\theta, c\theta)$, etc.
- Denote by $\theta \cdot \theta'$ substitution first performing θ and then θ'

Substitution

Key concept **substitution**:

- Used to replace variables by σ-terms
- More general: substitution is function θ mapping σ-terms to σ-terms such that

$$c\theta = c$$
$$f(t_1, \ldots, t_k)\theta = f(t_1\theta, \ldots, t_k\theta)$$

- Extends canonically to arbitrary formulas, e.g. $P(x, c)\theta = P(x\theta, c\theta)$, etc.
- Denote by $\theta \cdot \theta'$ substitution first performing θ and then θ'

Example

Let $\theta = [f(y)/x]$, $\theta' = [g(c, z)/y]$, and $P(x, c)$ be an atomic formula,

Substitution

Key concept **substitution**:

- Used to replace variables by σ-terms
- More general: substitution is function θ mapping σ-terms to σ-terms such that

$$c\theta = c$$
$$f(t_1, \ldots, t_k)\theta = f(t_1\theta, \ldots, t_k\theta)$$

- Extends canonically to arbitrary formulas, e.g. $P(x, c)\theta = P(x\theta, c\theta)$, etc.
- Denote by $\theta \cdot \theta'$ substitution first performing θ and then θ'

Example

Let $\theta = [f(y)/x]$, $\theta' = [g(c, z)/y]$, and $P(x, c)$ be an atomic formula, then $P(x, c)\theta = P(x\theta, c\theta) = P(f(y), c)$, $\theta \cdot \theta' = [f(g(c, z))/x]$, and $P(x, c)(\theta \cdot \theta') = P(f(g(c, z)), c)$.

Substitution

- For sets of literals $D = \{L_1, \ldots, L_k\}$, define $D\theta := \{L_1\theta, \ldots, L_k\theta\}$
- θ **unifies** D if $D\theta = \{L\}$ for some literal L

Substitution

- For sets of literals $D = \{L_1, \ldots, L_k\}$, define $D\theta := \{L_1\theta, \ldots, L_k\theta\}$
- θ **unifies** D if $D\theta = \{L\}$ for some literal L

Example

We have that $\theta = [f(a)/x][a/y]$ unifies $\{P(x), P(f(y))\}$ since

$$\{P(x)\theta, P(f(y))\theta\} = \{P(f(a)), P(f(a))\} = \{P(f(a))\},$$

but $\theta' = [f(y)/x]$ is also unifier.

Substitution

- For sets of literals $D = \{L_1, \dots, L_k\}$, define $D\theta := \{L_1\theta, \dots, L_k\theta\}$
- θ **unifies** D if $D\theta = \{L\}$ for some literal L

Example

We have that $\theta = [f(a)/x][a/y]$ unifies $\{P(x), P(f(y))\}$ since

$$\{P(x)\theta, P(f(y))\theta\} = \{P(f(a)), P(f(a))\} = \{P(f(a))\},$$

but $\theta' = [f(y)/x]$ is also unifier. Note that $\theta = \theta' \cdot [a/y]$.

Substitution

- For sets of literals $D = \{L_1, \ldots, L_k\}$, define $D\theta := \{L_1\theta, \ldots, L_k\theta\}$
- θ **unifies** D if $D\theta = \{L\}$ for some literal L

Example

We have that $\theta = [f(a)/x][a/y]$ unifies $\{P(x), P(f(y))\}$ since

$$\{P(x)\theta, P(f(y))\theta\} = \{P(f(a)), P(f(a))\} = \{P(f(a))\},$$

but $\theta' = [f(y)/x]$ is also unifier. Note that $\theta = \theta' \cdot [a/y]$.

Definition

We call θ a **most general unifier (mgu)** of D if θ is a unifier and for all other unifiers θ' there is unifier θ'' such that $\theta' = \theta \cdot \theta''$.

Substitution

- For sets of literals $D = \{L_1, \ldots, L_k\}$, define $D\theta := \{L_1\theta, \ldots, L_k\theta\}$
- θ **unifies** D if $D\theta = \{L\}$ for some literal L

Example

We have that $\theta = [f(a)/x][a/y]$ unifies $\{P(x), P(f(y))\}$ since

$$\{P(x)\theta, P(f(y))\theta\} = \{P(f(a)), P(f(a))\} = \{P(f(a))\},$$

but $\theta' = [f(y)/x]$ is also unifier. Note that $\theta = \theta' \cdot [a/y]$.

Definition

We call θ a **most general unifier (mgu)** of D if θ is a unifier and for all other unifiers θ' there is unifier θ'' such that $\theta' = \theta \cdot \theta''$.

Note:

- Not unique in general but unique up to renaming of variables
- Sometimes does not exist: $\{P(f(x)), P(g(x))\}$, $\{P(x), P(f(x))\}$

Most general unifier

Theorem (Unification Theorem)

A unifiable set of literals D has a most general unifier.

Most general unifier

Theorem (Unification Theorem)

A unifiable set of literals D has a most general unifier.

Proof.

Unification Algorithm
Input: Set of literals D
Output: Either a most general unifier of D or "fail"

$\theta :=$ identity substitution
while θ is not a unifier of D **do**
begin
  pick two distinct literals in $D\theta$ and
    find left-most positions at which they differ
  if one of the corresponding sub-terms is variable x and
    other term t not containing x
  then $\theta := \theta \cdot [t/x]$ **else** output "fail" and halt
end

□

Example

Example

Consider input $D = \{P(x, y), P(f(z), x)\}$:

$$\begin{aligned}
&\{P(\underline{x}, y), P(\underline{f}(z), x)\}, \text{ apply } [f(z)/x] \\
&\{P(f(z), \underline{y}), P(f(z), \underline{f}(z))\}, \text{ apply } [f(z)/y] \\
&\{P(f(z), f(z))\}
\end{aligned}$$

Thus $[f(z)/x][f(z)/y]$ is a most general unifier of the set D.

Example

Example

Consider input $D = \{P(x, y), P(f(z), x)\}$:

$$\{P(\underline{x}, y), P(\underline{f}(z), x)\},\ \text{apply } [f(z)/x]$$
$$\{P(f(z), \underline{y}), P(f(z), \underline{f}(z))\},\ \text{apply } [f(z)/y]$$
$$\{P(f(z), f(z))\}$$

Thus $[f(z)/x][f(z)/y]$ is a most general unifier of the set D.

Exercise

Unifiable?			Yes	No
	$P(f(x))$	$P(g(y))$		
	$P(x)$	$P(f(y))$		
	$P(x, f(y))$	$P(f(u), z)$		
	$P(x, f(y))$	$P(f(u), f(z))$		
	$P(x, f(x))$	$P(f(y), y)$		
	$P(x, g(x), g^2(x))$	$P(f(z), w, g(w))$		
$P(x, f(y))$	$P(g(y), f(a))$	$P(g(a), z)$		

Proof of unification theorem

Termination: at each loop iteration the algorithm either halts, or a variable x gets replaced by a term in which x does not occur

Proof of unification theorem

Termination: at each loop iteration the algorithm either halts, or a variable *x* gets replaced by a term in which *x* does not occur

Loop invariant: for any unifier θ' of $D\theta$, we have $\theta' = \theta \cdot \theta'$.

Proof of unification theorem

Termination: at each loop iteration the algorithm either halts, or a variable *x* gets replaced by a term in which *x* does not occur

Loop invariant: for any unifier θ' of $D\theta$, we have $\theta' = \theta \cdot \theta'$.

- Holds before entering while loop because θ is the identity substitution.

Proof of unification theorem

Termination: at each loop iteration the algorithm either halts, or a variable *x* gets replaced by a term in which *x* does not occur

Loop invariant: for any unifier θ' of $D\theta$, we have $\theta' = \theta \cdot \theta'$.

- Holds before entering while loop because θ is the identity substitution.
- Let θ' be unifier. Assume $\theta' = \theta \cdot \theta'$ holds at **begin** and algorithm does not halt. We show $\theta' = \theta \cdot \theta'$ holds again at **end**.
- Since algorithm does not halt, we find *x* and *t* in $D\theta$.

Proof of unification theorem

Termination: at each loop iteration the algorithm either halts, or a variable x gets replaced by a term in which x does not occur

Loop invariant: for any unifier θ' of $D\theta$, we have $\theta' = \theta \cdot \theta'$.

- Holds before entering while loop because θ is the identity substitution.
- Let θ' be unifier. Assume $\theta' = \theta \cdot \theta'$ holds at **begin** and algorithm does not halt. We show $\theta' = \theta \cdot \theta'$ holds again at **end**.
- Since algorithm does not halt, we find x and t in $D\theta$.
- Since θ' is unifier of $D\theta$, we have $t\theta' = x\theta'$.

Proof of unification theorem

Termination: at each loop iteration the algorithm either halts, or a variable x gets replaced by a term in which x does not occur

Loop invariant: for any unifier θ' of $D\theta$, we have $\theta' = \theta \cdot \theta'$.

- Holds before entering while loop because θ is the identity substitution.
- Let θ' be unifier. Assume $\theta' = \theta \cdot \theta'$ holds at **begin** and algorithm does not halt. We show $\theta' = \theta \cdot \theta'$ holds again at **end**.
- Since algorithm does not halt, we find x and t in $D\theta$.
- Since θ' is unifier of $D\theta$, we have $t\theta' = x\theta'$.
- So $\theta' = [t/x] \cdot \theta'$, hence

$$(\theta \cdot [t/x]) \cdot \theta' = \theta \cdot ([t/x] \cdot \theta') = \theta \cdot \theta' = \theta' \ .$$

Proof of unification theorem

Termination: at each loop iteration the algorithm either halts, or a variable x gets replaced by a term in which x does not occur

Loop invariant: for any unifier θ' of $D\theta$, we have $\theta' = \theta \cdot \theta'$.

- Holds before entering while loop because θ is the identity substitution.
- Let θ' be unifier. Assume $\theta' = \theta \cdot \theta'$ holds at **begin** and algorithm does not halt. We show $\theta' = \theta \cdot \theta'$ holds again at **end**.
- Since algorithm does not halt, we find x and t in $D\theta$.
- Since θ' is unifier of $D\theta$, we have $t\theta' = x\theta'$.
- So $\theta' = [t/x] \cdot \theta'$, hence

$$(\theta \cdot [t/x]) \cdot \theta' = \theta \cdot ([t/x] \cdot \theta') = \theta \cdot \theta' = \theta' \ .$$

- The assignment $\theta := \theta \cdot [t/x]$ establishes $\theta' = \theta \cdot \theta'$ again.

After termination: θ is unifier because of the loop condition, and loop invariant implies θ is mgu.

Resolution

For set of literals D, $\overline{D}$ denotes complement of all literals in D.

Resolution

For set of literals D, $\overline{D}$ denotes complement of all literals in D.

Definition (Resolution)

Let C_1, C_2 be clauses with no variables in common.

R is a **resolvent** of C_1 and C_2 if there are $D_1 \subseteq C_1$ and $D_2 \subseteq C_2$ such that $D_1 \cup \overline{D_2}$ has mgu θ and

$$R = (C_1\theta \setminus \{L\}) \cup (C_2\theta \setminus \{\overline{L}\})$$

with $L = D_1\theta$ and $\overline{L} = D_2\theta$.

Resolution

For set of literals D, $\overline{D}$ denotes complement of all literals in D.

Definition (Resolution)

Let C_1, C_2 be clauses with no variables in common.

R is a **resolvent** of C_1 and C_2 if there are $D_1 \subseteq C_1$ and $D_2 \subseteq C_2$ such that $D_1 \cup \overline{D_2}$ has mgu θ and

$$R = (C_1\theta \setminus \{L\}) \cup (C_2\theta \setminus \{\overline{L}\})$$

with $L = D_1\theta$ and $\overline{L} = D_2\theta$.

Let C_1, C_2 be clauses with variables in common.

R is resolvent if there are renamings θ_1, θ_2 such that $C_1\theta_1, C_2\theta_2$ have no variables in common, and R is resolvent of $C_1\theta_1$ and $C_2\theta_2$.

Example

Example

Given signature with constant symbol e, unary function symbols f and g, and ternary predicate symbol P, compute resolvent of

$$C_1 = \{\neg P(f(e), x, f(g(e)))\} \text{ and } C_2 = \{\neg P(x, y, z), P(f(x), y, f(z))\}$$

as in the figure above.

Example

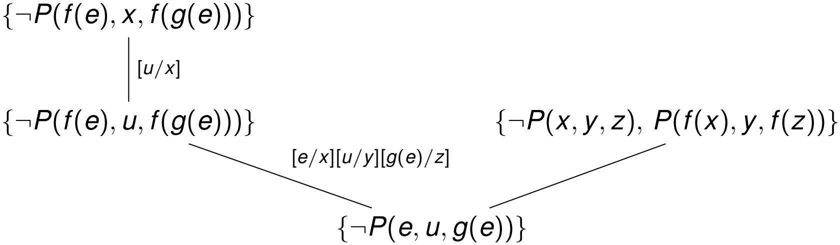


Figure: First-order resolution example

Example

Given signature with constant symbol e, unary function symbols f and g, and ternary predicate symbol P, compute resolvent of

$$C_1 = \{\neg P(f(e), x, f(g(e)))\} \text{ and } C_2 = \{\neg P(x, y, z), P(f(x), y, f(z))\}$$

as in the figure above.

Exercise

Have the following pairs of predicate clauses a resolvent?
How many resolvents are there?

C_1	C_2	Resolvents
$\{P(x), Q(x,y)\}$	$\{\neg P(f(x))\}$	
$\{Q(g(x)), R(f(x))\}$	$\{\neg Q(f(x))\}$	
$\{P(x), P(f(x))\}$	$\{\neg P(y), Q(y,z)\}$	

Predicate-resolution derivation

Use resolution in order to derive clause C from set of clauses F:

- Sequence of clauses $C_1, \ldots, C_m$ such that $C = C_m$
- Each C_i is either from F or obtained from resolution of C_j and C_k, $j, k < i$
- $\mathrm{Res}^*(F)$ is set of all clauses derivable from F

Putting it all together

$$F_1 : \forall x\, A(e, x, x)$$
$$F_2 : \forall x \forall y \forall z\, (\neg A(x, y, z) \vee A(s(x), y, s(z)))$$
$$F_3 : \forall x \exists y\, A(s(s(e)), x, y)$$

show that $F_1 \wedge F_2 \models F_3$, i.e. that $F_1 \wedge F_2 \wedge \neg F_3$ is unsat

Putting it all together

$$F_1 : \forall x\, A(e, x, x)$$
$$F_2 : \forall x \forall y \forall z\, (\neg A(x, y, z) \vee A(s(x), y, s(z)))$$
$$F_3 : \forall x \exists y\, A(s(s(e)), x, y)$$

show that $F_1 \wedge F_2 \models F_3$, i.e. that $F_1 \wedge F_2 \wedge \neg F_3$ is unsat

- Step 1: Skolemise separately

$$\neg F_3 = \exists y \forall z\, \neg A(s(s(e)), y, z) \rightsquigarrow G_3 := \forall z\, \neg A(s(s(e)), c, z)$$

Putting it all together

$$F_1 : \ \forall x\, A(e, x, x)$$
$$F_2 : \ \forall x \forall y \forall z\, (\neg A(x, y, z) \vee A(s(x), y, s(z)))$$
$$F_3 : \ \forall x \exists y\, A(s(s(e)), x, y)$$

show that $F_1 \wedge F_2 \models F_3$, i.e. that $F_1 \wedge F_2 \wedge \neg F_3$ is unsat

- Step 1: Skolemise separately

$$\neg F_3 = \exists y \forall z\, \neg A(s(s(e)), y, z) \rightsquigarrow G_3 := \forall z\, \neg A(s(s(e)), c, z)$$

- Step 2: Use resolution to derive empty clause

1. $\{\neg A(s(s(e)), c, z_1)\}$	clause of G_3
2. $\{\neg A(x_2, y_2, z_2), A(s(x_2), y_2, s(z_2))\}$	clause of F_2
3. $\{\neg A(s(e), c, z_3)\}$	1,2 Res. w/ $[s(e)/x_2][c/y_2][s(z_2)/z_1][z_3/z_2]$
4. $\{\neg A(e, c, z_4)\}$	2,3 Res. w/ $[e/x_2][c/y_2][s(z_2)/z_3][z_4/z_3]$
5. $\{A(e, y_5, y_5)\}$	clause of F_1
6. $\square$	4,5 Res. Sub $[c/y_5][c/z_4]$

Soundness of resolution

Lemma (Resolution Lemma)

Let $F = \forall x_1 \dots \forall x_n G$ be a closed formula in Skolem form, with G quantifier-free. Let R be a resolvent of two clauses in G. Then $F \equiv \forall x_1 \dots \forall x_n (G \wedge R)$.

Soundness of resolution

Lemma (Resolution Lemma)

Let $F = \forall x_1 \ldots \forall x_n G$ be a closed formula in Skolem form, with G quantifier-free. Let R be a resolvent of two clauses in G. Then $F \equiv \forall x_1 \ldots \forall x_n (G \wedge R)$.

Proof.

Abbreviate $\forall x_1 \ldots \forall x_n$ to $\forall^*$. Clearly $\forall^*(G \wedge R) \models F$.

Soundness of resolution

Lemma (Resolution Lemma)

Let $F = \forall x_1 \ldots \forall x_n G$ be a closed formula in Skolem form, with G quantifier-free. Let R be a resolvent of two clauses in G. Then $F \equiv \forall x_1 \ldots \forall x_n (G \wedge R)$.

Proof.

Abbreviate $\forall x_1 \ldots \forall x_n$ to $\forall^*$. Clearly $\forall^*(G \wedge R) \models F$.
For the converse direction it suffices to show $F \models R$ (exercise).
Suppose R is resolvent of clauses $C_1, C_2 \in G$, with
$R = (C_1\theta \setminus \{L\}) \cup (C_2\theta' \setminus \{\overline{L}\})$ for substitutions θ, θ' and complementary literals $L \in C_1\theta$ and $\overline{L} \in C_2\theta'$.

Soundness of resolution

Lemma (Resolution Lemma)

Let $F = \forall x_1 \dots \forall x_n G$ be a closed formula in Skolem form, with G quantifier-free. Let R be a resolvent of two clauses in G. Then $F \equiv \forall x_1 \dots \forall x_n (G \wedge R)$.

Proof.

Abbreviate $\forall x_1 \dots \forall x_n$ to $\forall^*$. Clearly $\forall^*(G \wedge R) \models F$.
For the converse direction it suffices to show $F \models R$ (exercise).
Suppose R is resolvent of clauses $C_1, C_2 \in G$, with
$R = (C_1\theta \setminus \{L\}) \cup (C_2\theta' \setminus \{\overline{L}\})$ for substitutions θ, θ' and complementary literals $L \in C_1\theta$ and $\overline{L} \in C_2\theta'$.
Let $\mathcal{A}$ be an assignment that satisfies $F = \forall^* G$.
Since $C_1, C_2 \in G$, we have $\mathcal{A} \models C_1\theta \wedge C_2\theta'$ (exercise, apply Translation Lemma; recall that $\mathcal{A}$ assigns values to free variables in $C_1\theta \wedge C_2\theta'$).

Soundness of resolution

Lemma (Resolution Lemma)

Let $F = \forall x_1 \ldots \forall x_n G$ be a closed formula in Skolem form, with G quantifier-free. Let R be a resolvent of two clauses in G. Then $F \equiv \forall x_1 \ldots \forall x_n (G \wedge R)$.

Proof.

Abbreviate $\forall x_1 \ldots \forall x_n$ to $\forall^*$. Clearly $\forall^*(G \wedge R) \models F$.
For the converse direction it suffices to show $F \models R$ (exercise).
Suppose R is resolvent of clauses $C_1, C_2 \in G$, with $R = (C_1\theta \setminus \{L\}) \cup (C_2\theta' \setminus \{\overline{L}\})$ for substitutions θ, θ' and complementary literals $L \in C_1\theta$ and $\overline{L} \in C_2\theta'$.
Let $\mathcal{A}$ be an assignment that satisfies $F = \forall^* G$.
Since $C_1, C_2 \in G$, we have $\mathcal{A} \models C_1\theta \wedge C_2\theta'$ (exercise, apply Translation Lemma; recall that $\mathcal{A}$ assigns values to free variables in $C_1\theta \wedge C_2\theta'$).
Since $\mathcal{A}$ satisfies at most one of L and $\overline{L}$, it follows that $\mathcal{A}$ satisfies at least one of $C_1\theta \setminus \{L\}$ and $C_2\theta' \setminus \{\overline{L}\}$.

Soundness of resolution

Lemma (Resolution Lemma)

Let $F = \forall x_1 \ldots \forall x_n G$ be a closed formula in Skolem form, with G quantifier-free. Let R be a resolvent of two clauses in G. Then $F \equiv \forall x_1 \ldots \forall x_n (G \wedge R)$.

Proof.

Abbreviate $\forall x_1 \ldots \forall x_n$ to $\forall^*$. Clearly $\forall^*(G \wedge R) \models F$.
For the converse direction it suffices to show $F \models R$ (exercise).
Suppose R is resolvent of clauses $C_1, C_2 \in G$, with $R = (C_1\theta \setminus \{L\}) \cup (C_2\theta' \setminus \{\overline{L}\})$ for substitutions θ, θ' and complementary literals $L \in C_1\theta$ and $\overline{L} \in C_2\theta'$.
Let $\mathcal{A}$ be an assignment that satisfies $F = \forall^* G$.
Since $C_1, C_2 \in G$, we have $\mathcal{A} \models C_1\theta \wedge C_2\theta'$ (exercise, apply Translation Lemma; recall that $\mathcal{A}$ assigns values to free variables in $C_1\theta \wedge C_2\theta'$).
Since $\mathcal{A}$ satisfies at most one of L and $\overline{L}$, it follows that $\mathcal{A}$ satisfies at least one of $C_1\theta \setminus \{L\}$ and $C_2\theta' \setminus \{\overline{L}\}$.
Conclude that $\mathcal{A}$ satisfies R, as required. □

Completeness of resolution

Lemma (Liting-lemma)

Let C_1, C_2 be predicate clauses and let C_1', C_2' be two ground instances of them that can be resolved into the resolvent R'.

Then there is predicate resolvent R of C_1, C_2 such that R' is a ground instance of R.

Completeness of resolution

Lemma (Liting-lemma)

Let C_1, C_2 be predicate clauses and let C_1', C_2' be two ground instances of them that can be resolved into the resolvent R'.

Then there is predicate resolvent R of C_1, C_2 such that R' is a ground instance of R.

C_1 C_2

—: Resolution
$\rightarrow$: Substitution

Completeness of resolution

Lemma (Liting-lemma)

Let C_1, C_2 be predicate clauses and let C_1', C_2' be two ground instances of them that can be resolved into the resolvent R'.

Then there is predicate resolvent R of C_1, C_2 such that R' is a ground instance of R.

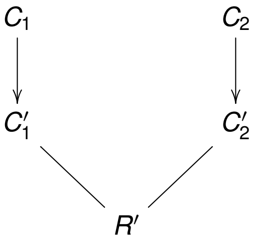
—: Resolution
$\rightarrow$: Substitution

Completeness of resolution

Lemma (Liting-lemma)

Let C_1, C_2 be predicate clauses and let C_1', C_2' be two ground instances of them that can be resolved into the resolvent R'.

Then there is predicate resolvent R of C_1, C_2 such that R' is a ground instance of R.



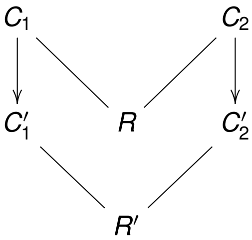
—: Resolution
→: Substitution

Completeness of resolution

Lemma (Liting-lemma)

Let C_1, C_2 be predicate clauses and let C_1', C_2' be two ground instances of them that can be resolved into the resolvent R'.

Then there is predicate resolvent R of C_1, C_2 such that R' is a ground instance of R.



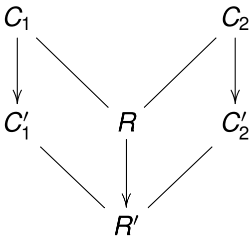
—: Resolution
→: Substitution

Completeness of resolution

Lemma (Liting-lemma)

Let C_1, C_2 be predicate clauses and let C'_1, C'_2 be two ground instances of them that can be resolved into the resolvent R'.

Then there is predicate resolvent R of C_1, C_2 such that R' is a ground instance of R.



—: Resolution
→: Substitution

Lifting-Lemma: example

$\{\neg P(f(x)), Q(x)\}$ $\qquad\qquad$ $\{P(f(g(y)))\}$

Lifting-Lemma: example

$$\{\neg P(f(x)), Q(x)\} \xrightarrow{[x/g(a)]} \{\neg P(f(g(a))), Q(g(a))\}$$

$$\{P(f(g(y)))\} \xrightarrow{[y/a]} \{P(f(g(a)))\}$$

Lifting-Lemma: example

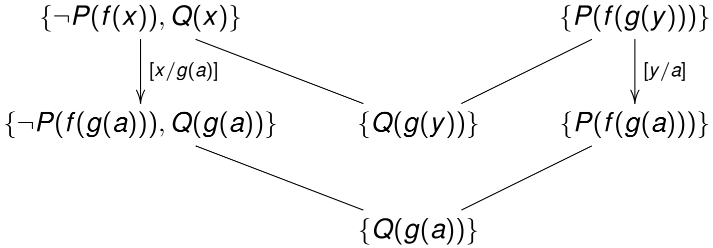
$$\{\neg P(f(x)), Q(x)\} \xrightarrow{[x/g(a)]} \{\neg P(f(g(a))), Q(g(a))\}$$

$$\{P(f(g(y)))\} \xrightarrow{[y/a]} \{P(f(g(a)))\}$$

$$\{\neg P(f(g(a))), Q(g(a))\} \quad \{P(f(g(a)))\}$$

$$\{Q(g(a))\}$$

Lifting-Lemma: example



Lifting-Lemma: example