Lecture 12
Resolution for predicate logic
Unification, resolution
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 $\sigma$-terms

- More general: substitution is function $\theta$ mapping $\sigma$-terms to $\sigma$-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.
  \[
P(x, c) \theta = P(x \theta, c \theta),\text{ etc.}
\]

- Denote by $\theta \cdot \theta'$ substitution first performing $\theta$ and then $\theta'$
Substitution

Key concept substitution:

- Used to replace variables by $\sigma$-terms
- More general: substitution is function $\theta$ mapping $\sigma$-terms to $\sigma$-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.
  \[
P(x, c)\theta = P(x\theta, c\theta), \text{ etc.}
\]
- Denote by $\theta \cdot \theta'$ substitution first performing $\theta$ and then $\theta'$

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 $\sigma$-terms
- More general: substitution is function $\theta$ mapping $\sigma$-terms to $\sigma$-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 $\theta$ and then $\theta'$

**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 \( D\theta := \{L_1\theta, \ldots, L_k\theta\} \)

\( \theta \) unifies \( D \) if \( D\theta = \{L\} \) for some literal \( L \)
For sets of literals $D = \{L_1, \ldots, L_k\}$, define $D\theta := \{L_1\theta, \ldots, L_k\theta\}$.

\[ \theta \text{ unifies } D \text{ if } D\theta = \{L\} \text{ for some literal } L \]

**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 $D = \{L_1, \ldots, L_k\}$, define $D\theta := \{L_1\theta, \ldots, L_k\theta\}$

- $\theta$ 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 $D\theta := \{L_1\theta, \ldots, L_k\theta\}$
- $\theta$ 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 $\theta$ a **most general unifier (mgu)** of $D$ if $\theta$ is a unifier and for all other unifiers $\theta'$ there is unifier $\theta''$ such that $\theta' = \theta \cdot \theta''$. 
Substitution

- For sets of literals $D = \{L_1, \ldots, L_k\}$, define $D\theta := \{L_1\theta, \ldots, L_k\theta\}$

- $\theta$ 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 $\theta$ a **most general unifier (mgu)** of $D$ if $\theta$ is a unifier and for all other unifiers $\theta'$ there is unifier $\theta''$ 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 $\theta$ 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

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

\[
\begin{align*}
\{P(x, y), P(f(z), x)\}, \text{ apply } [f(z)/x] \\
\{P(f(z), y), P(f(z), f(z))\}, \text{ apply } [f(z)/y] \\
\{P(f(z), f(z))\}
\end{align*}
\]

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

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

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

Thus \([f(z)/x][f(z)/y]\) is a most general unifier of the set \( D \).
<table>
<thead>
<tr>
<th>Unifiable?</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P(f(x))$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$P(g(y))$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$P(x)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$P(f(y))$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$P(x, f(y))$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$P(f(u), z)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$P(x, f(y))$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$P(f(u), f(z))$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$P(x, f(x))$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$P(f(y), y)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$P(x, g(x), g^2(x))$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$P(f(z), w, g(w))$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$P(x, f(y))$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$P(g(y), f(a))$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$P(g(a), z)$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
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 \( \theta' \) 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 $\theta'$ of $D\theta$, we have $\theta' = \theta \cdot \theta'$.

- Holds before entering while loop because $\theta$ 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 \( \theta' \) of \( D\theta \), we have \( \theta' = \theta \cdot \theta' \).

- Holds before entering while loop because \( \theta \) is the identity substitution.
- Let \( \theta' \) 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 \( \theta' \) of \( D\theta \), we have \( \theta' = \theta \cdot \theta' \).

- Holds before entering while loop because \( \theta \) is the identity substitution.
- Let \( \theta' \) 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 \( \theta' \) 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 $\theta'$ of $D\theta$, we have $\theta' = \theta \cdot \theta'$.

- Holds before entering while loop because $\theta$ is the identity substitution.
- Let $\theta'$ 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 $\theta'$ 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 \( \theta' \) of \( D\theta \), we have \( \theta' = \theta \cdot \theta' \).

- Holds before entering while loop because \( \theta \) is the identity substitution.
- Let \( \theta' \) be unifier. Assume \( \theta' = \theta \cdot \theta' \) holds at \textbf{begin} and algorithm does not halt. We show \( \theta' = \theta \cdot \theta' \) holds again at \textbf{end}.
- Since algorithm does not halt, we find \( x \) and \( t \) in \( D\theta \).
- Since \( \theta' \) 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: \( \theta \) is unifier because of the loop condition, and loop invariant implies \( \theta \) is mgu.
Resolution

For set of literals $D$, $\overline{D}$ denotes complement of all literals in $D$. 
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 $\theta$ 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 $\theta$ 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 $\theta_1, \theta_2$ 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

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

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

as in the figure above.
Example

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

\[
C_1 = \{ \neg P(f(e), x, f(g(e))) \} \quad \text{and} \quad C_2 = \{ \neg P(x, y, z), P(f(x), y, f(z)) \}
\]
as in the figure above.
Exercise

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

<table>
<thead>
<tr>
<th>$C_1$</th>
<th>$C_2$</th>
<th>Resolvents</th>
</tr>
</thead>
<tbody>
<tr>
<td>{ $P(x), Q(x, y)$ }</td>
<td>{ $\neg P(f(x))$ }</td>
<td></td>
</tr>
<tr>
<td>{ $Q(g(x)), R(f(x))$ }</td>
<td>{ $\neg Q(f(x))$ }</td>
<td></td>
</tr>
<tr>
<td>{ $P(x), P(f(x))$ }</td>
<td>{ $\neg P(y), Q(y, z)$ }</td>
<td></td>
</tr>
</tbody>
</table>
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 $C_k$, $j, k < i$
- $\text{Res}^*(F)$ is set of all clauses derivable from $F$
Putting it all together

\[ F_1 : \forall x A(e, x, x) \]
\[ F_2 : \forall x \forall y \forall z (\neg A(x, y, z) \lor A(s(x), y, s(z))) \]
\[ F_3 : \forall x \exists y A(s(s(e)), x, y) \]

show that \( F_1 \land F_2 \models F_3 \), i.e. that \( F_1 \land F_2 \land \neg F_3 \) is unsat
Putting it all together

\[ F_1 : \forall x \ A(e, x, x) \]
\[ F_2 : \forall x \forall y \forall z (\neg A(x, y, z) \lor A(s(x), y, s(z))) \]
\[ F_3 : \forall x \exists y \ A(s(s(e)), x, y) \]

show that \( F_1 \land F_2 \models F_3 \), i.e. that \( F_1 \land F_2 \land \neg F_3 \) is unsat

- **Step 1:** Skolemise separately
  \[ \neg F_3 = \exists y \forall z \neg A(s(s(e)), y, z) \sim G_3 := \forall z \neg A(s(s(e)), c, z) \]
Putting it all together

\[ F_1 : \forall x \ A(e, x, x) \]
\[ F_2 : \forall x \forall y \forall z \ (\neg A(x, y, z) \lor A(s(x), y, s(z))) \]
\[ F_3 : \forall x \exists y \ A(s(s(e)), x, y) \]

show that \( F_1 \land F_2 \models F_3 \), i.e. that \( F_1 \land F_2 \land \neg F_3 \) is unsat

- Step 1: Skolemise separately
  \[ \neg F_3 = \exists y \forall z \neg A(s(s(e)), y, z) \leadsto G_3 := \forall z \neg A(s(s(e)), c, z) \]

- Step 2: Use resolution to derive empty clause

1. \{\neg A(s(s(e)), c, z_1)\} \hspace{1cm} \text{clause of } G_3
2. \{\neg A(x_2, y_2, z_2), A(s(x_2), y_2, s(z_2))\} \hspace{1cm} \text{clause of } F_2
3. \{\neg A(s(e), c, z_3)\}
4. \{\neg A(e, c, z_4)\}
5. \{A(e, y_5, y_5)\}
6. \lozenge

1,2 Res. w/ \([s(e)/x_2][c/y_2][s(z_2)/z_1][z_3/z_2]\)
2,3 Res. w/ \([e/x_2][c/y_2][s(z_2)/z_3][z_4/z_3] \]
4,5 Res. Sub \([c/y_5][c/z_4] \]
**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 $F \equiv \forall x_1 \ldots \forall x_n (G \land R)$. 

Proof.

Abbreviate $\forall x_1 \ldots \forall x_n$ to $\forall \ast$. Clearly $\forall \ast (G \land R) \models F$.

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 $R = (C_1 \theta \{L\}) \cup (C_2 \theta' \{L\})$ for substitutions $\theta, \theta'$ and complementary literals $L \in C_1 \theta$ and $L \in C_2 \theta'$. Let $A$ be an assignment that satisfies $F = \forall \ast G$.

Since $C_1, C_2 \in G$, we have $A| = C_1 \theta \land C_2 \theta'$ (exercise, apply Translation Lemma; recall that $A$ assigns values to free variables in $C_1 \theta \land C_2 \theta'$).

Since $A$ satisfies at most one of $L$ and $L$, it follows that $A$ satisfies at least one of $C_1 \theta\{L\}$ and $C_2 \theta'\{L\}$.

Conclude that $A$ satisfies $R$, as required.
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 $F \equiv \forall x_1 \ldots \forall x_n (G \land R)$.

**Proof.**

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

### Proof.

Abbreviate $\forall x_1 \ldots \forall x_n$ to $\forall^*$. Clearly $\forall^* (G \land R) \models F$. 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 $\theta, \theta'$ and complementary literals $L \in C_1 \theta$ and $\overline{L} \in C_2 \theta'$. 
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 $F \equiv \forall x_1 \ldots \forall x_n (G \land R)$.

Proof.

Abbreviate $\forall x_1 \ldots \forall x_n$ to $\forall^\ast$. Clearly $\forall^\ast (G \land R) \models F$. 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 $\theta, \theta'$ and complementary literals $L \in C_1 \theta$ and $\overline{L} \in C_2 \theta'$.

Let $\mathcal{A}$ be an assignment that satisfies $F = \forall^\ast G$.

Since $C_1, C_2 \in G$, we have $\mathcal{A} \models C_1 \theta \land C_2 \theta'$ (exercise, apply Translation Lemma; recall that $\mathcal{A}$ assigns values to free variables in $C_1 \theta \land C_2 \theta'$).
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 $F \equiv \forall x_1 \ldots \forall x_n (G \land R)$.

**Proof.**

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

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 $\theta, \theta'$ and complementary literals $L \in C_1 \theta$ and $\overline{L} \in C_2 \theta'$.

Let $A$ be an assignment that satisfies $F = \forall^* G$.

Since $C_1, C_2 \in G$, we have $A \models C_1 \theta \land C_2 \theta'$ (exercise, apply Translation Lemma; recall that $A$ assigns values to free variables in $C_1 \theta \land C_2 \theta'$).

Since $A$ satisfies at most one of $L$ and $\overline{L}$, it follows that $A$ satisfies at least one of $C_1 \theta \setminus \{L\}$ and $C_2 \theta' \setminus \{\overline{L}\}$. 

14 / 16
# 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 $F \equiv \forall x_1 \ldots \forall x_n (G \land R)$.

## Proof.

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

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 $\theta, \theta'$ 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 $\mathcal{A} \models C_1 \theta \land C_2 \theta'$ (exercise, apply Translation Lemma; recall that $\mathcal{A}$ assigns values to free variables in $C_1 \theta \land C_2 \theta'$).

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
→: 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 $R$.

$C_1 \downarrow \downarrow C'_1 \quad C_2 \downarrow \downarrow C'_2$

—: 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 $R$.

$\begin{array}{c}
C_1 \\
\downarrow \\
C'_1
\end{array}$ $\hspace{1cm}$ $\begin{array}{c}
C_2 \\
\downarrow \\
C'_2
\end{array}$ $\hspace{1cm}$ $\begin{array}{c}
R' \\
\downarrow \\
C_1 \\
\downarrow \\
C_2
\end{array}$

$\longrightarrow$: 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 $R$.

\[ C_1 \downarrow \downarrow C_2 \]

--- Resolution

\[ \rightarrow \text{: Substitution} \]
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 $R$.

\[ C_1 \xrightarrow{\text{Resolution}} C'_1 \xrightarrow{\text{Substitution}} R \xrightarrow{\text{Resolution}} R' \]

—: Resolution
→: Substitution
Lifting-Lemma: example

\[ \\{ \neg P(f(x)), Q(x) \} \quad \{ P(f(g(y))) \} \]
Lifting-Lemma: example

\[
\begin{align*}
\{ \neg P(f(x)), Q(x) \} & \quad \{ P(f(g(y))) \} \\
\downarrow^{[x/g(a)]} & \quad \downarrow^{[y/a]} \\
\{ \neg P(f(g(a))), Q(g(a)) \} & \quad \{ P(f(g(a))) \}
\end{align*}
\]
Lifting-Lemma: example

\[
\{ \neg P(f(x)), Q(x) \} \quad \{ P(f(g(y))) \}
\]

\[
\downarrow [x/g(a)]
\]

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

\[
\downarrow [y/a]
\]

\[
\{ Q(g(a)) \}
\]
Lifting-Lemma: example

\[
\{ \neg P(f(x)), Q(x) \} \quad \{ P(f(g(y))) \}
\]

\[
\{ \neg P(f(g(a))), Q(g(a)) \} \quad \{ Q(g(y)) \} \quad \{ P(f(g(a))) \}
\]

\[
\{ Q(g(a)) \}
\]
Lifting-Lemma: example

\[\{\neg P(f(x)), Q(x)\}\]

\[\{\neg P(f(g(a))), Q(g(a))\}\]

\[\{Q(g(y))\}\]

\[\{Q(g(a))\}\]

\[\{P(f(g(a)))\}\]

\[\{P(f(g(y)))\}\]

\[\{P(f(g(a)))\}\]

\[\{Q(g(y))\}\]

\[\{Q(g(a))\}\]

\[\{\neg P(f(g(a))), Q(g(a))\}\]