Base for automated proof: First-order resolution

Stéphane Devismes  Pascal Lafourcade  Michel Lévy
Course given by Ioana Ciuciu (ciuciu@imag.fr)

Université Joseph Fourier, Grenoble I

29 March 2013
Overview

Introduction

Clause form

Unification

1st order resolution

Completeness

Conclusion
Overview

Introduction

Clause form

Unification

1st order resolution

Completeness

Conclusion
Idea

Through skolemization, one obtains formulae without quantifiers.

Today, we propose a first-order generalization of the resolution:

- Clause form of the Skolem form.
- Definition of the generalization of the resolution.
- Coherence and completeness of the method.
Overview

Introduction

Clause form

Unification

1st order resolution

Completeness

Conclusion
Literal, clause

Definition 5.2.18

A positive literal is an atomic formula. Ex: $P(x, y)$

A negative literal is the negation of an atomic formula. Ex: $\neg Q(a)$

Every literal is either positive or negative.

A clause is a sum of literals. Ex: $P(x, y) \lor \neg Q(a)$
Clause form of a formula

Definition 5.2.19

Let $A$ a closed formula. The clause form of $A$, $F(A)$ is a set of clauses obtained in two steps:

1. skolemizing $A$, i.e. construct its Skolem form $B$

2. replace $B$ by an equivalent set $\Gamma$ obtained by distributivity of the sum over the product
Clause form of a formula

Property 5.2.20

The closure of a clause form of a closed formula $A$ has a model if and only if formula $A$ has a model. More precisely

- $\forall(F(A))$ has as consequence $A$
- if formula $A$ has a model then $\forall(f(A))$ has a model
Skolemization
Clause form

Proof

Proof.

Let $A$ closed formula, $B$ its Skolem form and $\Gamma$ its clause form. According to the skolemization properties:

- $\forall(B)$ has as consequence $A$.
- If $A$ has a model then $\forall(B)$ has a model.

Since $\Gamma$ is obtain by distributivity, $B$ and $\Gamma$ are equivalent hence $\forall(B)$ and $\forall(\Gamma)$ are also equivalent. Therefore in the two above properties, one can replace $\forall(B)$ by $\forall(\Gamma)$.

$\square$
Clause form of a set of formulae

Definition 5.2.21

Let \( \Gamma \) a set of closed formulae. We define the clause form of \( \Gamma \) as being the union of the clause form of each formula of \( \Gamma \), taking care during the skolemization to eliminate every occurrence of an existential quantifier using a new symbol.
Corollary 5.2.22

Let $\Gamma$ a set of closed formulae and $\Delta$ the clause form of $\Gamma$. We have:

- $\forall(\Delta)$ has as consequence $\Gamma$, and
- if $\Gamma$ has a model then $\forall(\Delta)$ has a model.
A Herbrand-like theorem applied to the clause forms

Theorem 5.2.23

Let $\Gamma$ a set of closed formulae and $\Delta$ the clause form of $\Gamma$. The set $\Gamma$ is unsatisfiable if and only if there exist a finite unsatisfiable subset of instances of the clauses of $\Delta$ over the signature of $\Delta$.

Proof.

According to corollary 5.2.22, the skolemization preserves the satisfiability, hence: $\Gamma$ is unsatisfiable if and only if $\forall(\Delta)$ is unsatisfiable.

According to the corollary of the Herbrand theorem 5.1.18, $\forall(\Delta)$ is unsatisfiable if and only if there is a finite unsatisfiable subset of instances of the clauses of $\Delta$ over the signature of $\Delta$. $\square$
Example 5.2.24 (1/2)

Let $A = \exists y \forall z (P(z, y) \iff \neg \exists x (P(z, x) \land P(x, z)))$. Let us compute the clause form of $A$. 

1. Let us bring $A$ to its normal form:

   $\exists y \forall z (\neg P(z, y) \lor \forall x (\neg P(z, x) \lor \neg P(x, z))) \land \exists x (P(z, x) \land P(x, z)) \lor P(z, y)$

2. Make the result proper:

   $\exists y \forall z (\neg P(z, y) \lor \forall x (\neg P(z, x) \lor \neg P(x, z))) \land \exists u (P(z, u) \land P(u, z)) \lor P(z, y)$

3. Eliminate the existential quantifiers:

   $\forall z (\neg P(z, a) \lor \forall x (\neg P(z, x) \lor \neg P(x, z))) \land (P(z, f(z)) \land P(f(z), z)) \lor P(z, a)$

4. Remove the universal quantifiers, we obtain the Skolem form of $A$:

   $(\neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z)) \land (P(z, f(z)) \land P(f(z), z)) \lor P(z, a)$

5. Transform in product of sum of literals, we obtain the clause form of $A$, which is the following set of clauses:

   $C_1 = \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z)$
   $C_2 = P(z, f(z)) \lor P(z, a)$
   $C_3 = P(f(z), z) \lor P(z, a)$
Example 5.2.24 (1/2)

Let $A = \exists y \forall z (P(z, y) \iff \neg \exists x (P(z, x) \land P(x, z)))$. Let us compute the clause form of $A$.

1. Let us bring $A$ to its normal form:

$$\exists y \forall z ((\neg P(z, y) \lor \forall x (\neg P(z, x) \lor \neg P(x, z))) \land \exists x (P(z, x) \land P(x, z)) \lor P(z, y))$$
Example 5.2.24 (1/2)

Let $A = \exists y \forall z (P(z, y) \iff \neg \exists x (P(z, x) \land P(x, z)))$. Let us compute the clause form of $A$.

1. Let us bring $A$ to its normal form:
   $$\exists y \forall z ((\neg P(z, y) \lor \forall x (\neg P(z, x) \lor \neg P(x, z))) \land \exists x (P(z, x) \land P(x, z)) \lor P(z, y))$$

2. Make the result proper:
   $$\exists y \forall z ((\neg P(z, y) \lor \forall x (\neg P(z, x) \lor \neg P(x, z))) \land \exists u (P(z, u) \land P(u, z)) \lor P(z, y))$$
Example 5.2.24 (1/2)

Let \( A = \exists y \forall z (P(z, y) \iff \neg \exists x (P(z, x) \land P(x, z))) \). Let us compute the clause form of \( A \).

1. Let us bring \( A \) to its normal form:
   \[
   \exists y \forall z ((\neg P(z, y) \lor \forall x (\neg P(z, x) \lor \neg P(x, z))) \land \exists x (P(z, x) \land P(x, z)) \lor P(z, y))
   \]

2. Make the result proper:
   \[
   \exists y \forall z ((\neg P(z, y) \lor \forall x (\neg P(z, x) \lor \neg P(x, z))) \land \exists u (P(z, u) \land P(u, z)) \lor P(z, y))
   \]

3. Eliminate the existential quantifiers:
   \[
   \forall z ((\neg P(z, a) \lor \forall x (\neg P(z, x) \lor \neg P(x, z))) \land (P(z, f(z)) \land P(f(z), z)) \lor P(z, a))
   \]
Example 5.2.24 (1/2)

Let $A = \exists y \forall z (P(z, y) \iff \neg \exists x (P(z, x) \land P(x, z)))$. Let us compute the clause form of $A$.

1. Let us bring $A$ to its normal form:
   $$\exists y \forall z ((\neg P(z, y) \lor \forall x (\neg P(z, x) \lor \neg P(x, z))) \land \exists x (P(z, x) \land P(x, z)) \lor P(z, y))$$

2. Make the result proper:
   $$\exists y \forall z ((\neg P(z, y) \lor \forall x (\neg P(z, x) \lor \neg P(x, z))) \land \exists u (P(z, u) \land P(u, z)) \lor P(z, y))$$

3. Eliminate the existential quantifiers:
   $$\forall z ((\neg P(z, a) \lor \forall x (\neg P(z, x) \lor \neg P(x, z))) \land (P(z, f(z)) \land P(f(z), z)) \lor P(z, a))$$

4. Remove the universal quantifiers, we obtain the Skolem form of $A$:
   $$((\neg P(z, a) \lor (\neg P(z, x) \lor \neg P(x, z))) \land (P(z, f(z)) \land P(f(z), z)) \lor P(z, a))$$
Example 5.2.24 (1/2)

Let $A = \exists y \forall z (P(z, y) \iff \neg \exists x (P(z, x) \land P(x, z)))$. Let us compute the clause form of $A$.

1. Let us bring $A$ to its normal form :
   $\exists y \forall z ((\neg P(z, y) \lor \forall x (\neg P(z, x) \lor \neg P(x, z))) \land \exists x (P(z, x) \land P(x, z)) \lor P(z, y))$

2. Make the result proper :
   $\exists y \forall z ((\neg P(z, y) \lor \forall x (\neg P(z, x) \lor \neg P(x, z))) \land \exists u (P(z, u) \land P(u, z)) \lor P(z, y))$

3. Eliminate the existential quantifiers :
   $\forall z ((\neg P(z, a) \lor \forall x (\neg P(z, x) \lor \neg P(x, z))) \land (P(z, f(z)) \land P(f(z), z)) \lor P(z, a))$

4. Remove the universal quantifiers, we obtain the Skolem form of $A$ :
   $((\neg P(z, a) \lor (\neg P(z, x) \lor \neg P(x, z))) \land (P(z, f(z)) \land P(f(z), z)) \lor P(z, a))$

5. Transform in product of sum of literals, we obtain the clause form of $A$, which is the following set of clauses :
   - $C_1 = \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z)$
   - $C_2 = P(z, f(z)) \lor P(z, a)$
   - $C_3 = P(f(z), z) \lor P(z, a)$
Example 5.2.24 (2/2)

\[ C_1 = \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z) \]
\[ C_2 = P(z, f(z)) \lor P(z, a) \]
\[ C_3 = P(f(z), z) \lor P(z, a) \]

A has no model if and only if there is a finite unsatisfiable set of instances of \( C_1, C_2, C_3 \) over the signature of these clauses. We search for these instances:
Example 5.2.24 (2/2)

\[ C_1 = \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z) \]
\[ C_2 = P(z, f(z)) \lor P(z, a) \]
\[ C_3 = P(f(z), z) \lor P(z, a) \]

A has no model if and only if there is a finite unsatisfiable set of instances of \( C_1, C_2, C_3 \) over the signature of these clauses.
We search for these instances:

\[ \text{Let } C'_1 \text{ obtained with } x := a, z := a \text{ in } C_1 : C'_1 = \neg P(a, a) \]
Example 5.2.24 (2/2)

- \( C_1 = \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z) \)
- \( C_2 = P(z, f(z)) \lor P(z, a) \)
- \( C_3 = P(f(z), z) \lor P(z, a) \)

A has no model if and only if there is a finite unsatisfiable set of instances of \( C_1, C_2, C_3 \) over the signature of these clauses. We search for these instances:

- Let \( C'_1 \) obtained with \( x := a, z := a \) in \( C_1 \) : \( C'_1 = \neg P(a, a) \)
- Let \( C''_1 \) obtained with \( x := a, z := f(a) \) in \( C_1 \) :
  \( C''_1 = \neg P(f(a), a) \lor \neg P(a, f(a)) \)
Example 5.2.24 (2/2)

\[ C_1 = \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z) \]

\[ C_2 = P(z, f(z)) \lor P(z, a) \]

\[ C_3 = P(f(z), z) \lor P(z, a) \]

A has no model if and only if there is a finite unsatisfiable set of instances of \( C_1, C_2, C_3 \) over the signature of these clauses.

We search for these instances:

- Let \( C'_1 \) obtained with \( x := a, z := a \) in \( C_1 \) : \( C'_1 = \neg P(a, a) \)
- Let \( C''_1 \) obtained with \( x := a, z := f(a) \) in \( C_1 \) : \\
\( C''_1 = \neg P(f(a), a) \lor \neg P(a, f(a)) \)
- Let \( C'_2 \) obtained with \( z := a \) in \( C_2 \) : \( C'_2 = P(a, f(a)) \lor P(a, a) \)
Example 5.2.24 (2/2)

- $C_1 = \neg P(z,a) \lor \neg P(z,x) \lor \neg P(x,z)$
- $C_2 = P(z,f(z)) \lor P(z,a)$
- $C_3 = P(f(z),z) \lor P(z,a)$

A has no model if and only if there is a finite unsatisfiable set of instances of $C_1, C_2, C_3$ over the signature of these clauses.

We search for these instances:

- Let $C'_1$ obtained with $x := a, z := a$ in $C_1$ : $C'_1 = \neg P(a,a)$
- Let $C''_1$ obtained with $x := a, z := f(a)$ in $C_1$ : $C''_1 = \neg P(f(a),a) \lor \neg P(a,f(a))$
- Let $C'_2$ obtained with $z := a$ in $C_2$ : $C'_2 = P(a,f(a)) \lor P(a,a)$
- Let $C'_3$ obtained with $z := a$ in $C_3$ : $C'_3 = P(f(a),a) \lor P(a,a)$

This set of instances is unsatisfiable, hence $A$ is unsatisfiable!
Example 5.2.24 (2/2)

- $C_1 = \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z)$
- $C_2 = P(z, f(z)) \lor P(z, a)$
- $C_3 = P(f(z), z) \lor P(z, a)$

A has no model if and only if there is a finite unsatisfiable set of instances of $C_1, C_2, C_3$ over the signature of these clauses.

We search for these instances:

- Let $C'_1$ obtained with $x := a, z := a$ in $C_1 : C'_1 = \neg P(a, a)$
- Let $C''_1$ obtained with $x := a, z := f(a)$ in $C_1 : C''_1 = \neg P(f(a), a) \lor \neg P(a, f(a))$
- Let $C'_2$ obtained with $z := a$ in $C_2 : C'_2 = P(a, f(a)) \lor P(a, a)$
- Let $C'_3$ obtained with $z := a$ in $C_3 : C'_3 = P(f(a), a) \lor P(a, a)$

This set of instances is unsatisfiable, hence $A$ is unsatisfiable!
Overview

Introduction
Clause form

Unification
1st order resolution
Completeness

Conclusion
Unification: expression, solution

Definition 5.3.1

- A term or a literal is an **expression**.
- A substitution $\sigma$ (see definition 5.1.3) is a **solution** of the equation $e_1 = e_2$ between two expressions, if the two expressions $e_1\sigma$ and $e_2\sigma$ are syntactically **identical**.
- A substitution is **solution of a set of equations** if it is solution of every equation of the set.
Unification : support de substitution

**Definition 5.3.3**

The support of a substitution $\sigma$ is the set of variables $x$ so that $x\sigma \neq x$.

We only consider substitutions of finite support (finite number of variables).

**Definition 5.3.3**

A substitution $\sigma$ of finite support is denoted $< x_1 := t_1, \ldots, x_n := t_n >$ or simply $x_1 := t_1, \ldots, x_n := t_n$ where there is no risk for ambiguity. The variables $x_1, \ldots, x_n$ are distinct and the substitution verified:

- for $i$ from 1 to $n$, $x_i \sigma = t_i$
- for all variable $y$ so that $y \notin \{x_1, \ldots, x_n\}$, we have: $y \sigma = y$
Unification: example 5.3.4

The equation $P(x, f(y)) = P(g(z), z)$ has as solution:

\[ x := g(f(y)), \quad z := f(y). \]

The system of equations $x = g(z), f(y) = z$ has as solution:

\[ x := g(f(y)), \quad z := f(y). \]
Unification : example 5.3.4

The equation $P(x, f(y)) = P(g(z), z)$ has as solution:

\[ x := g(f(y)), z := f(y). \]

The system of equations $x = g(z), f(y) = z$ has as solution:

\[ x := g(f(y)), z := f(y). \]
Unification : composition of the substitution

Definition 5.3.5

- Let $\sigma$ and $\tau$ 2 substitutions, we denote $\sigma\tau$ the substitution so that for every variable $x$, $x\sigma\tau = (x\sigma)\tau$.
- The substitution $\sigma\tau$ is an instance of $\sigma$.
- Two substitutions are equivalent if each of them is an instance of the other.
Unification : example 5.3.6

Consider the substitutions

- $\sigma_1 = < x := g(z), y := z >$
- $\sigma_2 = < x := g(y), z := y >$
- $\sigma_3 = < x := g(a), y := a, z := a >$

We have the following relations between these substitutions:
Unification : example 5.3.6

Consider the substitutions

- $\sigma_1 = \langle x := g(z), y := z \rangle$
- $\sigma_2 = \langle x := g(y), z := y \rangle$
- $\sigma_3 = \langle x := g(a), y := a, z := a \rangle$

We have the following relations between these substitutions:

- $\sigma_1 = \sigma_2 \langle y := z \rangle$
- $\sigma_2 = \sigma_1 \langle z := y \rangle$
- $\sigma_3 = \sigma_1 \langle z := a \rangle$
- $\sigma_3 = \sigma_2 \langle y := a \rangle$

The substitutions $\sigma_1$ and $\sigma_2$ are equivalent.
The substitution $\sigma_3$ is an instance of $\sigma_1$ as well as of $\sigma_2$, but it is not equivalent to them.
Unification: definition of the most general solution

Definition 5.3.7 (mgu)

A solution of a system of equations is called the most general if all other solutions are instances of it. Note that two « most general » solutions are equivalent.
Unification: definition of the most general solution

Definition 5.3.7 (mgu)

A solution of a system of equations is called the most general if all other solutions are instances of it. Note that two «most general» solutions are equivalent.

Example 5.3.8

Consider the equation \( f(x, g(z)) = f(g(y), x) \).

- \( \sigma_1 = < x := g(z), y := z > \),
- \( \sigma_2 = < x := g(y), z := y > \),
- \( \sigma_3 = < x := g(a), y := a, z := a > \)

are 3 solutions.

\( \sigma_1 \) and \( \sigma_2 \) are its most general solutions.
Definition 5.3.2

Let $\sigma$ a substitution and $E$ a set of expressions. $E\sigma = \{ t\sigma \mid t \in E \}$. The substitution $\sigma$ is a unifier of $E$ if and only if the set $E\sigma$ has only one element.

Let $\{ e_i \mid 1 \leq i \leq n \}$ a finite set of expressions. The substitution $\sigma$ is a unifier of this set if and only if it is solution of the following system of equations $\{ e_i = e_{i+1} \mid 1 \leq i < n \}$. 
The most general unifier

**Definition 5.3.9**

Let $E$ a set of expressions. We recall that an expression is a term or a literal. A unifier of $E$ is called the most general, if every other unifier is an instance of it.
The most general unifier and the most general solution

Remark 5.3.10

Let $E = \{ e_i \mid 1 \leq i \leq n \}$ a set of expressions. In the definition of a unifier, it is indicated that $\sigma$ is a unifier of $E$ if and only if $\sigma$ is a solution of the system $S = \{ e_i = e_{i+1} \mid 1 \leq i < n \}$. Hence the most general unifier of $E$ is the most general solution of $S$. 
Unification : the algorithm (overview)

The algorithm separates the equations into:

▶ equations to be solved, denoted by an equality
▶ solved equations, denoted by :=

Initially, there are no solved equations.

The algorithm stops when:

▶ there are no more equations to be solved: the list of solved equations is the most general solution of the initial system of equations.
▶ or when it declared that the system to solve has no solution.
Unification: the algorithm (the rules)

- **Remove the equation.** If the 2 members of an equation are identical.
- **Decompose.** If the 2 members of an equation are distinct:
  - $\neg A = \neg B$, becomes $A = B$.
  - $f(s_1, \ldots, s_n) = f(t_1, \ldots, t_n)$, becomes $s_1 = t_1, \ldots, s_n = t_n$.
    For $n = 0$ this decomposition removes the equation.
- **Failure of the decomposition** If an equation to be solved is of the form $f(s_1, \ldots, s_n) = g(t_1, \ldots, t_p)$ with $f \neq g$ then the algorithm declares that there is no solution.
  In particular, there is obviously a failure if we try to solve an equation between a positive literal and a negative literal.
Unification: the algorithm (the rules)

▶ Guidance. If an equation is of the form $t = x$ where $t$ is a term which is not a variable and $x$ is a variable, then we replace the equation by $x = t$.

▶ Elimination of a variable. If an equation to solve is of the form $x = t$ where $x$ is a variable and $t$ a term not containing $x$

1. remove the equations to solve
2. replace $x$ by $t$ in every (non solved and solved) equation
3. add $x := t$ to the solved part

▶ Failure of the elimination. If an equation to solve is of the form $x = t$ where $x$ is a variable and $t$ a term different from $x$ and containing $x$ then the algorithm declares that there is no solution.
Unification : the algorithm (example 5.3.11)

1. Solve $f(x, g(z)) = f(g(y), x)$.
   By decomposition, one obtains:
   $x = g(y)$, $g(z) = x$.
   By elimination of $x$, one obtains:
   $x := g(y)$, $g(z) = g(y)$.
   By decomposition, one obtains:
   $x := g(y)$, $z = y$.
   By elimination of $z$, one obtains the solution:
   $x := g(y)$, $z := y$.
   By removal of the identity, one obtains:
   $x := g(a)$, $y := a$, $a := a$.

2. Solve $f(x, x, a) = f(g(y), g(a), y)$.
   By decomposition, one obtains:
   $x = g(y)$, $x = g(a)$, $a = y$.
   By elimination of $x$, thanks to the first equation, one obtains:
   $x := g(y)$, $g(y) = g(a)$, $a = y$.
   By decomposition, one obtains:
   $x := g(y)$, $y := a$, $a := y$.
   By elimination of $y$, one obtains:
   $x := g(a)$, $y := a$, $a := a$. 

S. Devismes et al (Grenoble I)
Unification : the algorithm (example 5.3.11)

1. Solve $f(x, g(z)) = f(g(y), x)$.

By decomposition, one obtains:

\[ x = g(y), \quad g(z) = x \]

By elimination of $x$, one obtains:

\[ x := g(y), \quad g(z) = g(y) \]

By decomposition, one obtains:

\[ x := g(y), \quad z = y \]

By elimination of $z$, one obtains the solution:

\[ x := g(y), \quad z := y \]

By removal of the identity, one obtains:

\[ x := g(a), \quad y := a \]

\[ a = a \]
Unification: the algorithm (example 5.3.11)

1. Solve \( f(x, g(z)) = f(g(y), x) \).
Unification : the algorithm (example 5.3.11)

1. Solve $f(x, g(z)) = f(g(y), x)$.

By decomposition, one obtains: $x = g(y), g(z) = x$
Unification : the algorithm (example 5.3.11)

1. Solve \( f(x, g(z)) = f(g(y), x) \).

   By decomposition, one obtains: \( x = g(y), g(z) = x \)

   By elimination of \( x \), one obtains: \( x := g(y), g(z) = g(y) \)
Unification : the algorithm (example 5.3.11)

1. Solve \( f(x, g(z)) = f(g(y), x) \).

   By decomposition, one obtains : \( x = g(y), g(z) = x \)
   By elimination of \( x \), one obtains : \( x := g(y), g(z) = g(y) \)
   By decomposition, one obtains : \( x := g(y), z = y \)
Unification: the algorithm (example 5.3.11)

1. Solve $f(x,g(z)) = f(g(y),x)$.

   By decomposition, one obtains: $x = g(y), g(z) = x$
   By elimination of $x$, one obtains: $x := g(y), g(z) = g(y)$
   By decomposition, one obtains: $x := g(y), z = y$
   By elimination of $z$, one obtains the solution: $x := g(y), z := y$
Unification : the algorithm (example 5.3.11)

1. Solve $f(x, g(z)) = f(g(y), x)$.

   By decomposition, one obtains: $x = g(y), g(z) = x$
   By elimination of $x$, one obtains: $x := g(y), g(z) = g(y)$
   By decomposition, one obtains: $x := g(y), z = y$
   By elimination of $z$, one obtains the solution: $x := g(y), z := y$

2. Solve $f(x, x, a) = f(g(y), g(a), y)$.
Unification : the algorithm (example 5.3.11)

1. Solve \( f(x, g(z)) = f(g(y), x) \).

   By decomposition, one obtains: \( x = g(y), g(z) = x \)
   By elimination of \( x \), one obtains: \( x := g(y), g(z) = g(y) \)
   By decomposition, one obtains: \( x := g(y), z = y \)
   By elimination of \( z \), one obtains the solution: \( x := g(y), z := y \)

2. Solve \( f(x, x, a) = f(g(y), g(a), y) \).

   By decomposition, one obtains: \( x = g(y), x = g(a), a = y \)
Unification : the algorithm (example 5.3.11)

1. Solve \( f(x, g(z)) = f(g(y), x) \).
   
   By decomposition, one obtains: \( x = g(y), g(z) = x \)
   By elimination of \( x \), one obtains: \( x := g(y), g(z) = g(y) \)
   By decomposition, one obtains: \( x := g(y), z = y \)
   By elimination of \( z \), one obtains the solution: \( x := g(y), z := y \)

2. Solve \( f(x, x, a) = f(g(y), g(a), y) \).
   
   By decomposition, one obtains: \( x = g(y), x = g(a), a = y \)
   By elimination of \( x \), thanks to the first equation, one obtains: \( x := g(y), g(y) = g(a), a = y \)
Unification : the algorithm (example 5.3.11)

1. Solve \( f(x, g(z)) = f(g(y), x) \).

   By decomposition, one obtains : \( x = g(y), g(z) = x \)
   By elimination of \( x \), one obtains : \( x := g(y), g(z) = g(y) \)
   By decomposition, one obtains : \( x := g(y), z = y \)
   By elimination of \( z \), one obtains the solution : \( x := g(y), z := y \)

2. Solve \( f(x, x, a) = f(g(y), g(a), y) \).

   By decomposition, one obtains : \( x = g(y), x = g(a), a = y \)
   By elimination of \( x \), thanks to the first equation, one obtains :
   \( x := g(y), g(y) = g(a), a = y \)
   By decomposition, one obtains : \( x := g(y), y = a, a = y \)
Unification : the algorithm (example 5.3.11)

1. Solve $f(x, g(z)) = f(g(y), x)$.

By decomposition, one obtains: $x = g(y), g(z) = x$
By elimination of $x$, one obtains: $x := g(y), g(z) = g(y)$
By decomposition, one obtains: $x := g(y), z = y$
By elimination of $z$, one obtains the solution: $x := g(y), z := y$

2. Solve $f(x, x, a) = f(g(y), g(a), y)$.

By decomposition, one obtains: $x = g(y), x = g(a), a = y$
By elimination of $x$, thanks to the first equation, one obtains:
$x := g(y), g(y) = g(a), a = y$
By decomposition, one obtains: $x := g(y), y = a, a = y$
By elimination of $y$, one obtains: $x := g(a), y := a, a = a$
Unification : the algorithm (example 5.3.11)

1. Solve $f(x, g(z)) = f(g(y), x)$.

   By decomposition, one obtains: $x = g(y), g(z) = x$
   By elimination of $x$, one obtains: $x := g(y), g(z) = g(y)$
   By decomposition, one obtains: $x := g(y), z = y$
   By elimination of $z$, one obtains the solution: $x := g(y), z := y$

2. Solve $f(x, x, a) = f(g(y), g(a), y)$.

   By decomposition, one obtains: $x = g(y), x = g(a), a = y$
   By elimination of $x$, thanks to the first equation, one obtains:
   $x := g(y), g(y) = g(a), a = y$
   By decomposition, one obtains: $x := g(y), y = a, a = y$
   By elimination of $y$, one obtains: $x := g(a), y := a, a = a$
   By removal of the identity, one obtains: $x := g(a), y := a$
Unification : the algorithm (example 5.3.11)

1. Solve $f(x, x, x) = f(g(y), g(a), y)$.
Unification: the algorithm (example 5.3.11)

1. Solve $f(x, x, x) = f(g(y), g(a), y)$.

By decomposition, one obtains: $x = g(y), x = g(a), x = y$

Remark: the correction and termination proofs of the unification algorithm are available in the course support (poly).
Unification : the algorithm (example 5.3.11)

1. Solve $f(x, x, x) = f(g(y), g(a), y)$.

   By decomposition, one obtains: $x = g(y), x = g(a), x = y$

   By elimination of $x$, one obtains: $x := g(y), g(y) = g(a), g(y) = y$

Remark: the correction and termination proofs of the unification algorithm are available in the course support (poly).
Unification : the algorithm (example 5.3.11)

1. Solve $f(x, x, x) = f(g(y), g(a), y)$.

   By decomposition, one obtains: $x = g(y), x = g(a), x = y$
   By elimination of $x$, one obtains: $x := g(y), g(y) = g(a), g(y) = y$
   By guiding the equations, one obtains:
   $x := g(y), g(y) = g(a), y = g(y)$

Remark: the correction and termination proofs of the unification algorithm are available in the course support (poly).
Unification: the algorithm (example 5.3.11)

1. Solve \( f(x, x, x) = f(g(y), g(a), y) \).

   By decomposition, one obtains: \( x = g(y), x = g(a), x = y \).

   By elimination of \( x \), one obtains: \( x := g(y), g(y) = g(a), g(y) = y \).

   By guiding the equations, one obtains:
   \[ x := g(y), g(y) = g(a), y = g(y) \]

   The equation \( y = g(y) \) generates a failure. Hence the equation
   \( f(x, x, x) = f(g(y), g(a), y) \) has no solution.
Unification : the algorithm (example 5.3.11)

1. Solve $f(x, x, x) = f(g(y), g(a), y)$.

By decomposition, one obtains : $x = g(y), x = g(a), x = y$

By elimination of $x$, one obtains : $x := g(y), g(y) = g(a), g(y) = y$

By guiding the equations, one obtains :

$x := g(y), g(y) = g(a), y = g(y)$

The equation $y = g(y)$ generates a failure. Hence the equation $f(x, x, x) = f(g(y), g(a), y)$ has no solution.

**Remark** : the correction and termination proofs of the unification algorithm are available in the course support (poly).
Overview

Introduction

Clause form

Unification

1st order resolution

Completeness

Conclusion
Idea

Let $\Gamma$ a set of clauses. Suppose that $\forall(\Gamma)$ has no model. What to do?
Idea

Let \( \Gamma \) a set of clauses. Suppose that \( \forall(\Gamma) \) has no model. What to do?

The formal system «factoring, copy, binary resolution» is a formal system allowing to deduce \( \bot \) from \( \Gamma \).
Idea

Let Γ a set of clauses. Suppose that $\forall(\Gamma)$ has no model. What to do?

The formal system «factoring, copy, binary resolution» is a formal system allowing to deduce $\bot$ from Γ.

The completeness of this formal system is based on the Herbrand theorem. In order to find the contradictory instances of the clause, the rules use the unification algorithm.
Three rules

1. **Factoring** which, starting from the premise

\[ P(x, f(y)) \lor P(g(z), z) \lor Q(z, x) \]

deduces

\[ P(g(f(y)), f(y)) \lor Q(f(y), g(f(y))) \].

The deduced clause is obtained by calculating the most general solution

\[ x := g(f(y)), z := f(y) \] of \( P(x, f(y)) = P(g(z), z) \).

2. **Copy** which allows to rename the variable of a clause.

3. **Binary resolution** (BR) which, starting from the two premises with no common variable \( P(x, a) \lor Q(x) \) and \( \neg P(b, y) \lor R(f(y)) \)

deduces the resolvent \( Q(b) \lor R(f(a)) \), by calculating the most general solution \( x := b, y := a \) of \( P(x, a) = P(b, y) \).
Resolution : 3 Resolution rules

1. factoring,
2. copy,
3. resolvent
Resolution: 3 Resolution rules

1. factoring,
2. copy,
3. resolvent

A clause, which is a sum of literals, is identified by the set of its literals.
Factoring

**Definition 5.4.2**

The clause $C'$ is a factor of the clause $C$ if $C' = C$ or if there exist a subset $E$ of $C$ so that $E$ has at least two elements, $E$ is unifiable and $C' = C\sigma$ where $\sigma$ is the most general unifier of $E$. 

Example 5.4.3

The clause $P(x) \lor Q(g(x, y)) \lor P(f(a))$ has two factors: it itself and the factor $P(f(a)) \lor Q(g(f(a), y))$ obtained by applying the most general unifier $x := f(a)$ of the two underlined literals to the clause.
Factoring

Definition 5.4.2

The clause $C'$ is a factor of the clause $C$ if $C' = C$ or if there exist a subset $E$ of $C$ so that $E$ has at least two elements, $E$ is unifiable and $C' = C\sigma$ where $\sigma$ is the most general unifier of $E$.

Example 5.4.3

The clause $P(x) \lor Q(g(x,y)) \lor P(f(a))$ has two factors:
Factoring

Definition 5.4.2
The clause $C'$ is a factor of the clause $C$ if $C' = C$ or if there exist a subset $E$ of $C$ so that $E$ has at least two elements, $E$ is unifiable and $C' = C\sigma$ where $\sigma$ is the most general unifier of $E$.

Example 5.4.3
The clause $P(x) \lor Q(g(x, y)) \lor P(f(a))$ has two factors:

- itself and the factor $P(f(a)) \lor Q(g(f(a), y))$ obtained by applying the most general unifier $x := f(a)$ of the two underlined literals to the clause.
Factoring

Property 5.4.1

Let $A$ a formula with no quantifier and $B$ an instance of $A$. 
$\forall (A) \models \forall (B)$

Proof.

Cf. Poly
Factoring

Property 5.4.1

Let $A$ a formula with no quantifier and $B$ an instance of $A$.

$$\forall(A) \models \forall(B)$$

Proof.

Cf. Poly

Property 5.4.4

Let $C'$ a factor of the clause $C$.

$$\forall(C) \models \forall(C')$$

Proof.

Since $C'$ is an instance of $C$, it is a consequence of the property 5.4.1.
Definition 5.4.5

Let $C$ a clause and $\sigma$ a substitution, which only changes the variables of $C$ and whose restriction to the variables of $C$ is a bijection between these variables and those of the clause $C\sigma$.

The clause $C\sigma$ is a copy of the clause $C$.

The substitution $\sigma$ is also called a renaming of $C$. 
Definition 5.4.6

Let $C$ a clause and $\sigma$ a renaming of $C$. Let $f$ the restriction of $\sigma$ to the variables of $C$ and $f^{-1}$ the reciprocal application of $f$. Let $\sigma^{-1}_C$ the substitution thus defined for every variable $x$ :

- If $x$ is a variable of $C\sigma$ then $x\sigma^{-1}_C = xf^{-1}$
- Else $x\sigma^{-1}_C = x$.

This substitution is called the inverse of the renaming $\sigma$ of $C$. 
Copy

Example 5.4.7

Let $\sigma = < x := u, y := v >$.

$\sigma$ is a renaming of $P(x, y)$.

The literal $P(u, v)$, where $P(u, v) = P(x, y)\sigma$, is a copy of $P(x, y)$.

Let $\tau = < u := x, v := y >$. $\tau$ is the inverse of the renaming $\sigma$ of $P(x, y)$.

Note that $P(u, v)\tau = P(x, y)$: the literal $P(x, y)$ is a copy of $P(u, v)$ by the renaming $\tau$. 
Property 5.4.8

Let $C$ a clause and $\sigma$ a renaming of $C$.

1. $\sigma_C^{-1}$ is a renaming of $C\sigma$.
2. For every expression or clause $E$, whose variables are those of $C$, $E\sigma\sigma_C^{-1} = E$.

Hence $C\sigma\sigma_C^{-1} = C$ and consequently $C$ is a copy of $C\sigma$.

Proof.

Let $f$ the restriction of $\sigma$ to the variable of $C$. By definition of the renaming, $f$ is a bijection between the variables of $C$ and those of $C\sigma$.

1. By definition of $\sigma_C^{-1}$, this substitution only changes the variables of $C\sigma$ and its restriction to the variables of $C\sigma$ is the bijection $f^{-1}$. Hence, $\sigma_C^{-1}$ is a renaming of $C\sigma$.

2. Let $x$ a variable of $C$. By definition of $f$, $x\sigma\sigma_C^{-1} = xff^{-1} = x$. Hence, by recursion on the terms, literals and clauses, for every expression or clause $E$, whose variables are those of $C$, we have $E\sigma\sigma_C^{-1} = E$. 

□
Copy

Property 5.4.9

Consider two clauses which are copy of each other, their universal closures are equivalent.

Proof.

Let $C'$ a copy of $C$. By definition, $C'$ is an instance of $C$ and according to the preceding property, $C$ is a copy of $C'$, hence an instance of $C$.

Hence according to property 5.4.1, the universal closure of $C$ is a consequence of that of $C'$ and vice versa. Therefore, these two closures are equivalent.
Binary resolvent

**Definition 5.4.10**

Let $C$ and $D$ two clauses with no common variable. The clause $E$ is a binary resolvent of $C$ and $D$ if there is a literal $L \in C$ and a literal $M \in D$ so that $L$ and $M^c$ are unifiable and if $E = ((C - \{L\}) \cup (D - \{M\}))\sigma$ where $\sigma$ is the most general solution of the equation $L = M^c$. 
Binary resolvent

**Definition 5.4.10**

Let $C$ and $D$ two clauses with no common variable. The clause $E$ is a binary resolvent of $C$ and $D$ if there is a literal $L \in C$ and a literal $M \in D$ so that $L$ and $M^c$ are unifiable and if $E = ((C - \{L\}) \cup (D - \{M\})) \sigma$ where $\sigma$ is the most general solution of the equation $L = M^c$.

**Example 5.4.11**

Let $C = P(x, y) \lor P(y, k(z))$ and $D = \neg P(a, f(a, y_1))$. 
Binary resolvent

Definition 5.4.10

Let $C$ and $D$ two clauses with no common variable. The clause $E$ is a binary resolvent of $C$ and $D$ if there is a literal $L \in C$ and a literal $M \in D$ so that $L$ and $M^c$ are unifiable and if $E = ((C - \{L\}) \cup (D - \{M\}))\sigma$ where $\sigma$ is the most general solution of the equation $L = M^c$.

Example 5.4.11

Let $C = P(x, y) \lor P(y, k(z))$ and $D = \neg P(a, f(a, y_1))$.

$\langle x := a, y := f(a, y_1) \rangle$ is the most general solution of $P(x, y) = P(a, f(a, y_1))$, hence $P(f(a, y_1), k(z))$ is a binary resolvent of the clauses $C$ and $D$. 
Property 5.4.12

Let $E$ a binary resolvent of the clauses $C$ and $D : \forall(C), \forall(D) \models \forall(E)$.

Proof.

Cf. Poly
Réolution :

**Définition 5.4.13**

Soit $\Gamma$ un ensemble de clauses et $C$ une clause.

Un *preuve* de $C$ à partir de $\Gamma$ est une séquence de clauses terminant par $C$, chaque clause de la preuve est

- un élément de $\Gamma$,
- un facteur d'une clause précédente dans la preuve,
- une copie d'une clause précédente dans la preuve ou
- un résolvant binaire de deux clauses précédentes dans la preuve.

$C$ est déduit de $\Gamma$ en première ordre notée $\Gamma \Vdash_{1fcb} C$, si une preuve de $C$ à partir de $\Gamma$ existe.

Lorsqu'il n'y a pas d'ambiguïté, $\Vdash_{1fcb}$ est remplacé par $\Vdash$. 

S. Devismes *et al* (Grenoble I)
Property 5.4.14

Let $\Gamma$ a set of clauses and $C$ a clause.

If $\Gamma \vdash_{1fcb} C$ then $\forall (\Gamma) \models \forall (C)$

This property is an immediate consequence of the coherence of the factorization, of the copy and of the binary resolution. This proof is an induction requested in exercise 84.
Resolution : Example 5.4.15

Consider the following two clauses

1. \( C_1 = P(x, y) \lor P(y, x) \)
2. \( C_2 = \neg P(u, z) \lor \neg P(z, u) \)

Let us show by resolution that \( \forall (C_1, C_2) \) has no model.
Resolution : Example 5.4.15

Consider the following two clauses

1. $C_1 = P(x, y) \lor P(y, x)$
2. $C_2 = \neg P(u, z) \lor \neg P(z, u)$

Let us show by resolution that $\forall (C_1, C_2)$ has no model.

1. $P(x, y) \lor P(y, x)$ Hyp $C_1$
Resolution : Example 5.4.15

Consider the following two clauses
1. \( C_1 = P(x, y) \lor P(y, x) \)
2. \( C_2 = \neg P(u, z) \lor \neg P(z, u) \)

Let us show by resolution that \( \forall (C_1, C_2) \) has no model.

1. \( P(x, y) \lor P(y, x) \) Hyp \( C_1 \)
2. \( P(y, y) \) Factor of 1 by \( <x := y> \)
Resolution: Example 5.4.15

Consider the following two clauses

1. \( C_1 = P(x, y) \lor P(y, x) \)
2. \( C_2 = \neg P(u, z) \lor \neg P(z, u) \)

Let us show by resolution that \( \forall (C_1, C_2) \) has no model.

1. \( P(x, y) \lor P(y, x) \) Hyp \( C_1 \)
2. \( P(y, y) \) Factor of 1 by \( x := y \)
3. \( \neg P(u, z) \lor \neg P(z, u) \) Hyp \( C_2 \)
Resolution : Example 5.4.15

Consider the following two clauses

1. \( C_1 = P(x, y) \lor P(y, x) \)
2. \( C_2 = \neg P(u, z) \lor \neg P(z, u) \)

Let us show by resolution that \( \forall (C_1, C_2) \) has no model.

1. \( P(x, y) \lor P(y, x) \) Hyp \( C_1 \)
2. \( P(y, y) \) Factor of 1 by \( < x := y > \)
3. \( \neg P(u, z) \lor \neg P(z, u) \) Hyp \( C_2 \)
4. \( \neg P(z, z) \) Factor of 3 by \( < u := z > \)
Resolution : Example 5.4.15

Consider the following two clauses
1. $C_1 = P(x,y) \lor P(y,x)$
2. $C_2 = \neg P(u,z) \lor \neg P(z,u)$

Let us show by resolution that $\forall(C_1, C_2)$ has no model.

1. $P(x,y) \lor P(y,x)$ Hyp $C_1$
2. $P(y,y)$ Factor of 1 by $<x := y>$
3. $\neg P(u,z) \lor \neg P(z,u)$ Hyp $C_2$
4. $\neg P(z,z)$ Factor of 3 by $<u := z>$
5. $\bot$ BR 2, 4 by $<y := z>$
Resolution: Example 5.4.15

Consider the following two clauses

1. $C_1 = P(x, y) \lor P(y, x)$
2. $C_2 = \neg P(u, z) \lor \neg P(z, u)$

Let us show by resolution that $\forall(C_1, C_2)$ has no model.

1. $P(x, y) \lor P(y, x)$ Hyp $C_1$
2. $P(y, y)$ Factor of 1 by $< x := y >$
3. $\neg P(u, z) \lor \neg P(z, u)$ Hyp $C_2$
4. $\neg P(z, z)$ Factor of 3 by $< u := z >$
5. $\bot$ BR 2, 4 by $< y := z >$

This example shows, a contrario, that the binary resolution alone is incomplete, without factorization, one can not deduce the empty clause.
Resolution: Example 5.4.16

1. \( C_1 = \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z) \)
2. \( C_2 = P(z, f(z)) \lor P(z, a) \)
3. \( C_3 = P(f(z), z) \lor P(z, a) \)

We prove that \( \forall (C_1, C_2, C_3) \) has no model.
Resolution: Example 5.4.16

1. $C_1 = \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z)$
2. $C_2 = P(z, f(z)) \lor P(z, a)$
3. $C_3 = P(f(z), z) \lor P(z, a)$

We prove that $\forall(C_1, C_2, C_3)$ has no model.

1. $\neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z)$ Hyp $C_1$
Resolution : Example 5.4.16
1. $C_1 = \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z)$
2. $C_2 = P(z, f(z)) \lor P(z, a)$
3. $C_3 = P(f(z), x) \lor P(z, a)$

We prove that $\forall (C_1, C_2, C_3)$ has no model.

1. $\neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z)$ Hyp $C_1$
2. $P(z, f(z)) \lor P(z, a)$ Hyp $C_2$
Resolution : Example 5.4.16

1. \( C_1 = \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z) \)
2. \( C_2 = P(z, f(z)) \lor P(z, a) \)
3. \( C_3 = P(f(z), z) \lor P(z, a) \)

We prove that \( \forall (C_1, C_2, C_3) \) has no model.

1. \( \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z) \) Hyp \( C_1 \)
2. \( P(z, f(z)) \lor P(z, a) \) Hyp \( C_2 \)
3. \( P(v_0, f(v_0)) \lor P(v_0, a) \) Copy 2 by \( < z := v_0 > \)
Resolution : Example 5.4.16

1. \( C_1 = \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z) \)
2. \( C_2 = P(z, f(z)) \lor P(z, a) \)
3. \( C_3 = P(f(z), z) \lor P(z, a) \)

We prove that \( \forall (C_1, C_2, C_3) \) has no model.

1. \( \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z) \) Hyp \( C_1 \)
2. \( P(z, f(z)) \lor P(z, a) \) Hyp \( C_2 \)
3. \( P(v_0, f(v_0)) \lor P(v_0, a) \) Copy 2 by \(< z := v_0 >\)
4. \( \neg P(f(v_0), a) \lor \neg P(f(v_0), v_0) \lor P(v_0, a) \) BR 1(3), 3(1) by \(< z := f(v_0); x := v_0 >\)
Resolution : Example 5.4.16

1. \( C_1 = \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z) \)
2. \( C_2 = P(z, f(z)) \lor P(z, a) \)
3. \( C_3 = P(f(z), z) \lor P(z, a) \)

We prove that \( \forall (C_1, C_2, C_3) \) has no model.

1. \( \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z) \) Hyp \( C_1 \)
2. \( P(z, f(z)) \lor P(z, a) \) Hyp \( C_2 \)
3. \( P(v_0, f(v_0)) \lor P(v_0, a) \) Copy 2 by \(< z := v_0 >\)
4. \( \neg P(f(v_0), a) \lor \neg P(f(v_0), v_0) \lor P(v_0, a) \) BR 1(3), 3(1) by \(< z := f(v_0); x := v_0 >\)
5. \( \neg P(f(a), a) \lor P(a, a) \) Fact 4 by \(< v_0 := a >\)
Resolution: Example 5.4.16

1. $C_1 = \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z)$
2. $C_2 = P(z, f(z)) \lor P(z, a)$
3. $C_3 = P(f(z), z) \lor P(z, a)$

We prove that $\forall (C_1, C_2, C_3)$ has no model.

1. $\neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z)$ Hyp $C_1$
2. $P(z, f(z)) \lor P(z, a)$ Hyp $C_2$
3. $P(v_0, f(v_0)) \lor P(v_0, a)$ Copy 2 by $< z := v_0 >$
4. $\neg P(f(v_0), a) \lor \neg P(f(v_0), v_0) \lor P(v_0, a)$ BR 1(3), 3(1) by $< z := f(v_0); x := v_0 >$
5. $\neg P(f(a), a) \lor P(a, a)$ Fact 4 by $< v_0 := a >$
6. $\neg P(a, a)$ Fact 1 by $< x := a; z := a >$
Resolution: Example 5.4.16
1. \( C_1 = \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z) \)
2. \( C_2 = P(z, f(z)) \lor P(z, a) \)
3. \( C_3 = P(f(z), z) \lor P(z, a) \)

We prove that \( \forall (C_1, C_2, C_3) \) has no model.

1. \( \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z) \) Hyp \( C_1 \)
2. \( P(z, f(z)) \lor P(z, a) \) Hyp \( C_2 \)
3. \( P(v_0, f(v_0)) \lor P(v_0, a) \) Copy 2 by \(< z := v_0 >\)
4. \( \neg P(f(v_0), a) \lor \neg P(f(v_0), v_0) \lor P(v_0, a) \) BR 1(3), 3(1) by \(< z := f(v_0); x := v_0 >\)
5. \( \neg P(f(a), a) \lor P(a, a) \) Fact 4 by \(< v_0 := a >\)
6. \( \neg P(a, a) \) Fact 1 by \(< x := a; z := a >\)
7. \( P(f(z), z) \lor P(z, a) \) Hyp \( C_3 \)
Resolution : Example 5.4.16
1. $C_1 = \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z)$
2. $C_2 = P(z, f(z)) \lor P(z, a)$
3. $C_3 = P(f(z), z) \lor P(z, a)$

We prove that $\forall (C_1, C_2, C_3)$ has no model.

1. $\neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z)$ Hyp $C_1$
2. $P(z, f(z)) \lor P(z, a)$ Hyp $C_2$
3. $P(v_0, f(v_0)) \lor P(v_0, a)$ Copy 2 by $< z := v_0 >$
4. $\neg P(f(v_0), a) \lor \neg P(f(v_0), v_0) \lor P(v_0, a)$ BR 1(3), 3(1) by $< z := f(v_0); x := v_0 >$
5. $\neg P(f(a), a) \lor P(a, a)$ Fact 4 by $< v_0 := a >$
6. $\neg P(a, a)$ Fact 1 by $< x := a; z := a >$
7. $P(f(z), z) \lor P(z, a)$ Hyp $C_3$
8. $P(f(v_0), v_0) \lor P(v_0, a)$ Copy 7 by $< z := v_0 >$
Resolution : Example 5.4.16

1. \( C_1 = \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z) \) Hyp \( C_1 \)
2. \( C_2 = P(z, f(z)) \lor P(z, a) \) Hyp \( C_2 \)
3. \( C_3 = P(f(z), z) \lor P(z, a) \) Hyp \( C_3 \)

We prove that \( \forall (C_1, C_2, C_3) \) has no model.

1. \( \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z) \) Hyp \( C_1 \)
2. \( P(z, f(z)) \lor P(z, a) \) Hyp \( C_2 \)
3. \( P(v_0, f(v_0)) \lor P(v_0, a) \) Copy 2 by \( < z := v_0 > \)
4. \( \neg P(f(v_0), a) \lor \neg P(f(v_0), v_0) \lor P(v_0, a) \) BR 1(3), 3(1) by \( < z := f(v_0); x := v_0 > \)
5. \( \neg P(f(a), a) \lor P(a, a) \) Fact 4 by \( < v_0 := a > \)
6. \( \neg P(a, a) \) Fact 1 by \( < x := a; z := a > \)
7. \( P(f(z), z) \lor P(z, a) \) Hyp \( C_3 \)
8. \( P(f(v_0), v_0) \lor P(v_0, a) \) Copy 7 by \( < z := v_0 > \)
9. \( P(f(a), a) \) BR 6(1), 8(2) by \( < v_0 := a > \)
Resolution : Example 5.4.16

1. \( C_1 = \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z) \)
2. \( C_2 = P(z, f(z)) \lor P(z, a) \)
3. \( C_3 = P(f(z), z) \lor P(z, a) \)

We prove that \( \forall (C_1, C_2, C_3) \) has no model.

1. \( \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z) \) Hyp \( C_1 \)
2. \( P(z, f(z)) \lor P(z, a) \) Hyp \( C_2 \)
3. \( P(v_0, f(v_0)) \lor P(v_0, a) \) Copy 2 by \( < z := v_0 > \)
4. \( \neg P(f(v_0), a) \lor \neg P(f(v_0), v_0) \lor P(v_0, a) \) BR 1(3), 3(1) by \( < z := f(v_0); x := v_0 > \)
5. \( \neg P(f(a), a) \lor P(a, a) \) Fact 4 by \( < v_0 := a > \)
6. \( \neg P(a, a) \) Fact 1 by \( < x := a; z := a > \)
7. \( P(f(z), z) \lor P(z, a) \) Hyp \( C_3 \)
8. \( P(f(v_0), v_0) \lor P(v_0, a) \) Copy 7 by \( < z := v_0 > \)
9. \( P(f(a), a) \) BR 6(1), 8(2) by \( < v_0 := a > \)
10. \( P(a, a) \) BR 5(1), 9(1)
Resolution : Example 5.4.16

1. \( C_1 = \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z) \)
2. \( C_2 = P(z, f(z)) \lor P(z, a) \)
3. \( C_3 = P(f(z), z) \lor P(z, a) \)

We prove that \( \forall(C_1, C_2, C_3) \) has no model.

1. \( \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z) \) Hyp \( C_1 \)
2. \( P(z, f(z)) \lor P(z, a) \) Hyp \( C_2 \)
3. \( P(v_0, f(v_0)) \lor P(v_0, a) \) Copy 2 by \( < z := v_0 > \)
4. \( \neg P(f(v_0), a) \lor \neg P(f(v_0), v_0) \lor P(v_0, a) \) BR 1(3), 3(1) by \( < z := f(v_0); x := v_0 > \)
5. \( \neg P(f(a), a) \lor P(a, a) \) Fact 4 by \( < v_0 := a > \)
6. \( \neg P(a, a) \) Fact 1 by \( < x := a; z := a > \)
7. \( P(f(z), z) \lor P(z, a) \) Hyp \( C_3 \)
8. \( P(f(v_0), v_0) \lor P(v_0, a) \) Copy 7 by \( < z := v_0 > \)
9. \( P(f(a), a) \) BR 6(1), 8(2) by \( < v_0 := a > \)
10. \( P(a, a) \) BR 5(1), 9(1)
11. \( \bot \) BR 6(1), 10(1)
Overview

Introduction

Clause form

Unification

1st order resolution

Completeness

Conclusion
1° order resolution

We define a new rule, the 1° order resolution, which is a combination of the three rules of factoring, copy and binary resolution.

**Definition 5.4.17**

The clause $E$ is 1° order resolvent of the clauses $C$ and $D$ if $E$ is a binary resolvent of $C'$ and $D'$ where $C'$ is a factor of $C$ and $D'$ is a copy without common variables with $C'$ of a factor of $D$.

The rule which allows to deduce $E$ from $C$ and $D$ is called the 1° order resolution.
Example 5.4.18

Let $C = \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z)$ and $D = P(z, f(z)) \lor P(z, a)$. 
Example 5.4.18

Let \( C = \neg P(z, a) \lor \neg P(z, x) \lor \neg P(x, z) \) and \( D = P(z, f(z)) \lor P(z, a) \).

\( C' = \neg P(a, a) \) is a factor of \( C \).

The clause \( P(a, f(a)) \) is a binary resolvent of \( C' \) and of \( D \) (which is a factor of itself) hence it is a first-order resolvent of \( C \) and \( D \).
Three notions of proof by resolution

let $\Gamma$ a set of clauses and $C$ a clause.

Notations
Three notions of proof by resolution

let $\Gamma$ a set of clauses and $C$ a clause.

**Notations**

1. $\Gamma \vdash_p C$ : proof of $C$ starting from $\Gamma$ by propositional resolution (without substitution).
Three notions of proof by resolution

let $\Gamma$ a set of clauses and $C$ a clause.

Notations

1. $\Gamma \vdash_p C$ : proof of $C$ starting from $\Gamma$ by propositional resolution (without substitution).

2. $\Gamma \vdash_{1fcb} C$ : proof of $C$ starting from $\Gamma$ by factoring, copy and binary resolution.
Three notions of proof by resolution

let $\Gamma$ a set of clauses and $C$ a clause.

**Notations**

1. $\Gamma \vdash_p C$ : proof of $C$ starting from $\Gamma$ by propositional resolution (without substitution).
2. $\Gamma \vdash_{1fcb} C$ : proof of $C$ starting from $\Gamma$ by factoring, copy and binary resolution.
3. $\Gamma \vdash_{1r} C$ : proof of $C$ starting from $\Gamma$ obtained by 1$^\circ$ order resolution.
Three notions of proof by resolution

let $\Gamma$ a set of clauses and $C$ a clause.

Notations

1. $\Gamma \vdash_p C$ : proof of $C$ starting from $\Gamma$ by propositional resolution (without substitution).
2. $\Gamma \vdash_{1fcb} C$ : proof of $C$ starting from $\Gamma$ by factoring, copy and binary resolution.
3. $\Gamma \vdash_{1r} C$ : proof of $C$ starting from $\Gamma$ obtained by 1⁰ order resolution.

By definition we have : $\Gamma \vdash_{1r} C$ implies $\Gamma \vdash_{1fcb} C$
Theorem of raising (1/3)

Theorem 5.4.19

Let $C$ and $D$ two clauses. Let $C'$ an instance of $C$ and $D'$ an instance of $D$. Let $E'$ a propositional resolvent of $C'$ and $D'$, there exist $E$ a first-order resolvent of $C$ and $D$ which has $E'$ as instance.

Proof.

Cf. Poly.
Theorem of raising (1/3)

**Theorem 5.4.19**

Let $C$ and $D$ two clauses. Let $C'$ an instance of $C$ and $D'$ an instance of $D$. Let $E'$ a propositional resolvent of $C'$ and $D'$, there exist $E$ a first-order resolvent of $C$ and $D$ which has $E'$ as instance.

**Proof.**

Cf. Poly.

**Example 5.4.20**

Let $C = P(x) \lor P(y) \lor R(y)$ et $D = \neg Q(x) \lor P(x) \lor \neg R(x) \lor P(y)$. 
Theorem of raising (1/3)

**Theorem 5.4.19**

Let $C$ and $D$ two clauses. Let $C'$ an instance of $C$ and $D'$ an instance of $D$. Let $E'$ a propositional resolvent of $C'$ and $D'$, there exist $E$ a first-order resolvent of $C$ and $D$ which has $E'$ as instance.

**Proof.**

Cf. Poly.

**Example 5.4.20**

Let $C = P(x) \lor P(y) \lor R(y)$ et $D = \neg Q(x) \lor P(x) \lor \neg R(x) \lor P(y)$.

- The clauses $C' = P(a) \lor R(a)$ and $D' = \neg Q(a) \lor P(a) \lor \neg R(a)$ are instances of $C$ and $D$ respectively.
Theorem of raising (1/3)

Theorem 5.4.19

Let $C$ and $D$ two clauses. Let $C'$ an instance of $C$ and $D'$ an instance of $D$. Let $E'$ a propositional resolvent of $C'$ and $D'$, there exist $E$ a first-order resolvent of $C$ and $D$ which has $E'$ as instance.

Proof.

Cf. Poly.

Example 5.4.20

Let $C = P(x) \lor P(y) \lor R(y)$ et $D = \neg Q(x) \lor P(x) \lor \neg R(x) \lor P(y)$.

- The clauses $C' = P(a) \lor R(a)$ and $D' = \neg Q(a) \lor P(a) \lor \neg R(a)$ are instances of $C$ and $D$ respectively.

- The clause $E' = P(a) \lor \neg Q(a)$ is a propositional resolvent of $C'$ and $D'$. 
Theorem of raising (1/3)

Theorem 5.4.19

Let $C$ and $D$ two clauses. Let $C'$ an instance of $C$ and $D'$ an instance of $D$. Let $E'$ a propositional resolvent of $C'$ and $D'$, there exist $E$ a first-order resolvent of $C$ and $D$ which has $E'$ as instance.

Proof.

Cf. Poly.

Example 5.4.20

Let $C = P(x) \lor P(y) \lor R(y)$ et $D = \neg Q(x) \lor P(x) \lor \neg R(x) \lor P(y)$.

- The clauses $C' = P(a) \lor R(a)$ and $D' = \neg Q(a) \lor P(a) \lor \neg R(a)$ are instances of $C$ and $D$ respectively.

- The clause $E' = P(a) \lor \neg Q(a)$ is a propositional resolvent of $C'$ and $D'$.

- The clause $E = P(x) \lor \neg Q(x)$ is a 1st order resolvent of $C$ and $D$ which has $E'$ as instance.
Theorem 5.4.21

Let $\Gamma$ a set of clauses and $\Delta$ a set of instances of the clauses of $\Gamma$, and $C_1, \ldots, C_n$ a proof by propositional resolution starting from $\Delta$.

There exist a proof $D_1, \ldots, D_n$ by 1\textsuperscript{st} order resolution starting from $\Gamma$ so that for $i$ from 1 to $n$, the clause $C_i$ is an instance of $D_i$. 

Proof.

By induction on $n$.

Let $C_1, \ldots, C_n, C_{n+1}$ a proof by propositional resolution starting from $\Delta$. By induction, there is a proof $D_1, \ldots, D_n$ by 1\textsuperscript{st} order resolution starting from $\Gamma$ so that for $i$ from 1 to $n$, the clause $C_i$ is an instance of $D_i$.

1. Suppose $C_{n+1} \in \Delta$. There exist $E \in \Gamma$ whose $C_{n+1}$ is an instance hence we take $D_{n+1} = E$.

2. Suppose that $C_{n+1}$ is a propositional resolvent of $C_j$ and $C_k$ where $j, k \leq n$. According to the previous slide, there exist $E_1$ order resolvent of $D_j$ and $D_k$: we take $D_{n+1} = E$.
Theorem of raising (2/3)

**Theorem 5.4.21**

Let $\Gamma$ a set of clauses and $\Delta$ a set of instances of the clauses of $\Gamma$, and $C_1, \ldots, C_n$ a proof by propositional resolution starting from $\Delta$.

There exist a proof $D_1, \ldots, D_n$ by 1° order resolution starting from $\Gamma$ so that for $i$ from 1 to $n$, the clause $C_i$ is an instance of $D_i$.

**Proof.**

By induction on $n$. Let $C_1, \ldots, C_n, C_{n+1}$ a proof by propositional resolution starting from $\Delta$. By induction, there is a proof $D_1, \ldots, D_n$ by 1° order resolution starting from $\Gamma$ so that for $i$ from 1 to $n$, the clause $C_i$ is an instance of $D_i$.

1. Suppose $C_{n+1} \in \Delta$. There exist $E \in \Gamma$ whose $C_{n+1}$ is an instance hence we take $D_{n+1} = E$.

2. Suppose that $C_{n+1}$ is a propositional resolvent of $C_j$ and $C_k$ where $j, k \leq n$. According to the previous slide, there exist $E$ 1° order resolvent of $D_j$ and $D_k$ : we take $D_{n+1} = E$. 
Theorem of raising(3/3)

**Corollary 5.4.22**

Let $\Gamma$ a set of clauses and $\Delta$ a set of instances of the clauses of $\Gamma$.

Suppose that $\Delta \vdash p \ C$.

There exists $D$ so that $\Gamma \vdash_{1r} D$ and $C$ is an instance of $D$. 
Example 5.4.23

let the set of clauses $P(f(x)) \lor P(u), \neg P(x) \lor Q(z), \neg Q(x) \lor \neg Q(y)$.

The universal closure of this set of clauses is unsatisfiable and we prove it in three ways

1. By instantiation on the domain of Herbrand $a, f(a), f(f(a)), \ldots$:
Example 5.4.23

Let the set of clauses $P(f(x)) \lor P(u), \neg P(x) \lor Q(z), \neg Q(x) \lor \neg Q(y)$. The universal closure of this set of clauses is unsatisfiable and we prove it in three ways:

1. **By instantiation on the domain of Herbrand $a, f(a), f(f(a)), \ldots$**:
   - $P(f(x)) \lor P(u)$ is instantiated by $x := a, u := f(a)$ in $P(f(a))$
   - $\neg P(x) \lor Q(z)$ is instantiated by $x := f(a), z := a$ in $\neg P(f(a)) \lor Q(a)$
   - $\neg Q(x) \lor \neg Q(y)$ is instantiated by $x := a, y := a$ in $\neg Q(a)$

The set of these 3 instantiations is unsatisfiable, as it is shown in the proof by propositional resolution below:
Example 5.4.23

let the set of clauses \( P(f(x)) \lor P(u), \neg P(x) \lor Q(z), \neg Q(x) \lor \neg Q(y) \). The universal closure of this set of clauses is unsatisfiable and we prove it in three ways

1. **By instantiation on the domain of Herbrand** \( a, f(a), f(f(a)), \ldots \):
   - \( P(f(x)) \lor P(u) \) is instantiated by \( x := a, u := f(a) \) in \( P(f(a)) \)
   - \( \neg P(x) \lor Q(z) \) is instantiated by \( x := f(a), z := a \) in \( \neg P(f(a)) \lor Q(a) \)
   - \( \neg Q(x) \lor \neg Q(y) \) is instantiated by \( x := a, y := a \) in \( \neg Q(a) \)

The set of these 3 instantiations is unsatisfiable, as it is shown in the proof by propositional resolution below:

\[
\begin{array}{c}
P(f(a)) \\
\hline
\neg P(f(a)) \lor Q(a) \\
\hline
Q(a) \\
\hline
\neg Q(a)
\end{array}
\]
Example 5.4.23

\[ P(f(x)) \lor P(u), \neg P(x) \lor Q(z), \neg Q(x) \lor \neg Q(y). \]
Example 5.4.23

\[ P(f(x)) \lor P(u), \lnot P(x) \lor Q(z), \lnot Q(x) \lor \lnot Q(y). \]

2. This proof by propositional resolution is raised into a proof by the rule of first-order resolution:
Example 5.4.23

\[ P(f(x)) \lor P(u), \neg P(x) \lor Q(z), \neg Q(x) \lor \neg Q(y). \]

2. This proof by propositional resolution is raised into a proof by the rule of first-order resolution:

\[
\begin{array}{c}
\frac{P(f(x)) \lor P(u) \quad \neg P(x) \lor Q(z)}{Q(z) \quad \neg Q(x) \lor \neg Q(y)}
\end{array}
\]

\[ \bot \]
Example 5.4.23

\[ P(f(x)) \lor P(u), \neg P(x) \lor Q(z), \neg Q(x) \lor \neg Q(y). \]

2. This proof by propositional resolution is raised into a proof by the rule of first-order resolution:

\[
\frac{P(f(x)) \lor P(u)}{Q(z)} \quad \frac{\neg P(x) \lor Q(z)}{\neg Q(x) \lor \neg Q(y)} \quad \nabla
\]

3. Every rule of the first-order resolution is decomposed in factoring, copy and binary resolution:
Example 5.4.23

\[ P(f(x)) \lor P(u), \neg P(x) \lor Q(z), \neg Q(x) \lor \neg Q(y). \]

2. This proof by propositional resolution is raised into a proof by the rule of first-order resolution:

\[
\frac{P(f(x)) \lor P(u)}{Q(z)} \frac{\neg P(x) \lor Q(z)}{\neg Q(x) \lor \neg Q(y)} \bot
\]

3. Every rule of the first-order resolution is decomposed in factoring, copy and binary resolution:

\[
\frac{P(f(x)) \lor P(u)}{P(f(x)) \text{ fact}} \frac{\neg P(x) \lor Q(z)}{\neg P(y) \lor Q(z) \text{ copy}} \frac{\neg Q(x) \lor \neg Q(y)}{\neg Q(x) \text{ fact}} \frac{\bot}{\bot}
\]
Refutational completeness of the $1^{\circ}$ order resolution

**Theorem 5.4.24**

Let $\Gamma$ a set of clauses. The propositions : (1) $\Gamma \vdash_{1r} \bot$, (2) $\Gamma \vdash_{1fcb} \bot$, and (3) $\forall (\Gamma) \models \bot$ are equivalent.
Refutational completeness of the $1^o$ order resolution

**Theorem 5.4.24**

Let $\Gamma$ a set of clauses. The propositions: (1) $\Gamma \vdash_{1r} \bot$, (2) $\Gamma \vdash_{1fcb} \bot$, and (3) $\forall (\Gamma) \models \bot$ are equivalent.

**Démonstration.**

- (1) implies (2) since the $1^o$ order resolution is a combination of factoring, copy and binary resolution.
Refutational completeness of the 1° order resolution

Theorem 5.4.24

Let $\Gamma$ a set of clauses. The propositions : (1) $\Gamma \vdash_{1r} \bot$, (2) $\Gamma \vdash_{1fcb} \bot$, and (3) $\forall (\Gamma) \models \bot$ are equivalent.

Démonstration.

- (1) implies (2) since the 1° order resolution is a combination of factoring, copy and binary resolution.
- (2) implies (3) since the factoring, the copy and the binary resolution are coherent.
Refutational completeness of the 1° order resolution

Theorem 5.4.24

Let \( \Gamma \) a set of clauses. The propositions : (1) \( \Gamma \vdash_{1r} \bot \), (2) \( \Gamma \vdash_{1fcb} \bot \), and (3) \( \forall (\Gamma) \models \bot \) are equivalent.

Démonstration.

- (1) implies (2) since the 1° order resolution is a combination of factoring, copy and binary resolution.
- (2) implies (3) since the factoring, the copy and the binary resolution are coherent.
- Let us prove (3) implies (1). Suppose that \( \forall (\Gamma) \models \bot \), i.e. \( \forall (\Gamma) \) is unsatisfiable. According to Herbrand theorem, there is \( \Delta \) a finite set of instances without variables of the clauses of \( \Gamma \) which has no propositional model. By completeness of the propositional resolution, we have : \( \Delta \vdash_{p} \bot \). According to the corollary 5.4.22, there exists \( D \) so that \( \Gamma \vdash_{1r} D \) and \( \bot \) instance of \( D \). But in this case, we have \( D = \bot \).
Overview

Introduction

Clause form

Unification

1st order resolution

Completeness

Conclusion
Today

- Unification
- First-order resolution
- Completeness of the first-order resolution
Overview of the Semester

TODAY

▶ Propositional logic
▶ Propositional resolution
▶ Propositional natural deduction

MID-TERM EXAM

▶ First-order logic
▶ Base for the automated proof(“first-order resolution”) *
▶ first-order natural deduction

EXAM
Next course

First-order natural deduction

- Rules
- Examples
- Tactics
Conclusion

Thank you for your attention.

Questions?