

Artificial Intelligence

A course about foundations

First-Order Resolution

Marco Piastra

Propositional Resolution

a) Refutation $\Gamma \cup \{\neg\varphi\}$ and translation into *conjunctive normal form* (CNF)

$\beta_1 \wedge \beta_2 \wedge \dots \wedge \beta_n$ where each β_i is a disjunction of literals (i.e. A or $\neg A$)

b) Translation of $\Gamma \cup \{\neg\varphi\}$ in *clausal form* (CF)

$\{\beta_1, \beta_2, \dots, \beta_n\}$ where each β_i is a *clause* (i.e. a set of literals, representing a disjunction)

c) Exhaustive application of the resolution rule

1) Selection of two clauses $\{\beta_1, \beta_2, \dots, \beta_n, \alpha\}, \{\neg\alpha, \gamma_1, \gamma_2, \dots, \gamma_m\}$

2) Generation of the *resolvent*

$\{\beta_1, \beta_2, \dots, \beta_n, \alpha\}, \{\neg\alpha, \gamma_1, \gamma_2, \dots, \gamma_m\} \vdash \{\beta_1, \beta_2, \dots, \beta_n, \gamma_1, \gamma_2, \dots, \gamma_m\}$

Termination conditions:

1) The empty clause has been derived (*success*)

2) No further resolutions are possible – *fixed point (failure)*

Clausal Form in L_{FO}

1) Refutation: $\Gamma \cup \{\neg\varphi\}$

2) Translation into *prenex normal form* (PNF):

All wff are now in the form:

$Qx_1 Qx_2 \dots Qx_n \psi$ (the *matrix* ψ does not contain quantifiers)

3) Removal of all existential quantifiers - *skolemization*:

All wff are now in the form:

$\forall x_1 \forall x_2 \dots \forall x_m \chi$ (the *skolemized matrix* χ does not contain quantifiers)

Given that all wffs are universal sentences, the universal quantifiers can just be omitted

Example:

1: $\forall x (P(x) \rightarrow (\exists y Q(x,y) \wedge R(y)))$

2: $\forall x (\neg P(x) \vee (\exists y Q(x,y) \wedge R(y)))$

(removing $\rightarrow$)

2: $\forall x \exists y (\neg P(x) \vee (Q(x,y) \wedge R(y)))$

(PNF)

3: $\forall x (\neg P(x) \vee (Q(x, k(x)) \wedge R(k(x))))$

(Skolemization, with a new function $k/1$)

4: $\neg P(x) \vee (Q(x, k(x)) \wedge R(k(x)))$

(omitting universal quantifiers)

Just atoms, connectives and parentheses...

Clausal Form in L_{FO}

1) Refutation: $\Gamma \cup \{\neg\varphi\}$

2) Translation into PNF :

All wff are now in the form:

$Qx_1 Qx_2 \dots Qx_n \psi$ (the *matrix* ψ does not contain quantifiers)

3) Removal of all existential quantifiers - *skolemization*:

All wff are now in the form:

$\forall x_1 \forall x_2 \dots \forall x_m \chi$ (the *skolemized matrix* χ does not contain quantifiers)

Given that all wffs are universal sentences, the universal quantifiers can just be omitted

4) The *clausal form* can be obtained by just treating *atoms* as *propositions* and applying the rules of propositional logic

First translate in *conjunctive normal form* (CNF) and then in *clausal form* (CF)

Example:

5: $\neg P(x) \vee (Q(x, k(x)) \wedge R(k(x)))$

(from before)

6: $(\neg P(x) \vee Q(x, k(x))) \wedge (\neg P(x) \vee R(k(x)))$

(CNF, by distributing $\vee$)

7: $\{\neg P(x), Q(x, k(x))\}, \{\neg P(x), R(k(x))\}$

(*Clausal Form*)

Unificare necesse est, for resolution

■ Problem: $\Gamma \models \varphi$?

$\Gamma \equiv \{\forall x (Greek(x) \rightarrow Human(x)), \forall x (Human(x) \rightarrow Mortal(x)), Greek(socrates)\}$

$\varphi \equiv Mortal(socrates)$

Refutation, translation, clausal form:

1: $\{\forall x (Greek(x) \rightarrow Human(x)), \forall x (Human(x) \rightarrow Mortal(x)), Greek(socrates), \neg Mortal(socrates)\}$

($\Gamma \cup \{\neg\varphi\}$ is already in PNF, no skolemization is needed)

2: $\{\{Human(x), \neg Greek(x)\}, \{Mortal(x), \neg Human(x)\}, \{Greek(socrates)\}, \{\neg Mortal(socrates)\}\}$

(*Clausal Form*)

Resolution method (attempt):

3: Try resolving: $\{\neg Mortal(socrates)\}, \{Mortal(x), \neg Human(x)\}$

Technically, no resolution is applicable: no pairs of complementary literals

*Intuitively though,
the two literals $\neg Mortal(socrates)$ and $Mortal(x)$ are complementary, somehow...*

Unification

Replacing variables with terms to render two atoms identical

■ Unifier

A substitution of variables with terms $\sigma = [x_1 = t_1, x_2 = t_2 \dots x_n = t_n]$ that makes two complementary literals α and $\neg\beta$ *resolvable*

That is, it makes the two atoms *identical*: $\sigma(\alpha) = \sigma(\beta)$

- Obviously, a unifier does not necessarily exist:
for instance, $P(g(x, f(a)), a)$ and $\neg P(g(b, f(w)), k(w))$ are not unifiable

■ MGU - *most general unifier*

It is the minimal *unifier* of α and $\neg\beta$

$$\text{MGU } \mu \Leftrightarrow \forall \sigma \exists \sigma' : \sigma = \mu \cdot \sigma'$$

Any other unifier can be obtained as a composition of μ

Constructing the MGU

■ Martelli and Montanari's algorithm

Input: $[s_1 = t_1, s_2 = t_2 \dots s_n = t_n]$ (a system of *symbolic* equations)

Procedure:

Exhaustive application of the following rules to the system of symbolic equations
(each rule *transforms* the original system)

- | | | |
|---|---|---|
| (1) $f(s_1, \dots, s_n) = f(t_1, \dots, t_n)$ | <i>replace by the equations</i>
$s_1 = t_1, \dots, s_n = t_n,$ | |
| (2) $f(s_1, \dots, s_n) = g(t_1, \dots, t_m)$ where $f \neq g$ | <i>halt with failure,</i> | ← Applies even when
either m or n are 0
(i.e. with constants) |
| (3) $x = x$ | <i>delete the equation,</i> | |
| (4) $t = x$ where t is not a variable | <i>replace by the equation $x = t,$</i> | |
| (5) $x = t$ where x does not occur in t
and x occurs elsewhere | <i>apply the substitution $\{x/t\}$
to all other equations</i> | |
| (6) $x = t$ where x occurs in t and x differs from t | <i>halt with failure.</i> | |

Unless an explicit failure occurs (i.e. by rules (2) or (6)),
the procedure terminates with success when no further rule is applicable

Constructing the MGU: examples

Example: $[f(x, a) = f(g(z), y), h(u) = h(d)]$

$[x = g(z), y = a, h(u) = h(d)]$

$[x = g(z), y = a, u = d]$

Rule (1) on $f(x, a) = f(g(z), y)$

Rule (1) on $h(u) = h(d)$, MGU

Example: $[f(x, a) = f(g(z), y), h(x, z) = h(u, d)]$

$[x = g(z), y = a, h(x, z) = h(u, d)]$

$[x = g(z), y = a, h(g(z), z) = h(u, d)]$

$[x = g(z), y = a, u = g(z), z = d]$

$[x = g(d), y = a, u = g(d), z = d]$

Rule (1) on $f(x, a) = f(g(z), y)$

Rule (5) on $x = g(z)$

Rule (1) on $h(g(z), z) = h(u, d)$

Rule (5) on $z = d$, MGU

Example: $[f(x, a) = f(g(z), y), h(x, z) = h(d, u)]$

$[x = g(z), y = a, h(x, z) = h(d, u)]$

$[x = g(z), y = a, h(g(z), z) = h(d, u)]$

$[x = g(z), y = a, g(z) = d, z = u]$

Rule (1) on $f(x, a) = f(g(z), y)$

Rule (5) on $x = g(z)$

Rule (2) on $g(z) = d$ FAILURE

Standardization of variables is also necessary

- Example: $\Gamma \models \varphi$? (transitive property - in clausal form)
 $\{ \forall x \forall y \forall z (L(x,y) \wedge L(y,z)) \rightarrow L(x,z), L(a,b), L(b,c), L(c,d) \} \models L(a,d)$?

$$\Gamma \equiv \{ \{ \neg L(x,y), \neg L(y,z), L(x,z) \}, \{ L(a,b) \}, \{ L(b,c) \}, \{ L(c,d) \} \}$$
$$\varphi \equiv \{ L(a,d) \}$$

Refutation and resolution:

- 1: $\{ \{ \neg L(x,y), \neg L(y,z), L(x,z) \}, \{ L(a,b) \}, \{ L(b,c) \}, \{ L(c,d) \}, \{ \neg L(a,d) \} \}$
- 2: Unify and resolve $\{ \neg L(x,y), \neg L(y,z), \textcolor{red}{L(x,z)} \}$ and $\{ \textcolor{red}{\neg L(a,d)} \}$:
 $[x=a, z=d]$ with resolvent $\{ \neg L(a,y), \neg L(y,d) \}$
- 3: Unify and resolve $\{ \neg L(x,y), \neg L(y,z), \textcolor{red}{L(x,z)} \}$ and $\{ \textcolor{red}{\neg L(a,y)}, \neg L(y,d) \}$:
 $[x=a, z=y]$ with resolvent $\{ \neg L(a,y), \textcolor{blue}{\neg L(y,y)}, \neg L(y,d) \}$
- 4: *This seems to lead nowhere: $\textcolor{blue}{\neg L(y,y)}$ will never be resolved in $\Gamma \cup \{ \neg \varphi \}$*

Why is this??

Standardization of variables is also necessary

- Example: $\Gamma \models \varphi$? (transitive property - in clausal form)

$\Gamma \equiv \{\{\neg L(x,y), \neg L(y,z), L(x,z)\}, \{L(a,b)\}, \{L(b,c)\}, \{L(c,d)\}\}$

$\varphi \equiv \{L(a,d)\}$

Refutation and resolution, standardize variables before each resolution

(i.e. rename all variables with new, unique names)

- 1: $\{\{\neg L(x,y), \neg L(y,z), L(x,z)\}, \{L(a,b)\}, \{L(b,c)\}, \{L(c,d)\}, \{\neg L(a,d)\}\}$
- 2: Unify and resolve $\{\neg L(x_1,y_1), \neg L(y_1,z_1), \textcolor{red}{L(x_1,z_1)}\}$ and $\{\neg \textcolor{red}{L(a,d)}\}$:
[$x_1=a, z_1=d$] with resolvent $\{\neg L(a, y_1), \neg L(y_1,d)\}$
- 3: Unify and resolve $\{\neg L(x_2,y_2), \neg L(y_2,z_2), \textcolor{red}{L(x_2,z_2)}\}$ and $\{\neg \textcolor{red}{L(a,y_3)}, \neg L(y_3,d)\}$:
[$x_2=a, z_2=y_3$] with resolvent $\{\neg L(a, y_2), \neg L(\textcolor{blue}{y_2}, \textcolor{blue}{y_3}), \neg L(y_3,d)\}$
- 4: Unify and resolve $\{\neg \textcolor{red}{L(a, y_4)}, \neg L(y_4, y_5), \neg L(y_5,d)\}$ and $\{\textcolor{red}{L(a,b)}\}$:
[$y_4=b$] with resolvent $\{\neg L(b, y_5), \neg L(y_5,d)\}$
- 5: Unify and resolve $\{\neg \textcolor{red}{L(b, y_5)}, \neg L(y_5,d)\}$ and $\{\textcolor{red}{L(b,c)}\}$:
[$y_5=c$] with resolvent $\{\neg L(c,d)\}$
- 5: Resolve $\{\neg L(c,d)\}$ and $\{\textcolor{red}{L(c,d)}\}$:
resolvent $\{\}$
(success)

Resolution with unification for L_{FO}

A correct procedure for $\Gamma \vdash \varphi$ in L_{FO}

- a) Refutation $\Gamma \cup \{\neg\varphi\}$,
- b) Prenex normal form and skolemization $sko(\Gamma \cup \{\neg\varphi\})$
- c) Translation of $sko(\Gamma \cup \{\neg\varphi\})$ into CNF hence into CF
- d) Repeat application of the resolution method:
 - 1) Selection of two clauses $\{\beta_1, \beta_2, \dots, \beta_n, \alpha\}, \{\neg\alpha', \gamma_1, \gamma_2, \dots, \gamma_m\}$
 - 2) *Standardization* of variables
(i.e. create new copies of the two clauses having new and unique variables)
 - 3) Construction of the MGU μ (if it exists) for the two literals α e α'
 - 4) Generation of the resolvent by applying of μ
 $\{\beta_1, \beta_2, \dots, \beta_n, \alpha\}[\mu], \{\neg\alpha', \gamma_1, \gamma_2, \dots, \gamma_m\}[\mu] \vdash \{\beta_1, \beta_2, \dots, \beta_n, \gamma_1, \gamma_2, \dots, \gamma_m\}[\mu]$
- e) Until
 - 1) The empty clause has been derived (*success*)
 - 2) No further resolutions are possible – *fixed point (failure)*

Note: the method is not guaranteed to terminate (i.e. it might *diverge*)

The method might diverge...

Problem: $\{ \forall x (Q(f(x)) \rightarrow P(x)) \} \models \exists x (P(f(x)) \wedge \neg Q(f(x))) ?$

(The answer is negative: there is no entailment)

Refutation:

$\{ \forall x (Q(f(x)) \rightarrow P(x)) \} \cup \{ \neg \exists x (P(f(x)) \wedge \neg Q(f(x))) \}$

Prenex normal form:

$\{ \forall x (Q(f(x)) \rightarrow P(x)) \} \cup \{ \forall x \neg (P(f(x)) \wedge \neg Q(f(x))) \}$

(no skolemization required)

Clausal form:

$\{ Q(f(x)) \rightarrow P(x) \} \cup \{ \neg (P(f(x)) \wedge \neg Q(f(x))) \}$

$\{ \neg Q(f(x)) \vee P(x) \} \cup \{ \neg P(f(x)) \vee Q(f(x)) \}$

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

Resolution:

1: $\{ \neg Q(f(x_1)), P(x_1) \}, \{ \neg P(f(x_2)), Q(f(x_2)) \}, [x_1/f(x_2)] \vdash \{ \neg Q(f(f(x_2))), Q(f(x_2)) \}$

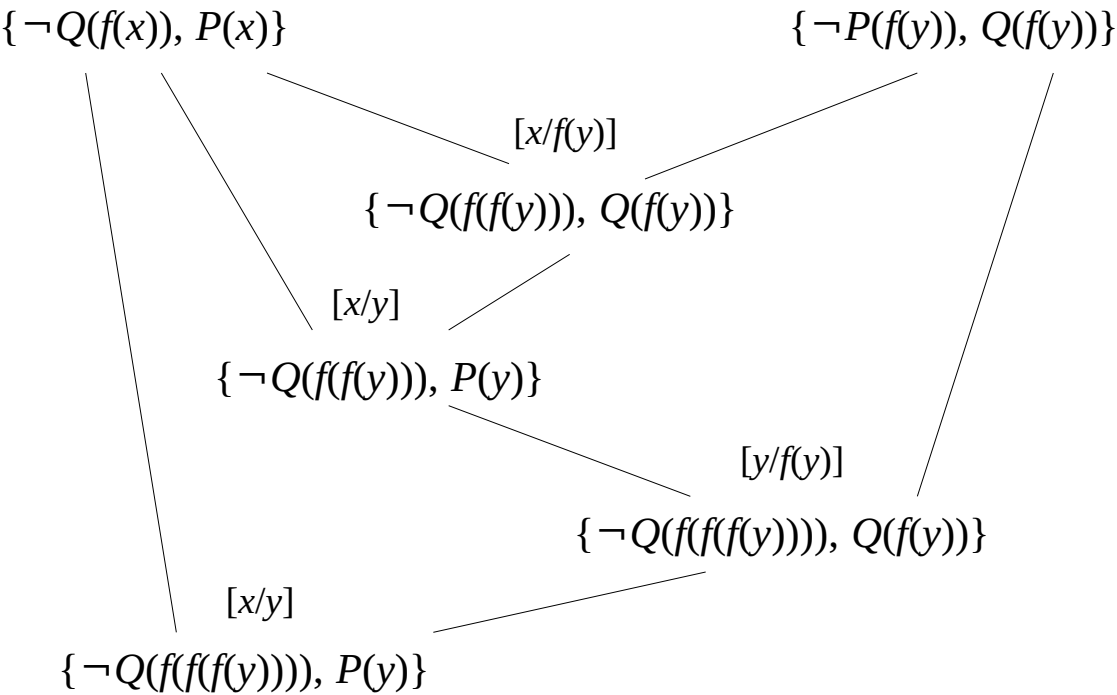
2: $\{ \neg Q(f(x_3)), P(x_3) \}, \{ \neg Q(f(f(x_4))), Q(f(x_4)) \}, [x_3/x_4] \vdash \{ \neg Q(f(f(x_4))), P(x_4) \}$

3: $\{ \neg Q(f(f(x_5))), P(x_5) \}, \{ \neg P(f(x_6)), Q(f(x_6)) \}, [x_5/f(x_6)] \vdash \{ \neg Q(f(f(f(x_6)))), Q(f(x_6)) \}$

4: $\{ \neg Q(f(x_7)), P(x_7) \}, , \{ \neg Q(f(f(f(x_8)))), Q(f(x_8)) \}, [x_7/x_8] \vdash \{ \neg Q(f(f(f(x_8)))), P(x_8) \}$

...

The method might diverge...



· (Standardization of variables not shown, for simplicity)
·
·

Properties of resolution with unification

- The method is *correct* in L_{FO}

If the method finds the empty clause for $sko(\Gamma \cup \{\neg\varphi\})$ then $\Gamma \models \varphi$

- Is the method *complete* in L_{FO} ?

Within the limits of semi-decidability, yes (Robinson, 1963)

When $\Gamma \models \varphi$, the method will eventually find the empty clause for $sko(\Gamma \cup \{\neg\varphi\})$

Very often (but not in the worst case) the method is more efficient than the one in the corollary of Herbrand's theorem

The advantage is due to *lifting*

(the method can resolve also non-ground clauses)

When $\Gamma \not\models \varphi$, the method might diverge

CAUTION: Unless the selection procedure is fair (more on this topic to follow) the method might diverge even when $\Gamma \models \varphi$

Critical aspect:

- Selecting the clauses and literals to be resolved