M.Lupea

Resolution in predicate Logic

- ***Normal forms of predicate (first-order) formulas***
  - **prenex, Skolem, clausal normal forms**
- ***Substitutions and unifiers***
- ***Resolution formal system***
- ***Refinements of predicate resolution***

Prenex normal form

M.Lupea

Definition

A predicate formula U is in ***prenex normal form*** if it has the form: $(Q_1x_1)\ldots(Q_nx_n)M$, where $Q_i, i=1,\ldots,n$ are quantifiers, and M is quantifier-free. The sequence $(Q_1x_1)\ldots(Q_nx_n)$ is called the ***prefix*** *of the formula* U, and M is called the ***matrix*** *of the formula* U. A predicate formula is in ***conjunctive prenex normal form*** if it is in prenex normal form and the matrix is in CNF.

Theorem:

A predicate formula admits a logical equivalent conjunctive prenex normal form.

The **prenex normal form** is obtained by applying transformations which preserve the logical equivalence, according to the following steps:

Step 1: The connectives '$\rightarrow$' and '$\leftrightarrow$' are replaced using the connectives: $\neg, \wedge, \vee$

Step 2: The bound variables are renamed such that they will be distinct.

Step 3: Application of infinitary DeMorgan's laws.

Step 4: The extraction of quantifiers in front of the formula.

Step 5: The matrix is transformed into CNF using DeMorgan's laws and the distributive laws.

Extraction of quantifiers in front of the formula

$A \vee (\exists x)B(x) \equiv (\exists x)(A \vee B(x))$	$A \vee (\forall x)B(x) \equiv (\forall x)(A \vee B(x))$
$A \wedge (\exists x)B(x) \equiv (\exists x)(A \wedge B(x))$	$A \wedge (\forall x)B(x) \equiv (\forall x)(A \wedge B(x))$
where A does not contain x as a free variable.	
$(\exists x)A(x) \vee B \equiv (\exists x)(A(x) \vee B)$	$(\forall x)A(x) \vee B \equiv (\forall x)(A(x) \vee B)$
$(\exists x)A(x) \wedge B \equiv (\exists x)(A(x) \wedge B)$	$(\forall x)A(x) \wedge B \equiv (\forall x)(A(x) \wedge B)$
where B does not contain x as a free variable.	

Skolem and clausal normal forms

M.Lupea

Definitions:

Let U be a first-order formula, and $U^p = (Q_1x_1)...(Q_nx_n)M$ be one of its conjunctive prenex normal form. A formula in ***Skolem normal form***, denoted by U^S corresponds to U and it is obtained as follows:

- For each existential quantifier Q_r from the prefix we apply the transformation:
  - if on the left side of Q_r there are no universal quantifiers, then we introduce a new constant ***a***, and we replace in M all the occurrences of x_r by ***a***. (Q_rx_r) is deleted from the prefix.
  - if $Q_{s_1},\ldots,Q_{s_m}, 1 \le s_1 < \ldots < s_m < r$, are all the universal quantifiers on the left side of Q_r in the prefix, then we introduce a new m-place function symbol, f, and we replace in M all the occurrences of x_r by $f(x_{s_1},\ldots,x_{s_m})$. (Q_rx_r) is deleted from the prefix.
- The constants and functions used to replace the existentially quantified variables are called ***Skolem constants*** and ***Skolem functions***. The prefix of the formula U^S contains only universal quantifiers, and the matrix is in conjunctive normal form.

A formula in ***clausal normal form*** denoted by U^c is obtained by deleting the prefix of U^S.

Theoretical results

M.Lupea

Remarks:

- In **Step 4** we begin with the innermost quantifiers and we extract them in front of the corresponding parent subformulas, paying attention to the scope of each quantifier and we continue in this manner to the outside of the formula.
  If the obtained formula contains n distinct and independent groups of quantifiers, these groups can be extracted in an arbitrary order, therefore
  there exist $n!$ prenex normal forms, logically equivalent to the initial formula.
- The **transformations** used in the Skolemization process **do not preserve the logical equivalence** but they **preserve the inconsistency** according to the next theorem.

Theorem:

Let $U_1, U_2, \ldots, U_n, V$ be first-order formulas.

1. V is inconsistent *if and only if* V^p is inconsistent *if and only if* V^S is inconsistent, *if and only if* V^c is inconsistent.	2. The set $\{U_1, U_2, \ldots, U_n\}$ is inconsistent *if and only if* the set $\{U_1^c, U_2^c, \ldots, U_n^c\}$ is inconsistent.

Example 1 Transform into normal forms the formula:

M.Lupea

$$U = \neg((\forall x)(P(x) \rightarrow Q(x)) \rightarrow ((\forall x)P(x) \rightarrow (\forall x)Q(x)))$$

$$U = \neg((\forall x)(P(x) \overset{1}{\rightarrow} Q(x)) \overset{2}{\rightarrow} ((\forall x)P(x) \overset{3}{\rightarrow} (\forall x)Q(x)))$$

replace '$\rightarrow$' connectives, denoted by 1 and 3

$$U \equiv \neg((\forall x)(\neg P(x) \vee Q(x)) \rightarrow (\neg(\forall x)P(x) \vee (\forall x)Q(x)))$$

replace '$\rightarrow$' connective

$$U \equiv \neg(\neg(\forall x)(\neg P(x) \vee Q(x)) \vee (\neg(\forall x)P(x) \vee (\forall x)Q(x)))$$

rename the bound variables

$$U \equiv \neg(\neg(\forall x)(\neg P(x) \vee Q(x)) \vee \neg(\forall y)P(y) \vee (\forall z)Q(z))$$

apply DeMorgan's laws

$$U \equiv (\forall x)(\neg P(x) \vee Q(x)) \wedge (\forall y)P(y) \wedge (\exists z)\neg Q(z)$$

extract the quantifiers in front of the formula

From all 6 **prenex forms** we choose 3:

$$U \equiv (\exists z)(\forall x)(\forall y)((\neg P(x) \vee Q(x)) \wedge P(y) \wedge \neg Q(z)) = U_1^{\,p}$$

$$U \equiv (\forall x)(\exists z)(\forall y)((\neg P(x) \vee Q(x)) \wedge P(y) \wedge \neg Q(z)) = U_2^{\,p}$$

$$U \equiv (\forall x)(\forall y)(\exists z)((\neg P(x) \vee Q(x)) \wedge P(y) \wedge \neg Q(z)) = U_3^{\,p}$$

Skolemization process applied to the formulas $U_1^{\,p}, U_2^{\,p}, U_3^{\,p}$ => **Skolem forms**

$$U_1^{\,S} = (\forall x)(\forall y)((\neg P(x) \vee Q(x)) \wedge P(y) \wedge \neg Q(a)),$$

$[z \leftarrow a]$, $\boldsymbol{a}$ is Skolem constant

$$U_2^{\,S} = (\forall x)(\forall y)((\neg P(x) \vee Q(x)) \wedge P(y) \wedge \neg Q(f(x)))$$

$[z \leftarrow f(x)]$, $\boldsymbol{f}$ is a unary Skolem function

$$U_3^{\,S} = (\forall x)(\forall y)((\neg P(x) \vee Q(x)) \wedge P(y) \wedge \neg Q(g(x,y)))$$

$[z \leftarrow g(x,y)]$, $\boldsymbol{g}$ is a binary Skolem function

Clausal forms:

$$U_1^{\,c} = (\neg P(x) \vee Q(x)) \wedge P(y) \wedge \neg Q(a)$$

$$U_2^{\,c} = (\neg P(x) \vee Q(x)) \wedge P(y) \wedge \neg Q(f(x))$$

$$U_3^{\,c} = (\neg P(x) \vee Q(x)) \wedge P(y) \wedge \neg Q(g(x,y))$$

Example 2

M.Lupea

Transform into clausal normal form the formula:

$$U=(\exists x)(\forall y)P(x,y)\vee(\exists z)(\neg Q(z)\vee(\forall u)(\exists t)R(z,u,t))$$

We extract first the group of quantifiers $(\forall u)(\exists t)$ in front of its parent subformula, after $(\exists z)$ to keep the scope of each of the three quantifiers:

$$U\equiv(\exists x)(\forall y)P(x,y)\vee(\exists z)(\forall u)(\exists t)(\neg Q(z)\vee R(z,u,t))$$

Now there are two groups of distinct and independent quantifiers: $(\exists x)(\forall y)$ and $(\exists z)(\forall u)(\exists t)$ and they can be extracted in two different ways.

To be noted that the order of the quantifiers inside a group cannot be changed.

We choose:

$$U^P=(\exists x)(\forall y)(\exists z)(\forall u)(\exists t)(P(x,y)\vee\neg Q(z)\vee R(z,u,t))$$

$$U^S=(\forall y)(\forall u)(P(a,y)\vee\neg Q(f(y))\vee R(f(y),u,g(y,u))),$$

$[x\leftarrow a],[z\leftarrow f(y)],[t\leftarrow g(y,u)]$, a - Skolem constant, f,g - Skolem functions

$$U^c=P(a,y)\vee\neg Q(f(y))\vee R(f(y),u,g(y,u))$$

Substitutions

M.Lupea

Definition:

A ***substitution*** is a mapping from the set of variables ***Var*** into the set of terms: ***TERMS***.

We denote by $\theta = [x_1 \leftarrow t_1, \ldots, x_k \leftarrow t_k]$, a substitution, representing a finite set of replacements of variables, where $x_1, \ldots, x_k$ are distinct variables, $t_1, \ldots, t_k$ are terms, such that $\forall i = 1, \ldots, k$: $t_i \neq x_i$ and x_i is not a subterm of t_i. $dom(\theta) = \{x_1, \ldots, x_k\}$ is called the ***domain*** of θ.

Definition:

The result of applying the substitution $\theta = [x_1 \leftarrow t_1, \ldots, x_k \leftarrow t_k]$ to a formula is defined recursively as follows:

$\theta(x_i) = t_i, x_i \in dom(\theta)$; $\theta(x) = x, x \notin dom(\theta)$; $\theta(c) = c, c$ – constant	$\theta(f(t_1,\ldots,t_n)) = f(\theta(t_1),\ldots,\theta(t_n)), f \in \mathrm{F}_n$ $\theta(p(t_1,\ldots,t_n)) = p(\theta(t_1),\ldots,\theta(t_n)), p \in \mathrm{P}_n$ $\theta(\neg U) = \neg\theta(U)$ $\theta(U \circ V) = \theta(U) \circ \theta(V), \circ \in \{\vee, \wedge, \rightarrow, \leftrightarrow\}$
An instance $\theta(U)$ of U is obtained by replacing simultaneously each occurrence of x_i in U by t_i.	

Substitutions and unifiers

M.Lupea

Definition:
The ***composition*** of two substitutions:

$\theta_1 = [x_1 \leftarrow t_1, \dots, x_k \leftarrow t_k]$ and $\theta_2 = [y_1 \leftarrow s_1, \dots, y_n \leftarrow s_n]$ is defined as:

$$\theta = \theta_1\theta_2 = [x_i \leftarrow \theta_2(t_i) \mid x_i \in dom(\theta_1), x_i \neq \theta_2(t_i)] \cup [y_j \leftarrow s_j \mid y_j \in dom(\theta_2) \setminus dom(\theta_1)]$$

Properties: Let $\theta, \theta_1, \theta_2, \theta_3$ be substitutions.

- $\varepsilon\theta = \theta\varepsilon = \theta$; ε - ***empty substitution***;
- $\theta_1(\theta_2\theta_3) = (\theta_1\theta_2)\theta_3$, ***associativity property***;
- $\theta_1\theta_2 \neq \theta_2\theta_1$, the ***composition*** of two substitutions ***is not commutative*** in general.

Definition:

- A substitution θ is a ***unifier*** of the terms t_1 and t_2 if $\theta(t_1) = \theta(t_2)$.
  The term $\theta(t_1)$ is called the ***common instance*** of the unified terms.
- A ***unifier of a set*** $\{U_1, \dots, U_n\}$ of formulas is a substitution θ such that
  $\theta(U_1) = \dots = \theta(U_n)$.
- The ***most general unifier*** (mgu) is a unifier μ such that any other unifier θ can be obtained from μ by a further substitution λ, $\theta = \mu\lambda$.

Example 3

Let $\theta_1 = [x \leftarrow f(y), y \leftarrow f(a), z \leftarrow u]$ and

$\theta_2 = [y \leftarrow g(a), u \leftarrow z, v \leftarrow f(f(a))]$ be two substitutions.

$$\lambda_1 = \theta_1\theta_2 = [x \leftarrow \theta_2(f(y)), y \leftarrow \theta_2(f(a)), z \leftarrow \theta_2(u)] \cup [u \leftarrow z, v \leftarrow f(f(a))] =$$

$$= [x \leftarrow f(g(a)), y \leftarrow f(a), z \leftarrow z, u \leftarrow z, v \leftarrow f(f(a))] =$$

$$= [x \leftarrow f(g(a)), y \leftarrow f(a), u \leftarrow z, v \leftarrow f(f(a))]$$

$$\lambda_2 = \theta_2\theta_1 = [y \leftarrow g(a), v \leftarrow f(f(a)), x \leftarrow f(y), z \leftarrow u]$$

$\lambda_1 \neq \lambda_2$, so **the composition of two substitutions is not commutative in general.**

Algorithm for computing the mgu of two literals

M.Lupea

input: $l_1 = P_1(t_{1_1}, t_{1_2}, \ldots, t_{1_n})$ and $l_2 = P_2(t_{2_1}, t_{2_2}, \ldots, t_{2_k})$ two literals

output: $mgu(l_1, l_2)$ or the message "l_1, l_2 are not unifiable"

begin

  if ($P_1 \neq P_2$) **then** write "l_1, l_2 are not unifiable"; exit; **end_if**

  if ($n \neq k$) **then** write "l_1, l_2 are not unifiable"; exit; **end_if**

  $\theta := \varepsilon$; // initialization with empty substitution

  while ($\theta(l_1) \neq \theta(l_2)$)

    find in $\theta(l_1)$, $\theta(l_2)$ the terms corresponding to the outermost function symbols

      or variables that are different and denote them by t_1 and t_2.

    if (neither one of t_1 and t_2 is a variable or one is a subterm of the other one)

      then write "l_1 and l_2 are not unifiable"; exit; **end_if**

    if (t_1 is a variable) // λ is the unifier of the terms t_1, t_2 in the current iteration

      then $\lambda := [t_1 \leftarrow t_2]$; **else** $\lambda := [t_2 \leftarrow t_1]$; **end_if**

    $\theta := \theta\lambda$;

    if (θ is not a substitution) **then** write "l_1 and l_2 are not unifiable"; exit; **end_if**

  end_while

  write "l_1 and l_2 are unifiable and θ is $mgu(l_1, l_2)$"

end

Example 4

M.Lupea

Find the **most general unifier (mgu)** of the literals $l_1 = P(a,x,f(g(y)))$ and $l_2 = P(y,f(z),f(z))$, where $x,y,z \in Var, a \in Const,\ f,g \in F_1,\ P \in P_3$.	
$\theta := \varepsilon$ $\theta(l_1) = P(\underline{a},x,f(g(y)))$, $\theta(l_2) = P(\underline{y},f(z),f(z))$	**second iteration:** $\lambda := [x \leftarrow f(z)]$; unifier of x and $f(z)$; $\theta := \theta\lambda = [y \leftarrow a][x \leftarrow f(z)] = [y \leftarrow a, x \leftarrow f(z)]$; $\theta(l_1) = P(a,f(z),f(\underline{g(a)}))$, $\theta(l_2) = P(a,f(z),f(\underline{z}))$
first iteration: $\lambda := [y \leftarrow a]$; unifier of the terms y and a $\theta := \theta\lambda = [y \leftarrow a]$; $\theta(l_1) = P(a,\underline{x},f(g(a)))$, $\theta(l_2) = P(a,\underline{f(z)},f(z))$	**third iteration:** $\lambda := [z \leftarrow g(a)]$; unifier of the terms z *and* $g(a)$ $\theta := \theta\lambda =$ $= [y \leftarrow a, x \leftarrow f(z)][z \leftarrow g(a)] =$ $[y \leftarrow a, x \leftarrow f(g(a)), z \leftarrow g(a)] = mgu(l_1,l_2)$ $\theta(l_1) = \theta(l_2) = P(a,f(g(a)),f(g(a)))$ is the common instance of l_1 and l_2

Example 5

M.Lupea

Check whether the literals $l_1 = Q(x, a, f(x,y))$ and $l_2 = Q(b, y, f(z,c))$, are unifiable or not, where $x, y, z \in Var$, $a, b, c \in Const$, $f \in F_2$, $Q \in P_3$.

$\theta := \varepsilon$;

$\theta(l_1) = Q(\underline{x}, a, f(x,y))$

$\theta(l_2) = Q(\underline{b}, y, f(z,c))$;

first iteration:

$\lambda := [x \leftarrow b]$, unifier of the terms x and b

$\theta := \theta\lambda = [x \leftarrow b]$;

$\theta(l_1) = Q(b, \underline{a}, f(b,y))$, $\theta(l_2) = Q(b, \underline{y}, f(z,c))$;

second iteration:

$\lambda := [y \leftarrow a]$, unifier of the terms y and a

$\theta := \theta\lambda$

$\theta := [x \leftarrow b][y \leftarrow a] = [x \leftarrow b, y \leftarrow a]$,

$\theta(l_1) = Q(b, a, f(\underline{b}, a))$, $\theta(l_2) = Q(b, a, f(\underline{z}, c))$;

third iteration:

$\lambda := [z \leftarrow b]$, unifier of the terms z and b

$\theta := \theta\lambda$

$\theta := [x \leftarrow b, y \leftarrow a][z \leftarrow b] = [x \leftarrow b, y \leftarrow a, z \leftarrow b]$

$\theta(l_1) = Q(b, a, f(b, \underline{a}))$, $\theta(l_2) = Q(b, a, f(b, \underline{c}))$;

fourth iteration:

The terms a and c, which are distinct constants, are not unifiable, so the terms $f(b,a)$ and $f(z,c)$ are not unifiable and we conclude that the ***literals*** l_1 ***and*** l_2 ***are not unifiable.***

Predicate resolution
- formal (axiomatic system) -

$\mathrm{Res}^{\mathrm{Pr}} = (\Sigma_{\mathrm{Res}}^{\mathrm{Pr}}, F_{\mathrm{Res}}^{\mathrm{Pr}}, A_{\mathrm{Res}}^{\mathrm{Pr}}, R_{\mathrm{Res}}^{\mathrm{Pr}})$, where:

- $\Sigma_{\mathrm{Res}}^{\mathrm{Pr}} = \Sigma_{\mathrm{Pr}} - \{\rightarrow, \leftrightarrow, \wedge, \exists, \forall\}$ is the ***alphabet***;
- $F_{\mathrm{Res}}^{\mathrm{Pr}} \cup \{\square\}$ is the set of ***well- formed formulas***;
  - $F_{\mathrm{Res}}^{\mathrm{Pr}}$ is the set of all clauses built using the alphabet $\Sigma_{\mathrm{Res}}^{\mathrm{Pr}}$;
  - $\square$ is ***empty clause***;
- $A_{\mathrm{Res}}^{\mathrm{Pr}} = \varnothing$ is the ***set of axioms***;
- $R_{\mathrm{Res}}^{\mathrm{Pr}} = \{res^{\mathrm{Pr}}, fact\}$ is the ***set of inference rules*** containing the ***resolution rule*** (res^{Pr}) and the ***factoring rule*** ($fact$).

$f \vee l_1, g \vee \neg l_2 \vdash_{res^{\mathrm{Pr}}} \lambda(f) \vee \lambda(g)$, where $\lambda = mgu(l_1, l_2)$, $f, g \in F_{\mathrm{Res}}^{\mathrm{Pr}}$.

$l_1 \vee l_2 \vee \ldots \vee l_k \vee l_{k+1} \vee \ldots \vee l_n \vdash_{fact} \lambda(l_1 \vee l_{k+1} \vee \ldots \vee l_n)$ where $\lambda = mgu(l_1, l_2, \ldots, l_k)$

Definitions

M.Lupea

Definition:

- The predicate clauses $C_1 = f \vee l_1$ and $C_2 = g \vee \neg l_2$, without common free variables, are called ***clashing clauses*** if the literals l_1 and l_2 are unifiable: there exists $\lambda = mgu(l_1, l_2)$.
- The ***binary resolvent*** of C_1 and C_2 is $C = \mathrm{Res}_{\lambda}^{\mathrm{Pr}}(C_1, C_2) = \lambda(f) \vee \lambda(g)$.
- If $C = l_1 \vee l_2 \vee \ldots \vee l_k \vee l_{k+1} \vee \ldots \vee l_n$; $l_1, l_2, \ldots, l_n$ - literals and $\lambda = mgu(l_1, l_2, \ldots, l_k)$, $Fact(C) = \lambda(l_1) \vee \lambda(l_{k+1}) \vee \ldots \vee \lambda(l_n)$ is called a ***factor*** of C.

Definition:

The ***predicate resolvent*** of two parent clauses C_1 and C_2 is one of the following:

1. the binary resolvent of C_1 and C_2;
2. the binary resolvent of C_1 and a factor of C_2;
3. the binary resolvent of a factor of C_1 and C_2;
4. the binary resolvent of a factor of C_1 and a factor of C_2.

Algorithm: *Predicate Resolution*

input: $U_1, U_2, \ldots U_n, V$ - first-order formulas.

output: message: " $U_1, U_2, \ldots U_n \vdash V$ " or „ $U_1, U_2, \ldots U_n \nvdash V$ " or
"we cannot decide if $U_1, U_2, \ldots U_n \vdash V$ or $U_1, U_2, \ldots U_n \nvdash V$ "

begin
build the set of clauses: $S := \{U_1^c, U_2^c, \ldots, U_n^c, (\neg V)^c\}$;
do{
select l_1, l_2, C_1, C_2 such that:
C_1, C_2 are clauses or factors of clauses of S;
l_1, l_2 are literals, $l_1 \in C_1$, and $\neg l_2 \in C_2$;
if (l_1 and l_2 are unifiable with $\theta := mgu\,(l_1, l_2)$) **then**
$C := \mathrm{Res}_{\theta}^{\mathrm{Pr}}(C_1, C_2)$;
if ($C = \square$) **then** write " $U_1, U_2, \ldots U_n \vdash V$ "; **exit**;
else $S := S \cup \{C\}$;
end_if
end_if
}**until** (no new resolvents can be derived **or**
a predefined quantity of effort was done)
if (no new resolvents can be derived) **then** write " $U_1, U_2, \ldots U_n \nvdash V$ ";
else write "we cannot decide: $U_1, U_2, \ldots U_n \vdash V$ or $U_1, U_2, \ldots U_n \nvdash V$ "
end_if
end

Theoretical results

M.Lupea

Theorem (*soundness* and *completeness* of predicate resolution)

A set S o predicate(first-order) clauses is **inconsistent** *if and only if* $S \vdash_{\text{Res}}^{\text{Pr}} \square$.

Theorem (*resolution* – a refutation proof method)

Let $U_1, U_2, \ldots U_n, V$ be first-order formulas.

- $\vdash V \ (\models V)$ *if and only if* $(\neg V)^c \vdash_{\text{Res}}^{\text{Pr}} \square$.
- $U_1, U_2, \ldots, U_n \vdash V$ *if and only if* $\{U_1^c, U_2^c, \ldots, U_n^c, (\neg V)^c\} \vdash_{\text{Res}}^{\text{Pr}} \square$.

Remarks:

- All the refinements and strategies of propositional resolution can be used in predicate logic.
- The resolution algorithm for predicate logic is a ***semi-decision procedure.***

Example 6

M.Lupea

Check the **inconsistency** of the set S of clauses using **lock resolution**.

$$S = \{\neg P(x) \vee Q(x) \vee R(x), \neg Q(y) \vee R(y), P(a), \neg R(a)\}$$

The literals are indexed as follows:

$C_1 = {}_{(3)}\neg P(x) \vee {}_{(2)} Q(x) \vee {}_{(1)} R(x)$ $\qquad$ $C_2 = {}_{(5)}\neg Q(y) \vee {}_{(4)} R(y)$

$C_3 = {}_{(6)} P(a),$ $\qquad$ $C_4 = {}_{(7)}\neg R(a)$

The following resolvents are obtained:

$$C_5 = \text{Res}^{\text{Pr}}_{\theta 1 = [x \leftarrow a]}(C_1, C_4) = {}_{(3)}\neg P(a) \vee {}_{(2)} Q(a)$$

$$C_6 = \text{Res}^{\text{Pr}}_{\theta 2 = [y \leftarrow a]}(C_2, C_4) = {}_{(5)}\neg Q(a)$$

$$C_7 = \text{Res}^{\text{Pr}}(C_5, C_6) = {}_{(3)}\neg P(a)$$

$$C_8 = \text{Res}^{\text{Pr}}(C_3, C_7) = \square$$

$S \vdash^{lock, \text{Pr}}_{\text{Res}} \square$ and thus S *is **inconsistent***.

Example 7

M.Lupea

Using **linear resolution** prove that $S=\{C_1,C_2,C_3,C_4\}$ is an ***inconsistent*** set of clauses:

$C_1 = P(x, f(x), e)$, $C_2 = \neg R(x) \vee \neg R(y) \vee \neg P(x, f(y), z) \vee R(z)$,

$C_3 = R(a)$, $C_4 = \neg R(e)$

Constants: a, e, function symbols: f, predicate symbols: P, R

For linear resolution we choose the *top clause*: C_4

$C_2 \vdash_{fact}^{[y\leftarrow x]} C_5 = \neg R(x) \vee \neg P(x, f(x), z) \vee R(z)$,

C_5 is a factor of C_2

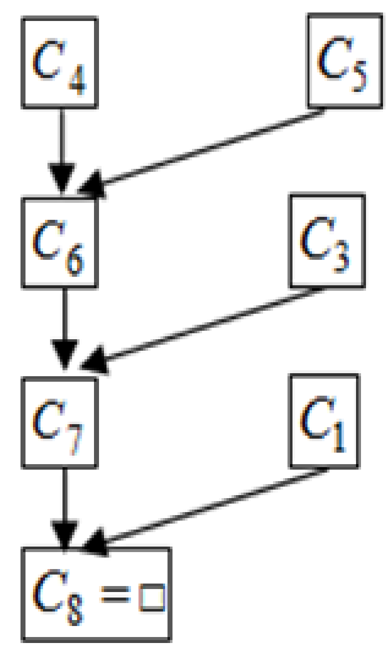
$C_4, C_5 \vdash_{res^{Pr}}^{[z\leftarrow e]} C_6 = \neg R(x) \vee \neg P(x, f(x), e)$, *central clause*

$C_6, C_3 \vdash_{res^{Pr}}^{[x\leftarrow a]} C_7 = \neg P(a, f(a), e)$, *central clause*

$C_7, C_1 \vdash_{res^{Pr}}^{[x\leftarrow a]} C_8 = \square$

side clauses: C_5, C_3, C_1.

From the set S we derived the empty clause,
therefore S *is* ***inconsistent***.



Example 8

M.Lupea

Prove the **semidistributivity of '$\forall$' over '$\rightarrow$'**:

$$\vdash (\forall x)(P(x) \rightarrow Q(x)) \rightarrow ((\forall x)P(x) \rightarrow (\forall x)Q(x))$$ and

$$\nvdash ((\forall x)P(x) \rightarrow (\forall x)Q(x)) \rightarrow (\forall x)(P(x) \rightarrow Q(x))$$

by applying predicate resolution.

We consider the predicate formulas.

$$U_1 = (\forall x)(P(x) \rightarrow Q(x)) \rightarrow ((\forall x)P(x) \rightarrow (\forall x)Q(x))$$ and

$$U_2 = ((\forall x)P(x) \rightarrow (\forall x)Q(x)) \rightarrow (\forall x)(P(x) \rightarrow Q(x))$$

We apply the theoretical results:

- $\vdash U_1$ *if and only if* $(\neg U_1)^c \vdash_{\text{Res}}^{\text{Pr}} \square$ and
- $\nvdash U_2$ *if and only if* $(\neg U_2)^c \nvdash_{\text{Res}}^{\text{Pr}} \square$.

Example 8 (contd.)

M.Lupea

Normal forms	**Resolution process**
$\neg U_1 = \neg((\forall x)(P(x) \to Q(x)) \to ((\forall x)P(x) \to (\forall x)Q(x)))$ replace '$\to$': $\equiv (\forall x)(P(x) \to Q(x)) \wedge \neg((\forall x)P(x) \to (\forall x)Q(x)) \equiv$ $\equiv (\forall x)(P(x) \to Q(x)) \wedge (\forall x)P(x) \wedge \neg(\forall x)Q(x) \equiv$ $\equiv (\forall x)(\neg P(x) \vee Q(x)) \wedge (\forall x)P(x) \wedge \neg(\forall x)Q(x)$ infinitary DeMorgan's law is applied $\equiv (\forall x)(\neg P(x) \vee Q(x)) \wedge (\forall x)P(x) \wedge (\exists x)\neg Q(x)$ rename the bound variables $\equiv (\forall x)(\neg P(x) \vee Q(x)) \wedge (\forall y)P(y) \wedge (\exists z)\neg Q(z)$ extraction of the quantifiers, '$\exists$' the first one extracted $(\neg U_1)^p = (\exists z)(\forall x)(\forall y)((\neg P(x) \vee Q(x)) \wedge P(y) \wedge \neg Q(z))$ **prenex form** $[z \leftarrow a]$, a- Skolem constant $(\neg U_1)^S = (\forall x)(\forall y)((\neg P(x) \vee Q(x)) \wedge P(y) \wedge \neg Q(a))$ **Skolem form** $(\neg U_1)^c = (\neg P(x) \vee Q(x)) \wedge P(y) \wedge \neg Q(a)$ - **clausal normal form**	The set of clauses: $S_1 = \{C_1 = \neg P(x) \vee Q(x),$ $C_2 = P(y),$ $C_3 = \neg Q(a)\}$ The **resolvents**: $C_4 = \mathrm{Res}^{\mathrm{Pr}}_{[x \leftarrow a]}(C_1, C_3) = \neg P(a)$ $C_5 = \mathrm{Res}^{\mathrm{Pr}}_{[y \leftarrow a]}(C_2, C_4) = \square$ We proved: $(\neg U_1)^c \vdash^{\mathrm{Pr}}_{\mathrm{Res}} \square$, Therefore: $\vdash U_1$

Example 8(contd.)

M.Lupea

Normal forms	**Resolution process**
$\neg U_2 = \neg(((\forall x)P(x) \to (\forall x)Q(x)) \to (\forall x)(P(x) \to Q(x)))$ replace '$\to$' and apply infinitary DeMorgan's law $\equiv ((\forall x)P(x) \to (\forall x)Q(x)) \wedge \neg(\forall x)(P(x) \to Q(x))$ $\equiv (\neg(\forall x)P(x) \vee (\forall x)Q(x)) \wedge (\exists x)(P(x)\neg Q(x))$ $\equiv ((\exists x)\neg P(x) \vee (\forall x)Q(x)) \wedge (\exists x)(P(x) \wedge \neg Q(x))$ rename the bound variables $\equiv ((\exists x)\neg P(x) \vee (\forall y)Q(y)) \wedge (\exists z)(P(z) \wedge \neg Q(z))$ extraction of the quantifiers, '$\exists$'s extracted first $(\neg U_2)^p = (\exists z)(\exists x)(\forall y)((\neg P(x) \vee Q(y)) \wedge P(z) \wedge \neg Q(z))$ **prenex form** $[z \leftarrow a, x \leftarrow b]$, a, b - Skolem constants $(\neg U_2)^S = (\forall y)((\neg P(b) \vee Q(y)) \wedge P(a) \wedge \neg Q(a))$ **Skolem form** $(\neg U_2)^c = (\neg P(b) \vee Q(y)) \wedge P(a) \wedge \neg Q(a)$ - **clausal normal form**	The set of clauses: $S_2 = \{C_1' = \neg P(b) \vee Q(y),$ $C_2' = P(a),$ $C_3' = \neg Q(a)\}$ The only **resolvent** is: $C_4' = \mathrm{Res}^{\mathrm{Pr}}_{[y \leftarrow a]}(C_1', C_3') = \neg P(b)$ $P(a), P(b)$ are not unifiable because a, b are distinct constants, so the clauses $C_2' = P(a), C_4' = \neg P(b)$ do not resolve. $(\neg U_2)^c \not\vdash^{\mathrm{Pr}}_{\mathrm{Res}} \square$, so $\not\models U_2$.
$\models U_1$ and $\not\models U_2$, '$\forall$' *is not distributive over the connective* '$\to$', *it is only semi-distributive.*	