

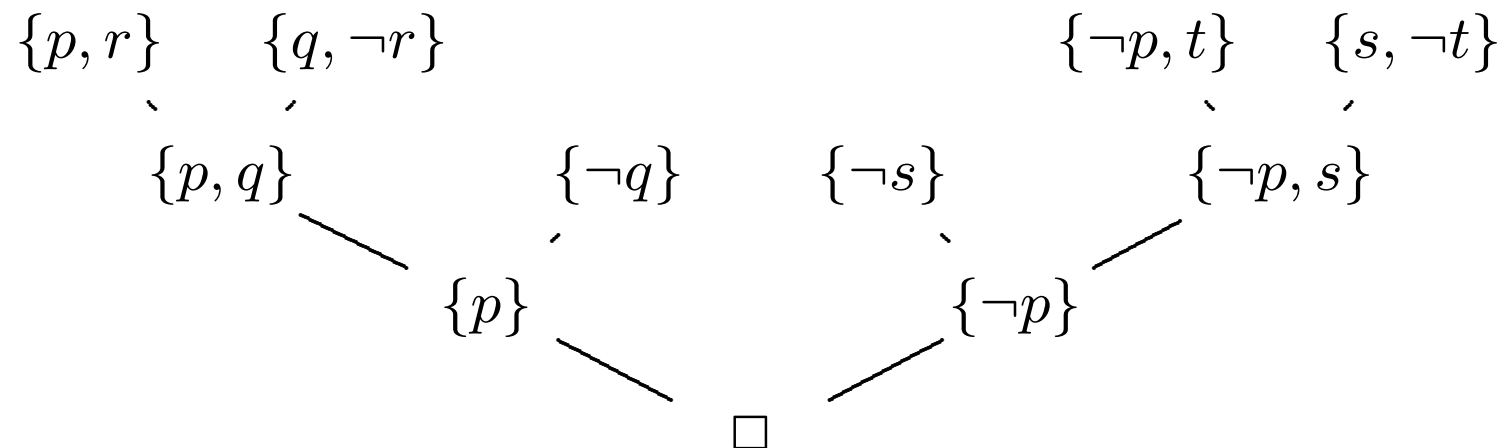
## Resolution in propositional logic – example

- $S \vdash_R \square$  ?

$$S = (p \vee r) \wedge (q \Leftrightarrow r) \wedge \neg q \wedge (t \Leftrightarrow p) \wedge \neg s \wedge (t \Leftrightarrow s)$$

$$S = (p \vee r) \wedge (q \vee \neg r) \wedge \neg q \wedge (\neg p \vee t) \wedge \neg s \wedge (s \vee \neg t)$$

$$S = \{\{p, r\}, \{q, \neg r\}, \{\neg q\}, \{\neg p, t\}, \{\neg s\}, \{s, \neg t\}\}$$



## Refining resolution I

- to narrow search space -  $\text{SAT} = \{ S \mid S \text{ is satisfiable} \}$  is NP-complete
  - to terminate the search along paths that are unpromising
  - to specify the order in which to go down alternative paths

## Refining resolution II

- if there is a literal that is only positive(negative), remove all clauses that contain such a literal
- *T-resolution* : no parent clause is a tautology
- *Semantic resolution*. Let  $\mathcal{I}$  be an interpretation. Semantic resolution with respect to  $\mathcal{I}$  permits applications of the resolution rule only when at least one of their premises has a ground instance which is not satisfied by  $\mathcal{I}$
- *Ordered resolution*. The propositional letters are indexed resolve on the literal with the higher index than any other in the parent clauses

- *Lock resolution.* Each occurrence of a literal has a distinct index the literal resolved on has in each parent the lowest index

## Resolution in predicate logic – introduction

- based on *refutation*
- suitable for automated theorem proving
- formulas in Skolem normal form
  - clause = disjunction of literals (atoms or negation of atoms), represented as a set
  - formula = conjunction of clauses, represented as a set

- Example:

$$\forall x \forall y ((P(x, f(x)) \vee \neg Q(y)) \wedge (\neg R(f(x)) \vee \neg Q(y)))$$

$\rightarrow$

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

# Unification

- a substitution  $\phi$  is a *unifier* for  $S = \{E_1, \dots, E_n\}$  if  $E_1\phi = E_2\phi = \dots = E_n\phi$ , i.e.,  $S\phi$  is singleton.  
 $S$  is said to be *unifiable* if it has a unifier.
- a unifier  $\phi$  for  $S$  is a *most general unifier (mgu)* for  $S$  if, for every unifier  $\psi$  for  $S$ , there is a substitution  $\lambda$  such that  $\phi\lambda = \psi$   
up to renaming variables there is only one result applying an mgu

## Unification – Examples

1. a unifier for  $\{P(x, c), P(b, c)\}$  is  
 $\phi = \{x/b\}$ ; is there any other?
2. a unifier for  $\{P(f(x), y), P(f(a), w)\}$  is  
 $\phi = \{x/a, y/w\}$   
but also  $\psi = \{x/a, y/a, w/a\}$ ,  
 $\sigma = \{x/a, y/b, w/b\}$  etc.
3.  $\{P(x, a), P(b, c)\}, \{P(f(x), z), P(a, w)\},$   
 $\{P(x, w), \neg P(a, w)\},$   
 $\{P(x, y, z), P(a, b)\}, \{R(x), P(x)\}$   
are not unifiable

mgu?

in (2.)  $\phi$  is the mgu:  $\psi = \phi\{w/a\}, \sigma = \phi\{w/b\}$



## Resolution in predicate logic – preliminaries

- variables are local for a clause (pozn.:  $\forall x(A(x) \wedge B(x)) \Leftrightarrow (\forall x A(x) \wedge \forall x B(x)) \Leftrightarrow (\forall x A(x) \wedge \forall y B(y))$ )  
i.e. there is no relation between variables equally named
- standardization of vars = renaming, necessary  
 $\{\{P(x)\}, \{\neg P(f(x))\}\}$  is unsatisfiable. Without renaming a variable no unification can be performed

## Resolvent – Examples

Example 1:  $\{P(x, a)\}, \{\neg P(x, x)\}$

- rename vars:  $\{P(x_1, a)\}$
- $mgu(\{P(x_1, a), P(x, x)\}) = \{x_1/a, x/a\}$
- resolvent  $\square$

Example 2:  $\{P(x, y), \neg R(x)\}, \{\neg P(a, b)\}$

- $mgu(\{P(x, y), P(a, b)\}) = \{x/a, y/b\}$
- apply mgu to  $\{\neg R(x)\}$
- resolvent  $\{\neg R(a)\}$

## Resolution rule in predicate logic

$C_1, C_2$  clauses that have no variables in common in the form

$$C_1 = C'_1 \sqcup \{P(\vec{x}_1), \dots, P(\vec{x}_n)\},$$
$$C_2 = C'_2 \sqcup \{\neg P(\vec{y}_1), \dots, \neg P(\vec{y}_m)\}$$

respectively. If  $\phi$  is an mgu for

$\{P(\vec{x}_1), \dots, P(\vec{x}_n), P(\vec{y}_1), \dots, P(\vec{y}_m)\}$ ,  
then  $C'_1\phi \cup C'_2\phi$  is a *resolvent* of  $C_1$  and  $C_2$   
(also called the *child* of *parents*  $C_1$  and  $C_2$ ).

## Resolution rule in predicate logic II

- *Resolution proofs of  $C$  from  $S$*  is a finite sequence  $C_1, C_2, \dots, C_N = C$  of clauses such that each  $C_i$  is either a member of  $S$  or a resolvent of clauses  $C_j, C_k$  for  $j, k < i$
- *resolution tree proof  $C$  from  $S$*  is a labeled binary tree  
the root is labeled  $C$   
the leaves are labeled with elements of  $S$  and  
if any nonleaf node is labeled with  $C_2$  and its immediate successors are labeled with  $C_0, C_1$  then  $C_2$  is a resolvent  $C_0$  and  $C_1$
- *(resolution) refutation of  $S$*  is a deduction of  $\square$  from  $S$

## Resolution – Examples II

Ex. 3:  $C_1 = \{Q(x), \neg R(y), P(x, y), P(f(z), f(z))\}$  a  
 $C_2 = \{\neg N(u), \neg R(w), \neg P(f(a), f(a)), \neg P(f(w), f(w))\}$

- choose the set of literal  
 $\{P(x, y), P(f(z), f(z)), P(f(a), f(a)), P(f(w), f(w))\}$
- mgu  $\phi = \{x/f(a), y/f(a), z/a, w/a\}$
- $C'_1 = \{Q(x), \neg R(y)\}$ ,  $C'_1\phi = \{Q(f(a)), \neg R(f(a))\}$
- $C'_2 = \{\neg N(u), \neg R(w)\}$ ,  $C'_2\phi = \{\neg N(u), \neg R(a)\}$
- the resolvent  
 $C'_1\phi \cup C'_2\phi = \{Q(f(a)), \neg R(f(a)), \neg N(u), \neg R(a)\}$

## **Resolution in the predicate logic**

- is sound (soundness) and complete
- systematic attempts at generating resolution proofs possible  
but redundant and inefficient: the search space is too huge
- what strategy of generating resolvents to choose?

# Linear resolution

$\{\{P(x, x)\}, \{\neg P(x, y), \neg P(y, z), P(z, x)\}, \{P(a, b)\}, \{\neg P(b, a)\}\}$

$\{\neg P(x, y), \neg P(y, z), P(z, x)\} \quad \{P(a, b)\}$

$x/a, y/b \quad \bigg| \quad \bigg/ \quad x/a, y/b$

$\{\neg P(b, z), P(z, a)\} \quad \{\neg P(b, a)\}$

$z/b \quad \bigg| \quad \bigg/ \quad z/b$

$\{\neg P(b, b)\} \quad \{P(x, x)\}$

$x/b \quad \bigg| \quad \bigg/ \quad x/b$

□

sound and complete

# LI-resolution

linear input resolution

$$\{\{P(x, x)\}, \{\neg P(x, y), \neg P(y, z), P(z, x)\}, \{P(a, b)\}, \{\neg P(b, a)\}\}$$

$$\{\neg P(b, a)\} \quad \{\neg P(x, y), \neg P(y, z), P(z, x)\}$$

$$x/a, z/b \quad \left| \quad \begin{array}{c} \diagup \\ x/a, z/b \end{array} \right.$$

$$\{\neg P(a, y), \neg P(y, b)\} \quad \{P(a, b)\}$$

$$y/b \quad \left| \quad \begin{array}{c} \diagup \\ y/b \end{array} \right.$$

$$\{\neg P(b, b)\} \quad \{P(x, x)\}$$

$$x/b \quad \left| \quad \begin{array}{c} \diagup \\ x/b \end{array} \right.$$

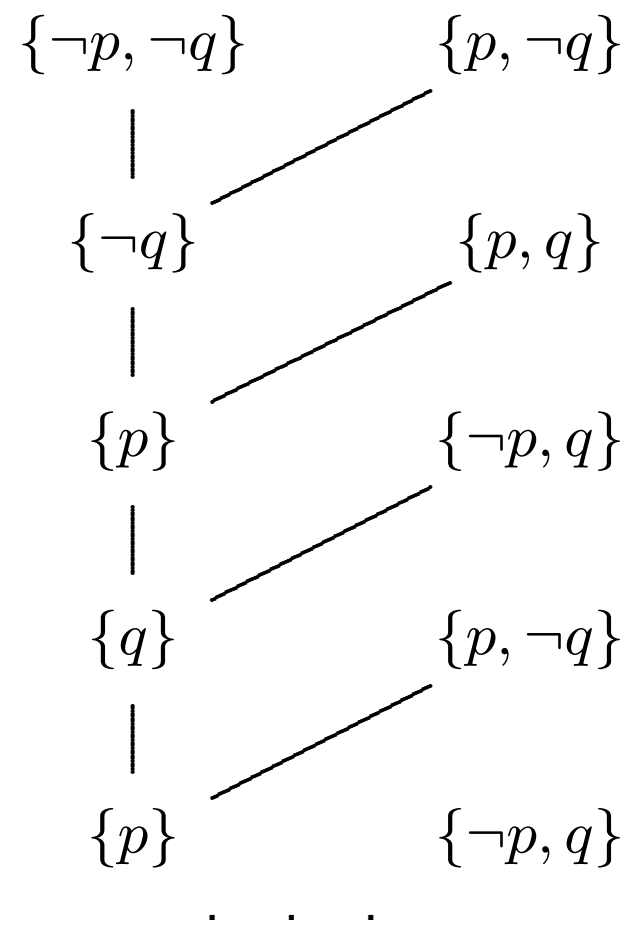
□



## LI-resolution II

sound but not complete in general

$$\text{Ex.: : } S = \{\{p, q\}, \{p, \neg q\}, \{\neg p, q\}, \{\neg p, \neg q\}\}$$



LI-resolution is complete for Horn clauses

## Horn clause

- max. one positive literal

which of  $\{\{p, q\}, \{p, \neg q\}, \{\neg p, q\}, \{\neg p, \neg q\}, \{p\}\}$  are Horn clauses?

- an alternative notation

$\{p \leftarrow q\}, \{p \rightarrow q\}, \{true \rightarrow p\}$

- the Prolog notation

● rule             $p \text{ :- } q.$

fact             $p.$

goal             $?- p, q.$

## LD-resolution

- from LI-resolution to an ordered resolution
- works with an *ordered clauses*;  $[P(x), \neg R(x, f(y)), \neg Q(a)]$

If  $G = [\neg A_1, \neg A_2, \dots, \neg A_n]$  and

$H = [B_0, \neg B_1, \neg B_2, \dots, \neg B_m]$  are ordered clauses and  $\phi$  an mgu for  $B_0$  and  $A_i$ ),

then the (*ordered*) *resolvent* of  $G$  a  $H$  is the ordered clause

$[\neg A_1\phi, \neg A_2\phi, \dots, \neg A_{i-1}\phi, \neg B_1\phi, \neg B_2\phi, \dots, \neg B_m\phi, \neg A_{i+1}\phi, \dots, \neg A_n\phi]$

LD – Linear Definite

# LD-resolution

$\{[P(x, x)], [P(z, x), \neg P(x, y), \neg P(y, z)], [P(a, b)], [\neg P(b, a)]\}$

$[\neg P(b, a)] \quad [P(z, x), \neg P(x, y), \neg P(y, z)]$

$x/a, z/b$  |  $x/a, z/b$

$[\neg P(a, y), \neg P(y, b)] \quad [P(a, b)]$

$y/a$  |  $y/a$

$[\neg P(a, a)] \quad [P(x, x)]$

$x/a$  |  $x/a$

□

LD-resolution is sound and complete for Horn clauses.

## SLD-resolution

- LD-resolution with a selection rule
- A *selection rule*  $R$  is a function that chooses a literal from every nonempty ordered clause  $C$ .
- If no  $R$  is mentioned we assume that the standard one of choosing the leftmost literal is intended.
- Example:  $G = [\neg A_1, \neg A_2, \dots, \neg A_n]$ ,  
 $H = [B_0, \neg B_1, \neg B_2, \dots, \neg B_m]$ ,  
The resolvent of  $G$  and  $H$  for  $\phi = mgu(B_0, A_1)$  is  
 $[\neg B_1\phi, \neg B_2\phi, \dots, \neg B_m\phi, \neg A_2\phi, \dots, \neg A_n\phi]$

SLD-resolution is sound and complete for Horn clauses

# SLD-resolution

selection rule = the leftmost literal

$[\neg P(b, a)]$        $[P(z, x), \neg P(x, y), \neg P(y, z)]$

$x/a, z/b$        $x/a, z/b$

$[\neg P(a, y), \neg P(y, b)]$        $[P(a, b)]$

$y/b$        $y/b$

$[\neg P(b, b)]$        $[P(x, x)]$

$x/b$        $x/b$

□



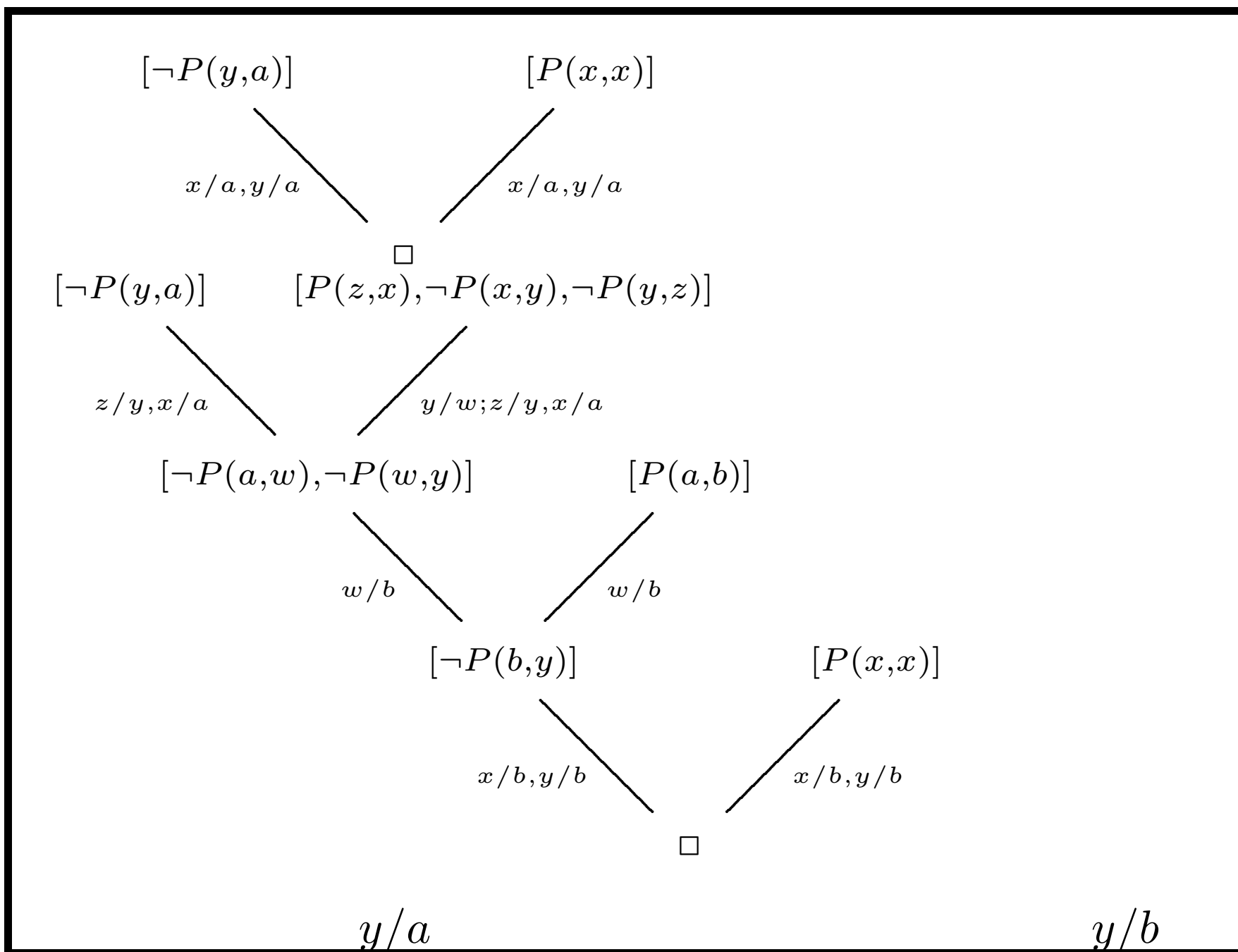
## Example

For

$P = \{[P(a, b)], [P(x, x)], [P(z, x), \neg P(x, y), \neg P(y, z)]\},$

find all solutions (i.e. substitutions of variables) of the goal

$[\neg P(y, a)]$



# SLD-trees

all SLD-derivations for a given goal  $G$  and the program  $P$

- |   |                            |                          |
|---|----------------------------|--------------------------|
| 1. $[P(x,y), \neg Q(x,z), \neg R(z,y)]$ | 5. $[Q(x,a), \neg R(a,x)]$ | 9. $[S(x), \neg T(x,x)]$ |
| 2. $[P(x,x), \neg S(x)]$                | 6. $[R(b,a)]$              | 10. $[T(a,b)]$           |
| 3. $[Q(x,b)]$                           | 7. $[S(x), \neg T(x,a)]$   | 11. $[T(b,a)]$           |
| 4. $[Q(b,a)]$                           | 8. $[S(x), \neg T(x,b)]$   | cíl: $[\neg P(x,x)]$     |

