Inductive Logic Programming. Part 2

Based partially on Luc De Raedt's slides http://www.cs.kuleuven.be/~lucdr/lrl.html

Specialisation and generalisation

A formula G is a **specialisation** of a formula F
iff F entails from G

$G \models F$

= each model of G is also a model of F.

Specialisation operator
assign a formula a set of all its specialisations

Generalisation = the other direction

$G \models F$

F follows *deductively* from G

G follows *inductively* from F

therefore induction is the *inverse* of deduction

this is an operational point of view because there are many deductive operators $\vdash$ that implement $\models$

take any deductive operator and invert it and one obtains an inductive operator

Resolution

father(X,Y) :- male(X) male(adam)

father(adam,kain)

Inverse resolution

Example: Learn a relation father/2 given domain knowledge parent/2 and male/2:
male(adam). male(kain). male(abdullah). male(muhammad). male(moses).
parent(adam,kain). parent(eve,kain). parent(abdullah,muhammad), and
an example father(adam,kain).

Inverse resolution

Example: Learn a relation father/2 given domain knowledge parent/2 and male/2:
male(adam). male(kain). male(abdullah). male(muhammad). male(moses).
parent(adam,kain). parent(eve,kain). parent(abdullah,muhammad), and
an example father(adam,kain).

father(adam,kain)

Inverse resolution

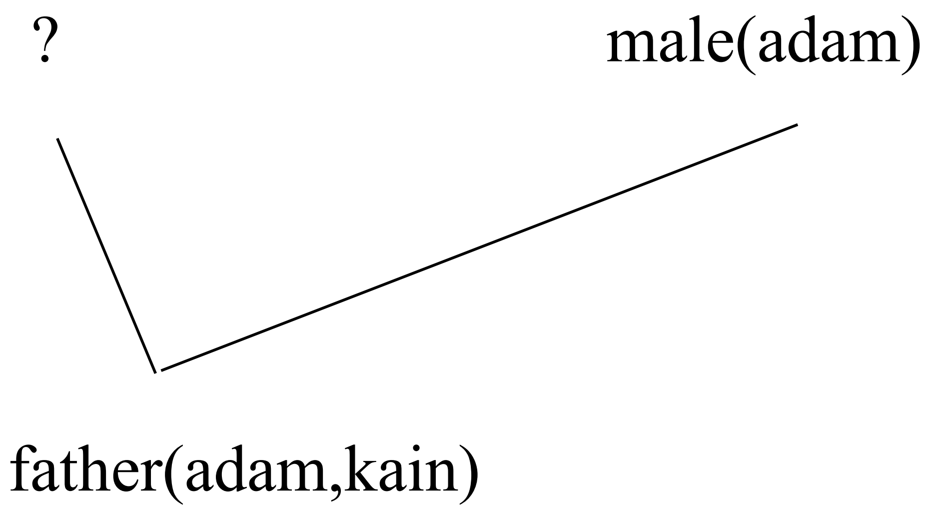
Example: Learn a relation father/2 given domain knowledge parent/2 and male/2:
male(adam). male(kain). male(abdullah). male(muhammad). male(moses).
parent(adam,kain). parent(eve,kain). parent(abdullah,muhammad), and
an example father(adam,kain)

male(adam)

father(adam,kain)

Inverse resolution

Example: Learn a relation father/2 given domain knowledge parent/2 and male/2:
male(adam). male(kain). male(abdullah). male(muhammad). male(moses).
parent(adam,kain). parent(eve,kain). parent(abdullah,muhammad), and
an example father(adam,kain)



Inverse resolution

father(X,Y) :- male(X)   male(adam)

father(adam,kain)

Inverse resolution

? parent(adam, kain)

father(X,Y) :- male(X) male(adam)

father(adam,kain)

Inverse resolution

father(X,Y) :- male(X),parent(X,Y)   parent(adam, kain)

father(X,Y) :- male(X)   male(adam)

father(adam,kain)

Inverse resolution

Given C_1 which is of the form A∨B, and resolvent which is of the form B∨C, the aim is to find C_2.

In propositional logic:

1. Find a literal L that appears in C_1 but not in the resolvent.
2. Then $C2$ is given by either

   (Resolvent - (Resolvent ∩ C_1)) ∪ {¬L}

   or by

   (Resolvent - (C_1 - {L})) ∪ {¬L}

Inverse resolution

father(X,Y) :- male(X)   male(adam)

father(adam,kain)

In predicate logic:

1. Find a literals L_1 in C_1 that is not in the resolvent.
   Then in **C_2** there must be L_2 that $L_1\,\Theta = L_2\Theta$.
2. Assume $\Theta=\Theta_1\Theta_2$ such that $L_1\Theta_1=L_2\Theta_2$ .Then $L_2 = \neg L_1\Theta_1\Theta_2^{-1}$
3. Then C_2 = (Resolvent - (C_1 - $\{L_1\}\Theta_1$)) $\Theta_2^{-1} \cup \neg L_1\Theta_1\Theta_2^{-1}$
4. C_1 is ground $\Rightarrow \Theta_1=\{\}$
   C_2 = (Resolvent - (C_1 - $\{L_1\}$)) $\Theta_2^{-1} \cup \neg L_1\Theta_2^{-1}$

Inverse resolution

Main drawback

nondeterminism

father(X,Y) :- male(X)

father(X,kain) :- male(X)

father(adam,kain) :- male(adam)

male(adam)

father(adam,kain)

Subsumption and Θ-subsumption

Clause G subsumes clause F if and only G $\models$ F or, equivalently G $\subseteq$ F

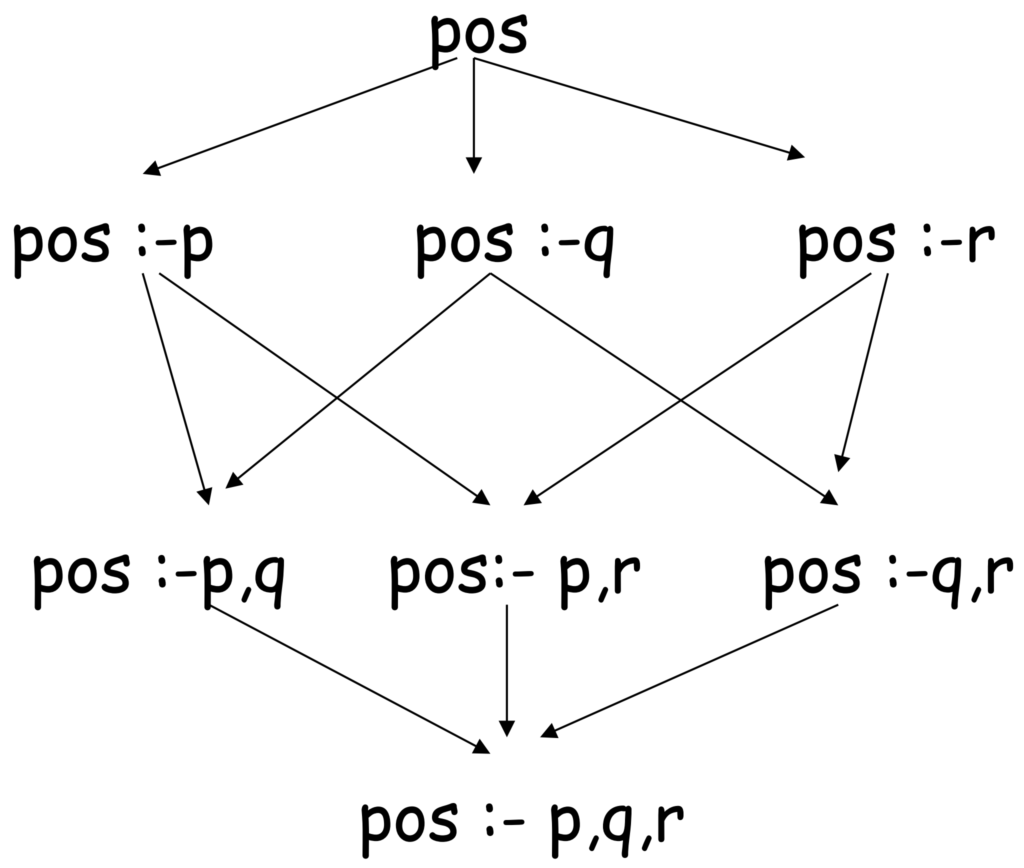
Example - propositional logic

pos :- p,q,r $\models$ pos :- p,q,r,s,t

because

$\{pos, \neg p, \neg q, \neg r\} \subseteq \{pos, \neg p, \neg q, \neg r, \neg s, \neg t\}$

Subsumption in propositional logic



Subsumption in propositional logic

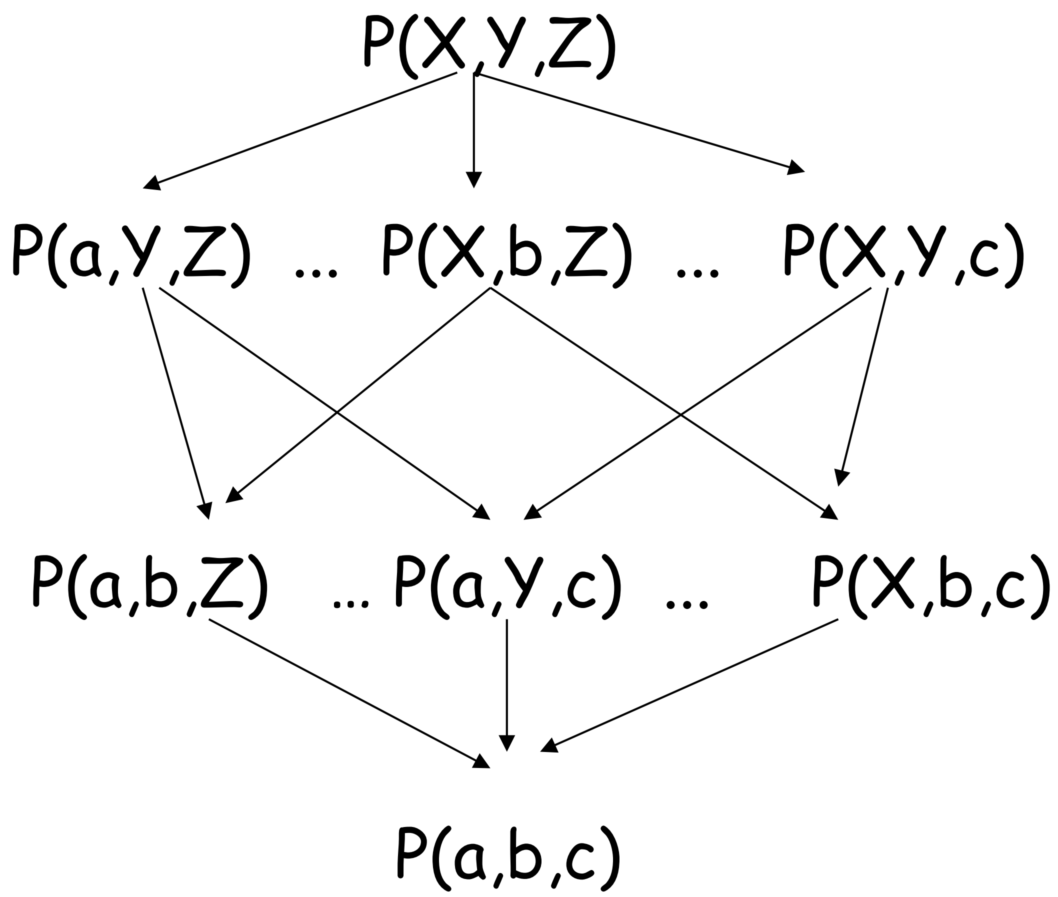
- Perfect structure
- Complete lattice
  - any two clauses have unique
    - least upper bound (least general generalization)
    - greatest lower bound
- No syntactic variants
- Easy specialization, generalization

Subsumption in predicate logic

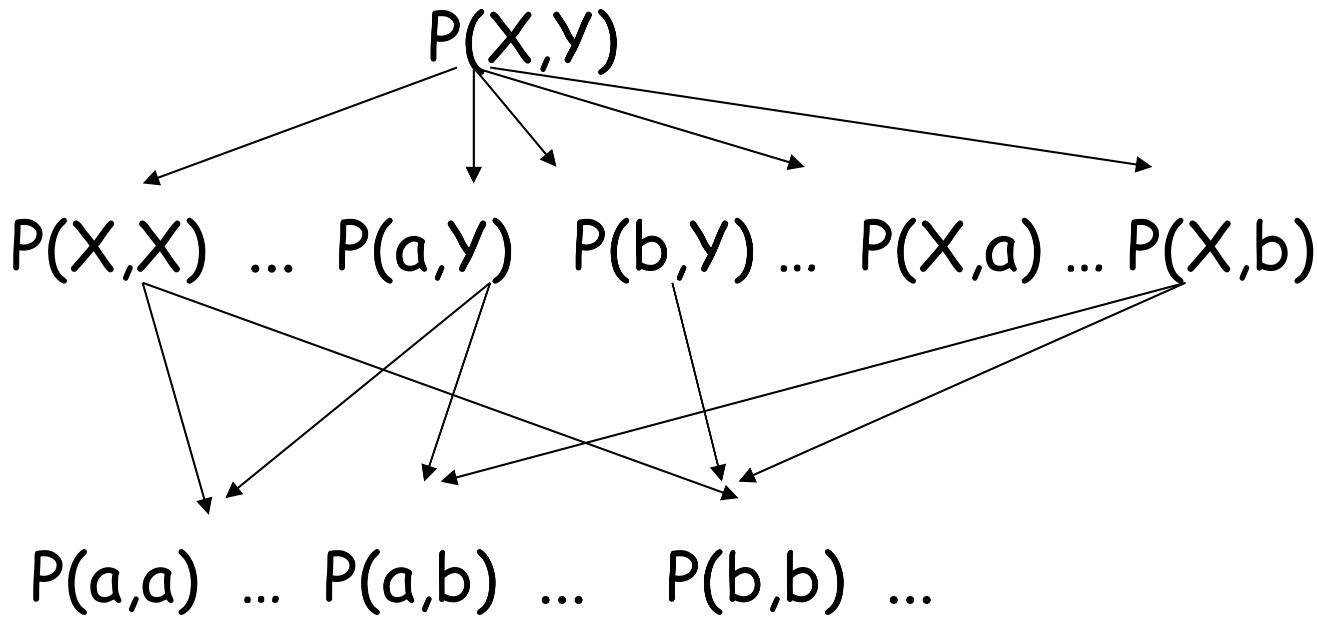
Subsumption in logical atoms

- g subsumes s if and only if there is a substiution θ such that $g\theta = s$
- e.g. p(X,Y,X) subsumes p(a,Y,a)
- e.g. p(f(X),Y) subsumes p(f(a),Y)

Subsumption in simple logical atoms



Subsumption in simple logical atoms



Subsumption in logical atoms

P(X)

P(f(Y)) … P(g(Y)) … P(h(Y,Z)) …

P(f(f(W)) P(f(g(W)))

P(f(f(f(U)))) …

P(f(f(f(f(V)))) …

Subsumption in logical atoms

G subsumes F iff there is a substitution θ such that $G\theta = F$

- Still nice properties and complete lattice up to variable renaming
  - p(X,a) and p(U,a)
  - greatest lower bound = unification
  - unification p(X,a) and p(b,U) gives p(b,a)

  - least upper bound = anti-unification = lgg
  - lgg p(X,a,b) and p(c,a,d) = p(X,a,Y)
  - lgg p(X,f(X,c)) and p(a,f(a,Y)) gives p(U,f(U,T))

Ideal Specialization Operator

- Ideal Specialization operator :
  - apply a substitution { X / Y } where X,Y already appear in atom
  - apply a substitution { X / f(Y1, … , Yn)} where Yi new variables
  - apply a substitution {X / c } where c is a constant

- Ideal Generalization operator :
  - apply an inverse substitution
    - Inverse substitution substitutes terms at specified places by variables
    - Invert one of the specialization steps above
      - Replace some (but not all) occurences of a variable X by a different variable Y
      - Replace all terms f(Y1,...,Yn) where Yi are distinct by a new variable X
      - Replace some occurences of a constant by a new variable

Ideal Specialization Operator

Properties

Ideal specialisation operator must be

- locally complete
- globally complete
- proper

Ideal Specialization Operator

Let A be an atom. Then

$$\rho_{s,a,i}(A) = \{A\theta \mid \theta \text{ is an elementary substitution}\} \tag{5.4}$$

where an elementary substitution θ is of the form

$$\theta = \begin{cases} \{X/f(X_1, ..., X_n)\} & \text{with } f \text{ a functor of arity } n \text{ and} \\ & \quad \text{the } X_i \text{ are variables not occurring in } A \\ \{X/c\} & \text{with } c \text{ a constant} \\ \{X/Y\} & \text{with } X \text{ and } Y \text{ are variables occurring in } A \end{cases} \tag{5.5}$$

It is relatively easy to see that $\rho_{s,a,i}$ is an ideal operator for atoms.

Optimal Specialization Operator

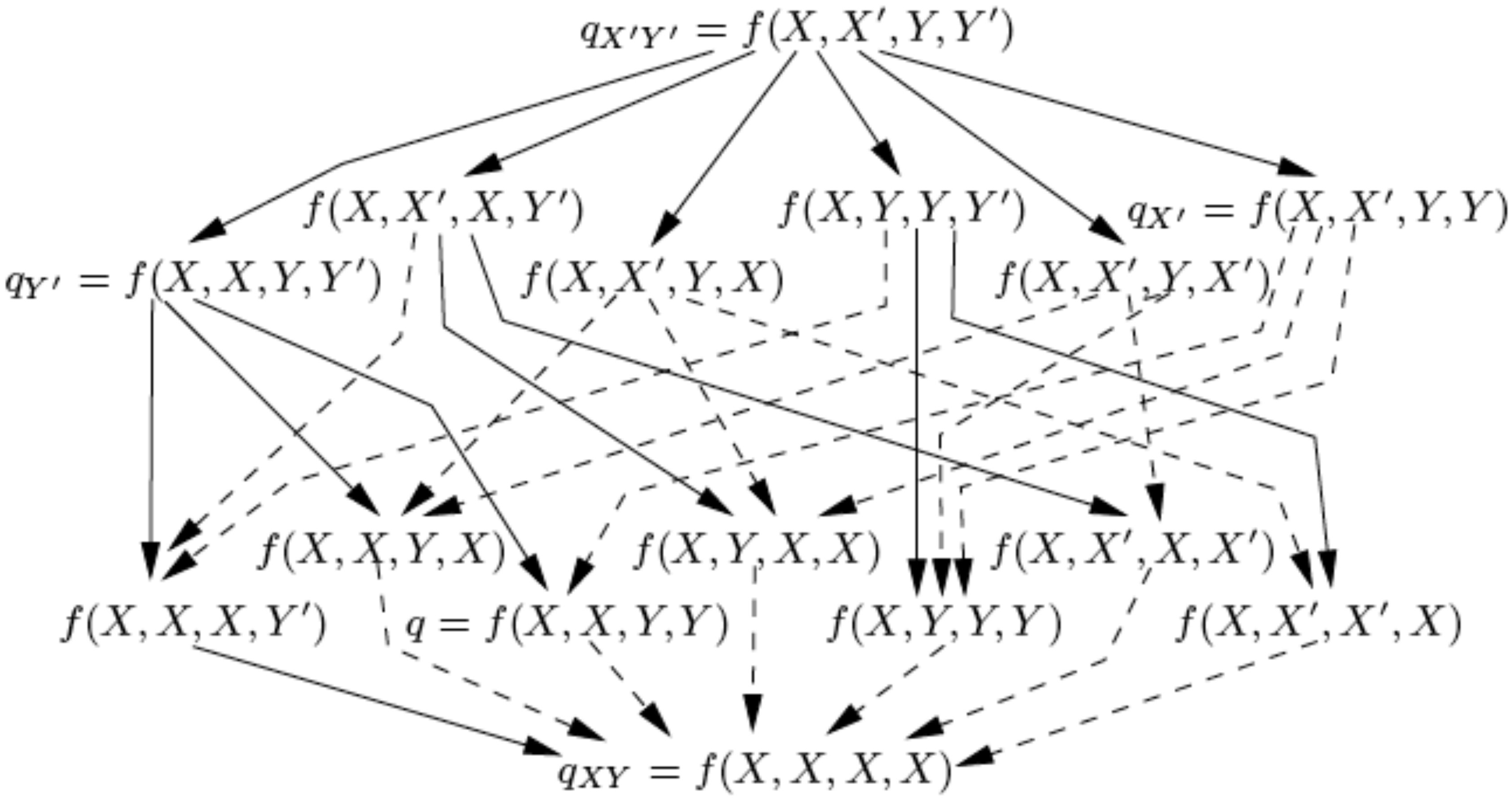


Fig. 5.6. Example of duplicate avoidance for Unification

Optimal Specialization Operator

Let A be an atom. Then

$$\rho_{s,a,o}(A) = \{A\theta \mid \theta \text{ is an optimal elementary substitution}\} \tag{5.6}$$

where an elementary substitution θ is of the form θ is an *optimal elementary substitution* for an atom A iff it is of the form

$$\theta = \begin{cases} \{X/f(X_1, ..., X_n)\} & \text{with } f \text{ a functor of arity } n \text{ and} \\ & \quad \text{the } X_i \text{ variables not occurring in } A \\ \{X/c\} & \text{with } c \text{ a constant} \\ \{X/Y\} & \text{where } X \text{ and } Y \text{ are variables occurring in } A \\ & \quad X \text{ occurs once, and all variables to the right of} \\ & \quad X \text{ occur only once in } A \end{cases} \tag{5.7}$$

Theta-subsumption (Plotkin 70)

- Most important framework for inductive logic programming. Used by all major ILP systems.
- F and G are single clauses
- Combines propositional subsumption and subsumption on logical atoms

- c1 theta-subsumes c2 if and only if there is a substitution θ such that $c1\ \theta \subseteq c2$

- c1 : father(X,Y) :- parent(X,Y),male(X)
- c2 : father(adam,kain) :- parent(adam,kain), parent(adam,an), male(adam), female(an)
- $\theta = \{ X / adam, Y / kain \}$

Example

- d1 : p(X,Y) :- q(X,Y), q(Y,X)
- d2 : p(Z,Z) :- q(Z,Z)
- d3 : p(a,a) :- q(a,a)
- theta(1,2) : {X / Z, Y /Z}
- theta(2,3) : {Z/a}
- d1 is a generalization of d3
- Mapping several literals onto one leads (sometimes) to combinatorial problems

Properties

- Soundness : if c1 theta-subsumes c2 then

c1 |= c2

- Incompleteness (but only for self-recursive clauses) wrt logical entailment
  - c1 : p(f(X)) :- p(X)
  - c2 : p(f(f(Y))) :- p(Y)
- Decidable (but NP-complete)
- transitive and reflexive but not anti-symmetric

Specialisation operations

binding of two distinct variables

path(X,Y) . . . *There is a path between nodes X and Y in a graph*

edge(X,Y). . . *There is an edge between X and Y*

spec(path(X, Y)) = path(X, X)

adding a most general atom into a clause body

arguments are distinct and so far unused variables

spec(path(X,Y)) = (path(X,Y) :- edge(U,V))

= a minimal set of specialisation operations for logic programs without function symbols:

Specialisation operations

Logic programs with functions:

A minimal set extended with

Substitution a variable with a most general term

arguments are distinct and so far unused variables

spec(number(X)) = number(0)
spec(number(X)) = number(s(Y)) .

Specialisation and generalisation

Domain-dependent operations - examples

triangle $\leq$ n-angle $\leq$ plannar object

town $\leq$ district $\leq$ region $\leq$ country $\leq$ continent

$[0,1) \leq [0,11) \leq [0,111) \leq [0,\mathrm{inf})$