Refinement operators [Procedural counterpart of $\geq$]

Recall

LUB

lgg [rlgg | lgi / Rgri

e1 e2

Gels well with Occams razor

Downward covers (used more often) why? Can generalize well

refinement: traverse along cover links

About refining clauses

p(e) = 0

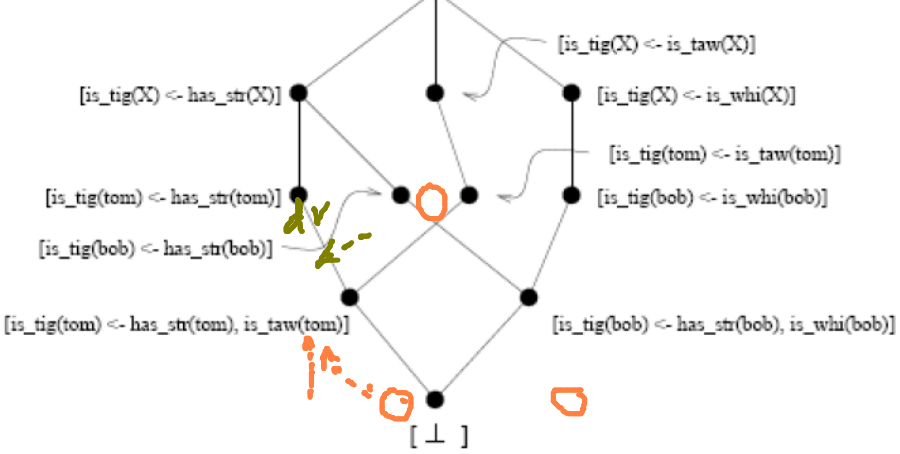
Generality of clause: $g(H) = \sum_{x \in X, H \models x, x \text{ is clause}} p(x)$

GLB: Very nondeterministic

t1 t2

Upward covers: Too many steps to generalize

e



Note: Taking 2 examples at a time can enable large steps: It may not be robust noise Lub(rlgg) takes the examples too serious

lub
glb

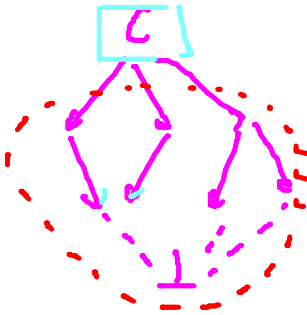
ur

Recall.

Refinement operators (also how to incorporate background knowledge)

Example downward refinement: $P(x) \rightarrow P(f(x))$

Top down search algos for ILP use downward refinement



$\rho(C) \subseteq$

- ρ is a *downward refinement operator* if $\forall C \in S : \rho(C) \subseteq \{D | D \in S \text{ and } C \succeq D\}$
- δ is an *upward refinement operator* if $\forall C \in S : \delta(C) \subseteq \{D | D \in S \text{ and } D \succeq C\}$

$$l = \begin{cases} V = W & \text{where } V, W \text{ occur in } C \\ V = f(X_1, X_2, \ldots, X_{n_f}) & \text{where } V \text{ occurs in } C \text{ and } f/n_f \in \mathcal{L} \text{ and the } X_i \text{ are distinct} \\ q(X_1, X_2, \ldots, X_{n_q}) & \text{where } q/n_q \in \mathcal{L} \text{ and the } X_i \text{ occur in } C \end{cases}$$

} L is language

Eg: of $\mathcal{S}(C)$: Assume $=/2$ is an equality relation

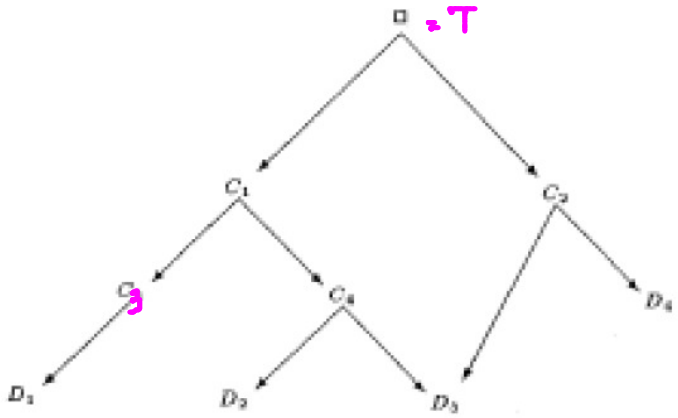
Assume, $V, W, f(X_1 \ldots X_{n_f})$, are terms in G

$q(X_1 \ldots X_{n_q})$ is a predicate

→ Note: $D \subseteq C \cup \{\ell\}$ is also another $\mathcal{S}(C)$

→ Thus many $\mathcal{S}(C)$ are possible. Q: Which one to choose

↳ For eg, this $\mathcal{S}(C)$ does not guarantee reachability



[Q: can $\rho^*(\square)$ help you reach Σ]

So many different $\rho(C)$ are possible. Which to choose?

Say :- $\Sigma = \{D_2, D_3, D_4\}$ is correct theory [could be that D_2, D_3 & D_4 cover ε^+ & D_1 covers ε^- pruned by eval]

$\rho(\square) = \{C_1, C_2\}$, $\rho(C_1) = \{C_3, C_4\}$, $\rho(C_2) = \{D_3, D_4\}$

$\rho(C_3) = \{D_1\}$, $\rho(C_4) = \{D_2, D_3\}$

ρ : chosen such that theories covering correct examples can be reached (& hopefully as quickly as possible)

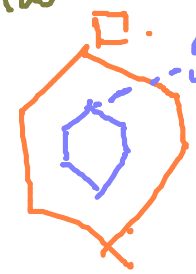
Roughly.. ρ handles recall, "eval" measure handles precision

[... can bias choice from ρ]

Want ρ s.t ρ^*(top clause) $\geq$ correct theory.

↳ If top clause = $T = \square$, then you want every clause to be reachable.

↳ If top clause = "gfather(X,Y) ←", you want every definition of gfather reachable.



Notations

$\rho(C)$

$\rho^n(C)$

$\rho^*(C) = \bigcup_{i=1}^{\infty} \rho^i(C)$

ρ chain:- from C to D is a sequence $C = C_0, C_1 \ldots C_n = D$ & $C_i \in \rho(C_{i-1})$ for $i = 1 \ldots n$

$\Rightarrow$ Gives you a "refinement graph".

Want:- C covering $C' \in$ refinement graph

Some desirable properties of ρ (and dually δ) are that they be:

1. **Locally Finite:** $\forall C \in \mathcal{C}$: $\rho(C)$ is finite and computable. Otherwise ρ will be of limited use in practice.
2. **Complete:** $\forall C \succ D$: $\exists E \in \rho^*(C)$ s.t. $E \sim D$. That is, every specialization should be reachable by a finite number of applications of the operator.
3. **Proper:** $\forall C \in \mathcal{C} : \rho(C) \subseteq \{D | D \in S \text{ and } C \succ D\}$. That is, it is better only to compute proper generalizations of a clause, for otherwise repeated application of the operator might get stuck in a sequence of equivalent clauses (such as the application of inverse reduction in Section 2.3), without ever achieving any real specialization.

E ≥ρ ∧ D ≥ E

If all 3 props, called IDEAL

i.e. if E ~ C then ρ(C) ∋ E

⇓

Your steps are non-trivial

⇒ A ref graph. < finite # edges starting at each node (1)
a path of finite length from C to some E ~ D (2)
no cycles (3)

ideal ref operators exist for atoms

Definition 7 *Let $\mathcal{A}$ be the set of atoms in a language. The* ideal *downward refinement operator $\rho_{\mathcal{A}}$ for $\mathcal{A}$ is defined as follows:*

1. *For every variable z in an atom A and every n-ary function symbol f in the language, let $x_1, \ldots, x_n$ be distinct variables not appearing in A. Let $\rho_{\mathcal{A}}(A)$ contain $A\{z/f(x_1, \ldots, x_n)\}$.*

2. *For every variable z in A and every constant a in the language let $\rho_{\mathcal{A}}(A)$ contain $A\{z/a\}$.*

3. *For every two distinct variables x and z in A, let $\rho_{\mathcal{A}}(A)$ contain $A\{z/x\}$.*

Eg: ① $p(x,y,z)$ → $p(x,y,f(x_1,x_2))$, $p(f(x_1,x_2),y,z)$, $p(x,f(x_1,x_2),z)$ } for every n-ary fn.

② Special case of ① with 0-ary fn = constant

③ $p(x,y,z)$ → $p(x,x,z)$, $p(x,z,z)$, $p(x,y,x)$ } ${}^3C_2$ (think of it as adding a new predicate "$x = z$".

An ideal upward refinement for atoms ($\delta(A)$)

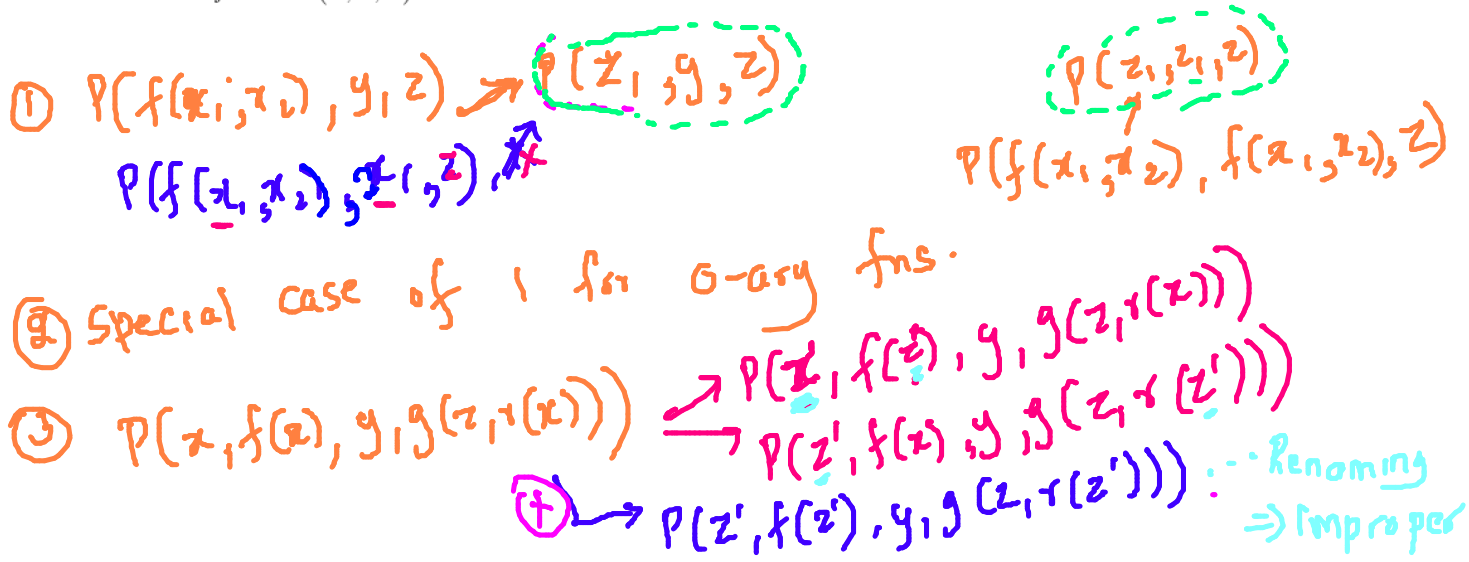
Definition 8 *Let $\mathcal{A}$ be the set of atoms in a language. The upward refinement operator $\delta_{\mathcal{A}}$ for $\mathcal{A}$ is defined as follows:*

1. *For every $t = f(x_1, \ldots, x_n)$ in A, for which $x_1, \ldots, x_n$ are distinct variables and each occurrence of some x_i in A is within an occurrence of t, $\delta_{\mathcal{A}}(A)$ contains an atom obtained by replacing all occurrences of t in A by some new variable z not in A.*

2. *For every constant a in A and every non-empty subset of the set of occurrences of a in A, $\delta_{\mathcal{A}}(A)$ contains an atom obtained by replacing those occurrences of a in A by some new variable z not in A.*

3. *For every variable x in A and every non-empty proper subset of the set of occurrences of x in A, $\delta_{\mathcal{A}}(A)$ contains an atom obtained by replacing those occurrences of x in A by some new variable z not in A. Note that in this last item, we cannot replace all occurrences of x by a new variable z, for then we would get a variant of A. For instance, $P(z, a, z)$ is a variant of $A = P(x, a, x)$.*

Locally finite
Complete
Proper



For general clauses, ideal refinements ops

do not exist

↳ Recall:- $C_n = \{ p(x_i, x_j) \mid i \neq j \ \& \ 1 \leq i, j \leq n \}$ $n \geq 2$

& $C = \{ p(x_1, x_2) \}$

can show: $C_2 > C_3 > C_4 \cdots > C_n > \cdots > C$.

⇒ C does not have an upward cover.

⇒ Completeness not possible. If properness is imposed (Completeness may require redundancy)

C_1
C_2
⋮
C_n
⋮
C

infinite elements just above C

Compromising . . . . . .

① Finiteness = indispensible

② Completeness is valuable

③ Properness is least important

④ Finite & Complete ρ_L is possible :- → by applying elem. subst^s → adding literals (most general)

Definition 9 *Let $\mathcal{C}$ be a clausal language. The locally finite and complete downward refinement operator $\rho_{\mathcal{L}}$ for $< \mathcal{C}, \succeq >$ is defined as follows:*

(a) **Apply a substitution to a clause** C *in one of the following ways:*

i. For every variable z in a clause C and every n-ary function symbol f in the language, let $x_1, \ldots, x_n$ be distinct variables not appearing in C. Let $\rho_{\mathcal{L}}(C)$ contain $C\{z/f(x_1, \ldots, x_n)\}$.

ii. For every variable z in C and every constant a in the language, let $\rho_{\mathcal{L}}(C)$ contain $C\{z/a\}$.

iii. For every two distinct variables x and z in C, let $\rho_{\mathcal{L}}(C)$ contain $C\{z/x\}$.

} Similar to $\rho(A)$

Not proper! check on $C = \{p(x)\}$ & $D = \{p(x), p(y)\}$ but x

(b) **Add a literal to a clause** C**:** *For every n-ary predicate symbol P in the language, let $x_1, \ldots, x_n$ be distinct variables not appearing in C. Then $\rho_{\mathcal{L}}$ contains both $C \vee \{P(x_1, \ldots, x_n)\}$ and $C \vee \{\neg P(x_1, \ldots, x_n)\}$. Note that the literals $P(x_1, \ldots, x_n)$ and $\neg P(x_1, \ldots, x_n)$ that are added to C by the fourth item in the definition are most general with respect to C.*

} additional part . . .

↳ $x_1 \ldots x_n$ do not appear in C

θ: $\{p(x)\} \dashrightarrow \{p(a), p(b)\}$

Definition 10 *Let $\mathcal{C}$ be a clausal language. The locally finite and complete upward refinement operator δ_u for $<\mathcal{C}, \succeq>$ is defined as follows:*

(a) **Apply one of the following inverse substitution operations:**

i. For every $t = f(x_1, \ldots, x_n)$ in C, for which all x_i are distinct variables and each occurrence of x_i in a clause C is within an occurrence of t, $\delta_u(C)$ contains the clause obtained by replacing all occurrences of t in C by some new variable z not previously in C.

ii. For every constant a in C and every non-empty subset of the set of occurrences of a in $\overline{C} = dup(C, a)$, if $\overline{D}$ is the ordered clause obtained by replacing those occurrences of a in $\overline{C}$ by the new variable z, then $\delta_u(C)$ contains D, where D is the set of literals in the ordered clause $\overline{D}$.
$dup(C, t)$ is defined as follows. If $C = \{L_1, \ldots, L_n\}$ is a clause and t a term occurring in C and suppose t occurs k_1 times in L_1, k_2 times in L_2, etc. Then $dup(C, t) = \overline{C}$ is an ordered clause consisting of 2^{k_1} copies of L_1, 2^{k_2} copies of L_2, ... and 2^{k_n} copies of L_n. Note that if some $L \in C$ does not contain the term t, then $\overline{C}$ contains L $2^0 = 1$ times, as it should.

iii. For every variable x in C and every non-empty proper subset of the set of occurrences of x in $\overline{C} = dup(C, x)$, if $\overline{D}$ is the ordered clause obtained by replacing those occurrences of x in $\overline{C}$ by the new variable z, then $\delta_u(C)$ contains D.

(b) **Remove a literal from the body of a clause:** *If $C = D \cup \{L\}$ and L is a most general literal with respect to D, then $\delta_u(C)$ contains D.*

Complete & finite upward refinement.

Compromising requirement of Completeness

MIS, FOIL, CLAUDIEN, LINUS, PROGOL

↳ For reasons of efficiency

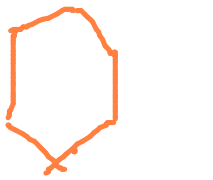
FOIL

$p(x,y) \leftarrow$

① head clause

② considers only addition of literals (Li)

$q(x_1^{new}, x_2^{new} \ldots x_1^{old} \ldots x_2^{old})$ OR $x_1^{old} = x_2^{old}$



Choose one with highest Gain

$C_{i+1} = \text{argmax}_{C \in \rho(C_i)} \text{Gain}(C)$

(No function/constant substs)

Training examples		*Background knowledge*	
$daughter(mary, ann).$	$\oplus$	$parent(ann, mary).$	$female(ann).$
$daughter(eve, tom).$	$\oplus$	$parent(ann, tom).$	$female(mary).$
$daughter(tom, ann).$	$\ominus$	$parent(tom, eve).$	$female(eve).$
$daughter(eve, ann).$	$\ominus$	$parent(tom, ian).$	

$\mathcal{E}_i$ = examples covered by C_i

Current clause c_1 :	$daughter(X, Y) \leftarrow$			
$\mathcal{E}_1$	$(mary, ann)$	$\oplus$	$n_1^{\oplus} = 2$	$I(c_1) = 1.00$
	(eve, tom)	$\oplus$	$n_1^{\ominus} = 2$	
	(tom, ann)	$\ominus$	$L_1 = female(X)$	
	(eve, ann)	$\ominus$	$Gain(L_1) = 0.84$ $\quad n_1^{\oplus\oplus} = 2$	
Current clause c_2 :	$daughter(X, Y) \leftarrow female(X), L_1$			
$\mathcal{E}_2$	$(mary, ann)$	$\oplus$	$n_2^{\oplus} = 2$	$I(c_2) = 0.58$
	(eve, tom)	$\oplus$	$n_2^{\ominus} = 1$	
	(eve, ann)	$\ominus$	$L_2 = parent(Y, X)$	
			$Gain(L_2) = 1.16$ $\quad n_2^{\oplus\oplus} = 2$	
Current clause c_3 :	$daughter(X, Y) \leftarrow female(X), parent(Y, X)$			
$\mathcal{E}_3$	$(mary, ann)$	$\oplus$	$n_3^{\oplus} = 2$	$I(c_3) = 0.00$
	(eve, tom)	$\oplus$	$n_3^{\ominus} = 0$	

$S(C_1)$

$S(C_2)$

$\mathcal{E}_2$ = set of egs in $\mathcal{E}_1$ which satisfy L_1

$\mathcal{E}_{i+1} = \mathcal{E}_i \bowtie L_i$

$I(C_i) = -\log_2\left(\frac{n_i^{\oplus}}{n_i}\right)$

info needed for +ve signal using C_i

$Gain(L_i) = \left(I(C_i) - I(C_{i+1})\right) * n_i^{\oplus\oplus}$

Reflects +vity of L_i

$n_i^{\oplus\oplus}$ = # of +ve tuples in $\mathcal{E}_i$ represented in $\mathcal{E}_{i+1}$

What if L_i introduces new vars

Current clause c_1 : $daughter(X,Y) \leftarrow$				
$\mathcal{E}_1$	$(mary, ann)$	$\oplus$		$n_1^{\oplus} = 2$
	(eve, tom)	$\oplus$		$n_1^{\ominus} = 2$
	(tom, ann)	$\ominus$	$L_1 = parent(Y,Z)$	
	(eve, ann)	$\ominus$	Gain : η_1	$n_1^{\oplus\oplus} = 2$
Current clause c_2 : $daughter(X,Y) \leftarrow parent(Y,Z)$				
$\mathcal{E}_2$	$(mary, ann, mary)$	$\oplus$		$n_2^{\oplus} = 4$
	$(mary, ann, tom)$	$\oplus$		
	(eve, tom, eve)	$\oplus$		
	(eve, tom, ian)	$\oplus$		
	$(tom, ann, mary)$	$\ominus$		$n_2^{\ominus} = 4$
	(tom, ann, tom)	$\ominus$		
	$(eve, ann, mary)$	$\ominus$		
	(eve, ann, tom)	$\ominus$		

Jon story continues

$\mathcal{E}_{i+1} = \mathcal{E}_i \bowtie L_i$

You do not seem to get rid of any $\ominus$ vcs in $\mathcal{E}_2$

Pruning criterion: Best gain so far $> n_{i,k}^{\oplus\oplus} * I(C_i)$

(L_1, L_{k-1})

L_1 L_2 L_k

By introducing new vars in L_i

'Overall FOIL'

1) $\mathcal{E}_{curr} = \mathcal{E}$
2) $\mathcal{H} = \phi$
3) repeat set covering {
4) ---- Initialize $C := T$ ←
5) --- Find C_{best}
6) -- $C = C_{best}$
7) - Post process C by removing irrelevant lits to get C'

Mainly determinate literals.

8) ..... $\mathcal{H}' = \mathcal{H} \cup \{C'\}$
9) .... $\mathcal{E}'_{curr} = \mathcal{E}_{curr} - covers(B, \{C'\}, \mathcal{E}_{curr})$
10) .... $\mathcal{E}_{curr} = \mathcal{E}'_{curr}$, $\mathcal{H} = \mathcal{H}'$

For noise handling

11) until $\mathcal{E}_{curr} = \phi$ or encoding length restriction violated

Encoding length for c_i w.r.t $\mathcal{E}_{curr}$

$$EL(c_i, \mathcal{E}_{curr}) = \log_2(n_{curr}) + \log_2\left(\binom{n_{curr}}{n^{\oplus}(c_i)}\right)$$

# of set choices possible

# of bits needed to specify tve egs covered by a c_i

(related MDL)

Requirement : $EL(c_i, \mathcal{E}_{curr}) \leq D$ (so that c_i does not become toooo long)

(FOIL 2.0) **Determinate literals:**

A lit is det, if each of its vars that do not appear in preceding lits has only one possible binding, given the bindings of its vars that appear in preceding lits

Eg:

Training examples	
$grandmother(ann, bob).$	$\oplus$
$grandmother(ann, sue).$	$\oplus$
$grandmother(bob, sue).$	$\ominus$
$grandmother(tom, bob).$	$\ominus$

Background knowledge		
$father(zak, tom).$	$father(pat, ann).$	$father(zak, jim).$
$mother(ann, tom).$	$mother(liz, ann).$	$mother(ann, jim).$
$father(tom, sue).$	$father(tom, bob).$	$father(jim, dave).$
$mother(eve, sue).$	$mother(eve, bob).$	$mother(jean, dave).$

$$grandmother(X, Y) \leftarrow father(Z, Y), mother(X, Z).$$
$$grandmother(X, Y) \leftarrow mother(U, Y), mother(X, U).$$

Det lits help, broaden horizon

Both clauses are determinate ⇐

Z in father(Z,Y) has only one value given a value for Y

"Since every person has only one father"

If L_i is det $\Rightarrow \eta^{\oplus}_{i+1} = \eta^{\oplus}_i$, $\eta^{\ominus}_{i+1} = \eta^{\ominus}_i \Rightarrow$ Gain $(L_i) = 0$

While traditional FOIL does greedy hill climbing, it avoids det lits (gain being 0) & ∴ may miss good theories

Q: When should you consider adding a det lit?

Ans:- If no lit has gain close to max, all determinate lits are added.

In the gain ≅ max, it means ∃ a literal which gets rid of all -ve examples

Problem:- Many determinate lits added, may never be expanded on

$$grandmother(X,Y) \leftarrow father(Z,Y), mother(X,Z).$$
$$grandmother(X,Y) \leftarrow mother(U,Y), mother(X,U).$$

mother(U,Y) → removed thru past processing....

If Gain (C) ≡ Likelihood ≡ nfoil.

↳ Highest SVM F1 ≡ kfoil