Lenguajes lógicos: cláusulas de Horn

- Lenguaje subyacente: *FOL* ("first order logic")

  Atomos: $p(t_1,\ldots,t_n)$

  - **p** símbolo de predicado y $\mathbf{t_i}$ término (i = 1,...,n)
  - **t** término →
    - símbolo de variable (X, Y, ..)
    - símbolo de constante (a, b, c,..)
    - $\mathbf{f(s_1,\ldots,s_m)}$, con **f** símbolo de función y $\mathbf{s_j}$ término (j = 1,...,m)

- Cláusulas de programa:
  - hechos: A. (A átomo)
  - reglas: A :- $A_1, \ldots, A_n$. (n>0, y $A, A_1, \ldots, A_n$ átomos)

- Programa lógico = fórmula en *FOL*

  $P = \{c_1, \ldots, c_k\}$ → $rep(P) = rep(c_1) \wedge \ldots \wedge rep(c_k)$

  donde:

  $c = A.$ → $rep(c) = A^{\forall}$

  $c = A$:- $A_1, \ldots, A_n.$ → $rep(c) = (A_1 \wedge \ldots \wedge A_n \rightarrow A)^{\forall}$

- "Goals" o preguntas al programa:

  $G = B_1, \ldots, B_m.$ → $rep(G) = (B_1 \wedge \ldots \wedge B_m)^{\exists}$

  ?G es considerar su negación $\neg\, rep(G) = (\neg B_1 \vee \ldots \vee \neg B_m)^{\forall}$

  (Bj átomo, j=1,...,m. Si m=0, la cláusula vacía se denota por □)

Nota: $\mathbf{f}^{\forall}$ denota la clausura universal de la fórmula **f** y

$\mathbf{f}^{\exists}$ denota la clausura existencial de la fórmula **f**

Ejemplo: Programa Lógico "HOSPITAL"

Hechos:

1. padece(jon, gripe).
2. padece(jon, hepatitis).
3. padece(ana, gripe).
4. padece(carlos, alergia).
5. es-síntoma(fiebre, gripe).
6. es-síntoma(cansancio, gripe).
7. es-síntoma(estornudos, alergia).
8. suprime(paracetamol, fiebre).
9. suprime(antihistamínico, estornudos).

Reglas:

10. debe-tomar(Per, Far) :- padece(Per, Enf), alivia(Far, Enf).
11. alivia(Far, Enf) :- es-síntoma(Sin, Enf), suprime(Far, Sin).

Ejemplo: Preguntas al programa "HOSPITAL"

Preguntas posibles:

? padece(carlos, gripe).
? padece(jon, Z).
? alivia(paracetamol, gripe).
? alivia(X, gripe).
? debe-tomar(Y, antihistamínico).
? alivia(X, Y).
? suprime(X, fiebre), suprime(X, estornudos).
? ....

- Proceso de cómputo: SLD-resolución

  combina dos mecanismos: reemplazamiento y unificación

- Regla de resolución:

literal seleccionado (Func. Selec.)

$?G \equiv ?\ B_1, ..., \underline{B_i}, ..., B_m$.

$A \text{ :- } A_1, ..., A_n$. $\equiv c$ (de P)

$. . . \theta = umg(B_i, A)$

$?G_1 \equiv ?\ (B_1, ..., A_1, ..., A_n, ..., B_m)\theta$.

$?G_1$ se llama ***resolvente*** de $?G$ y c (sobre el literal B_i), con

θ = "unificador más general" de B_i y la cabeza A de c.

- SLD-derivación de $P \cup \{?G_0\}$

es una secuencia maximal de la forma:

$?\ G_0$ c_0 (de P)

$. . . \theta_0$

$?\ G_1$ c_1 (de P)

$. . . \theta_1$

$?\ G_i$ c_i (de P)

$. . . \theta_i$

$?\ G_{i+1}$

Cada $?G_{i+1}$ es un resolvente de $?G_i$ y c_i (con umg θ_i) y c_{i+1} no tiene variables comunes con $?G_0$, c_0 ,...,c_i

- SLD-refutación de $P \cup \{?G\}$

  - Una SLD-derivación puede ser finita ó infinita. Si <u>es finita</u> y su último resolvente es $?G_n$ entonces

    si $G_n = \square$ → derivación de <u>éxito</u>

    sino → derivación de <u>fallo</u>

  - Una SLD-refutación de un programa P y una pregunta ?G es una SLD-derivación de <u>éxito</u> de $P \cup \{?G\}$.

  - Si G tiene variables, la "respuesta del sistema" a dicha refutación es la sustitución:

$$\sigma = (\theta_0 \bullet \theta_1 \bullet .... \bullet \theta_{n-1}) \mid_{Var(G)}$$

obtenida por la composición de los umg θ_i de cada paso, referente a las variables de G.

<u>Ejemplo</u>: SLD-refutación en "HOSPITAL"

?G ≡ ? debe-tomar(ana,X).

(10) debe-tomar(P1, F1) :- padece(P1, E1), alivia(F1, E1).

θ_1 = {ana/P1, X/F1}

? padece(ana, E1), alivia(X, E1). (3) padece(ana, gripe).

θ_2 = {gripe/E1}

? alivia(X, gripe).

(11) alivia(F2, E2) :- es-síntoma(S2, E2), suprime(F2, S2).

θ_3 = {X/F2, gripe/E2}

? es-síntoma(S2, gripe), suprime(X, S2).

? es-síntoma(S2, gripe), suprime(X, S2).

(5) es-síntoma(fiebre, gripe).

θ_4 = {fiebre/S2}

? suprime(X, fiebre).

(8) suprime(paracetamol, fiebre).

θ_5 = {paracetamol/X}

? $\square$

Respuesta del sistema:

$\sigma = (\theta_1 \bullet \theta_2 \bullet \theta_3 \bullet \theta_4 \bullet \theta_5) \mid_{Var(G)}$ = {ana/P1, paracetamol/F1, gripe/E1, paracetamol/F2, gripe/E2, fiebre/S2, paracetamol/X} $\mid_{\{X\}}$ =
{paracetamol/X}

Semántica operacional de los lenguajes lógicos

- Dados P (programa lógico) y {?G } (pregunta ó input) se trata de ver si existe una SLD-refutación de P∪{?G}
- La semántica operacional viene determinada por
  - Regla de cómputo: regla de SLD-resolución
  - Estrategia de resolución:
    - ***Función de selección*** que determina sobre qué átomo de G se va a resolver
    - ***Elección de la cláusula input*** (de P) cuya cabeza unifique con el átomo seleccionado de G

- En Prolog:

Función de selección: primer átomo (a la izquierda) de G y

Elección de la cláusula input: en el orden introducido en P.

- Semántica operacional: Conjunto de éxitos

Dado un programa lógico P, se definen:

"Universo de Herbrand de P"

U_P ={términos (de base) formados con los símbolos de constante y símbolos de función que aparecen en P}

"Base de Herbrand de P"

B_P ={p($t_1,\ldots,t_n$) / p es un símbolo de predicado de P y $t_1,\ldots,t_n \in U_P$}

"Conjunto de éxitos de P"

O(P) = {A / A$\in B_P$, P∪{?A} $\vdash_{SLD} \square$}

- La semántica operacional de P determina el ***conjunto de éxitos*** de P, que es independiente de la función de selección elegida.

- Modelos de un programa lógico P

Para definir qué es un modelo de un programa lógico se necesitan dos conceptos: *H-interpretación* y *satisfacción* (|=)

- Una *H-interpretación* I para P es cualquier subconjunto de B_P
- *I satisface P (ó I es modelo de P),* denotado I |= P, si se tiene que

  I |= c', para toda instancia de base c' de cada cláusula c de P,

  donde la definición de I |= c' (con c' sin variables) es:

  I |= A . sii A$\in$I

  I |= A :- $A_1, \ldots, A_n$. sii (si $A_i \in$ I, $\forall$i=1,...,n entonces A$\in$I)

Nota: c' es una instancia de base de c si c' se obtiene al sustituir (consistentemente) las variables de c por elementos de U_P

Ejemplo: Modelos del programa "HOSPITAL"

$U_{HOSPITAL}$ = { jon, ana, carlos, gripe, hepatitis, alergia, fiebre, cansancio, estornudos, paracetamol, antihistamínico}

$B_{HOSPITAL}$ = { padece(t, t'), es-síntoma(t, t'), suprime(t, t'), alivia(t, t'), debe-tomar(t, t') / t, t'∈ $U_{HOSPITAL}$ }

Las H-interpretaciones J y K son modelos del programa pero I no:

I = {(1),...,(9)} ∪ { alivia(paracetamol,gripe), debe-tomar(jon,paracetamol) } |≠ HOSPITAL

J = I ∪ { alivia(antihistamínico,alergia), debe-tomar(ana, paracetamol), debe-tomar(carlos, antihistamínico) } |= HOSPITAL

K = J ∪ { debe-tomar(carlos, paracetamol) } |= HOSPITAL

Semántica "declarativa" (o denotacional)

- El "*menor modelo* de P", denotado M_P, es la H-interpretación que verifica:
  - M_P |= P
  - Para cualquier otra I tal que I |= P, se tiene que $M_P \subseteq$ I
- La *semántica declarativa de P* se define como M_P : "el modelo que satisface lo que afirma P y nada más"
- **Existencia del menor modelo de P**

  Dado P, existe siempre un menor modelo de P. Dem:

  B_P |= P

  Si I |= P y J |= P entonces I∩J |= P

  Por tanto, M_P= ∩{I / I |= P} es el menor modelo de P

- **Construcción del menor modelo de P**

Dado P, el *"operador de consecuencias inmediatas de P"*

T_P : {H-interpretaciones para P}→{H-interpretaciones para P} definido:

$T_P(I)$ = {A / A :- A_1, ..., A_n. es inst. base de alguna c de P (n≥0) y A_i∈ I, ∀i=1,...,n }

servirá para construir M_P gracias al teorema: $\boxed{\mathbf{M_P = T_P^{w}(\varnothing)}}$

siendo las potencias de T_P:

$T_P^0(I) = I$

$T_P^{n+1}(I) = T_P(T_P^n(I))$

$T_P^w(I) = \cup \{T_P^n(I) / n<w\}$ con w=cardinal de los naturales

Ejemplo: Construcción de M_P para P=HOSPITAL

$I_0 = T_P^0(\varnothing) = \varnothing$

$I_1 = T_P^1(\varnothing) = T_P(I_0)$ = {inst.base de los hechos de P} = {(1),(2),(3),(4),(5),(6),(7),(8),(9)}

$I_2 = T_P^2(\varnothing) = T_P(I_1) = I_1 \cup$ {alivia(paracetamol,gripe), alivia(antihistamínico,alergia)}

$I_3 = T_P^3(\varnothing) = T_P(I_2) = I_2 \cup$ {debe-tomar(jon,paracetamol), debe-tomar(ana, paracetamol), debe-tomar(carlos, antihistamínico)}

$T_P^4(\varnothing) = T_P(I_3) = I_3 \Rightarrow I_3$ es el menor punto fijo de T_P

Por lo tanto, el menor modelo del programa HOSPITAL es

$\boxed{M_{HOSPITAL} = I_3} = T_P^3(\varnothing) = T_P^4(\varnothing) = ... = T_P^w(\varnothing)$

Equivalencia entre semántica declarativa y operacional

Teorema 1: Para todo programa P $\boxed{O(P) \equiv M_P}$

"El menor modelo de P coincide con el conjunto de éxitos de P"

es decir, $\{A \,/\, A \in B_P \,,\, P \cup \{?A\} \vdash_{SLD} \square\} \equiv M_P$

Teorema 2 (Completitud de la SLD-resolución):

Para todo programa P y todo "goal" G,

$\boxed{rep(P) \wedge \{\neg\, rep(G)\} \text{ es inconsistente } \Leftrightarrow P \cup \{?G\} \vdash_{SLD} \square}$

[Nota: una fórmula es inconsistente cuando no tiene modelos]

- **Sustitución de respuesta correcta**

  σ es una <u>sustitución de respuesta correcta</u> para un programa P y un "goal" G, si σ es una sustitución de base para G (Gσ no tiene variables) y $M_P \models G\sigma$

- **Relación entre sust. resp. del sistema y sust. resp. correcta**

Teorema3 (Generalización del Teorema de Completitud):

Para todo programa P y todo "goal" G, se verifica que

*"para cada sust. resp. correcta **σ** existe una sust. resp. del sistema **θ** tal que **Gθ** es más general que **Gσ**"*

Este teorema nos dice que cualquier respuesta correcta se puede obtener como un caso particular de alguna respuesta del sistema.

<u>**Ejemplo 1**</u> Sea P = HOSPITAL y G = alivia(F,E)

σ = {paracetamol/F, gripe/E} es una *sust. resp. correcta* y también es una *sust. resp. del sistema* para P y G, porque:

- $M_P \models$ alivia(paracetamol, gripe)
- $P \cup \{$?alivia(F,E)$\} \vdash_{SLD} \square$ con respuesta σ (*ejercicio)

<u>**Ejemplo 2**</u>

Sea P = {quiere(Z,jon)., estudiante(ana)., estudiante(gorka).}
y sea G = quiere(X,Y)

σ = {ana/X, jon/Y} es una *sust. resp. correcta* para P y G porque Gσ = quiere(ana,jon) se satisface en M_P ya que

M_P={quiere(jon,jon), quiere(gorka,jon), quiere(ana, jon), estudiante(ana), estudiante(gorka)}

Sin embargo, σ **no** se puede obtener directamente como *respuesta del sistema* para P∪{?G }. Pero σ sí es un caso particular de la *respuesta del sistema* θ obtenida:

? quiere(X,Y)   quiere(Z,jon).

θ = {Z/X, jon/Y}

□