Tema AA–5: ILP: Sistemas y aplicaciones

José A. Alonso Jiménez
Miguel A. Gutiérrez Naranjo

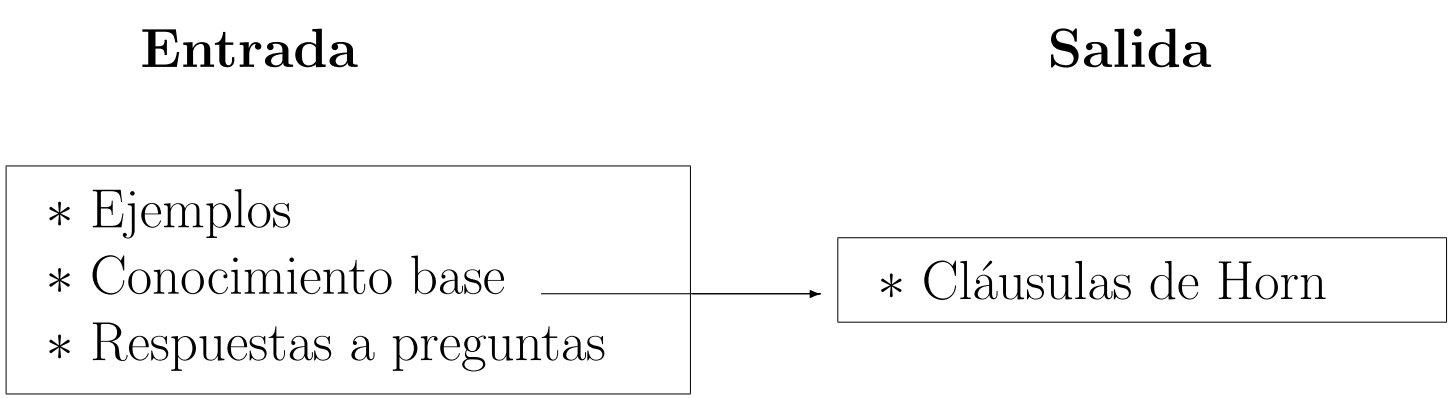
Dpto. de Ciencias de la Computación e Inteligencia Artificial
Universidad de Sevilla

Introducción

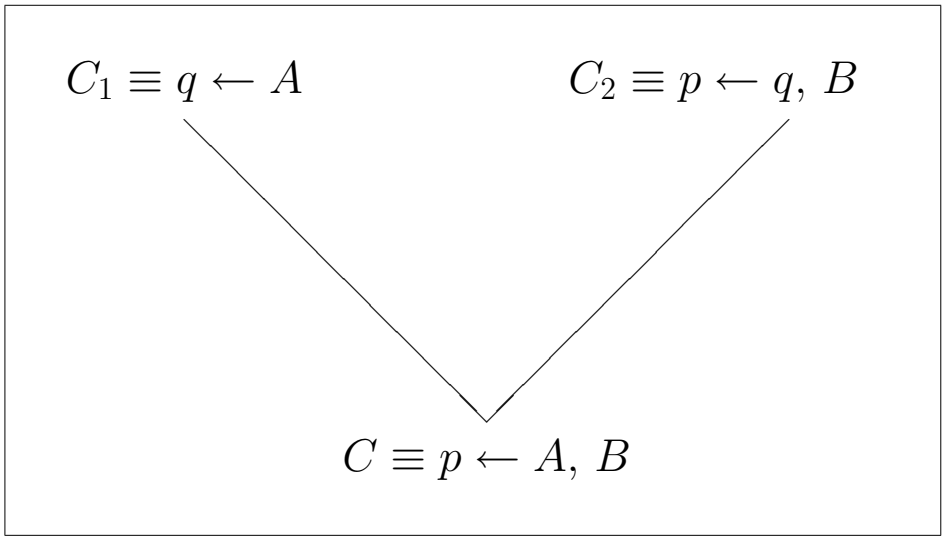
- **Sistemas:** CIGOL, Golem, Progol, ...
  - Breve descripción
  - Experimentos de laboratorio
  - Aplicaciones en el mundo real
- **Nuevas fronteras:** Invención de predicados
- **Documentación:**
  - Artículos
  - Internet

CIGOL (S. Muggleton y W. Buntine, 1988)

- Sistema interactivo
- Método ascendente
- Basado en resolución inversa



Operadores en CIGOL (I)

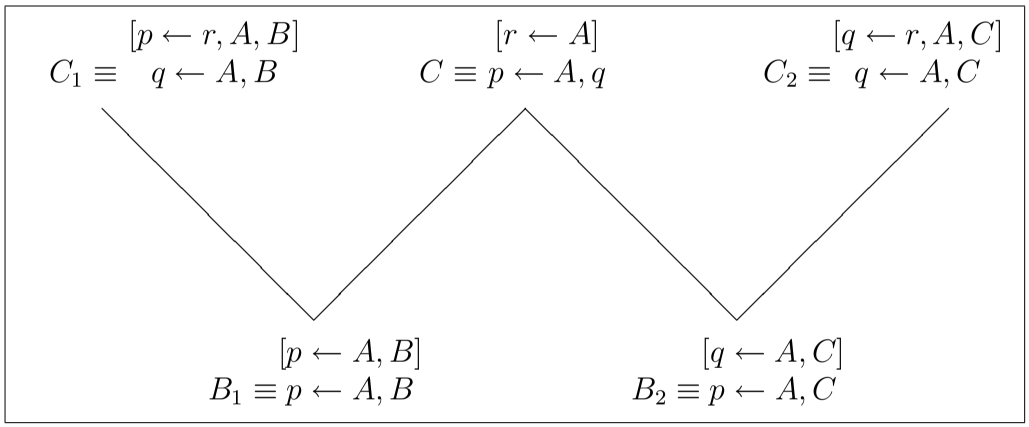


Absorción: $\dfrac{q \leftarrow A \qquad p \leftarrow A, B}{q \leftarrow A \qquad p \leftarrow q, B}$

Identificación: $\dfrac{p \leftarrow A, B \qquad p \leftarrow q, B}{q \leftarrow A \qquad p \leftarrow q, B}$

Operadores en CIGOL (II)

W–operadores



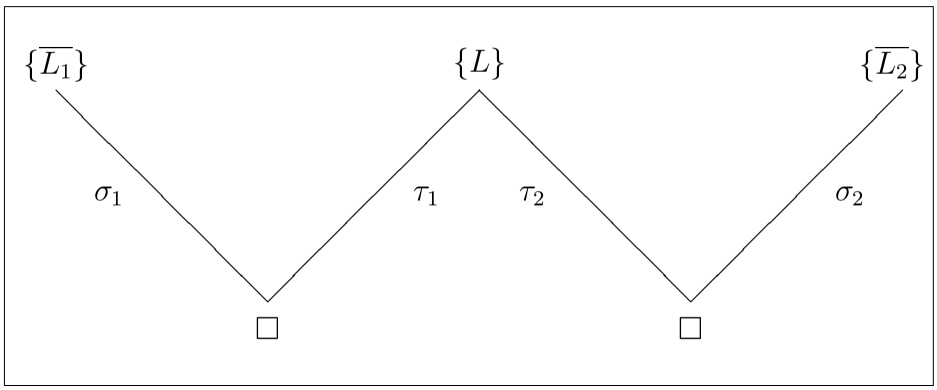
Intra–construcción:

$$\frac{p \leftarrow A, B \qquad\qquad p \leftarrow A, C}{q \leftarrow A, B \quad p \leftarrow A, q \quad q \leftarrow A, C}$$

Inter–construcción:

$$\frac{p \leftarrow A, B \qquad\qquad q \leftarrow A, C}{p \leftarrow r, A, B \quad r \leftarrow A \quad q \leftarrow r, A, C}$$

El operador de truncamiento



Ejemplos con CIGOL (I)

Concatenación de listas

| ?- **cigol.**

!- conc([s],[t],[s,t]).

...

I know:

conc([s],[t],[s,t]).

!- conc([],[a],[a]).

...

Is conc(A,[B],[C|D]) always true? **n.**

...

I know:

conc([],[a],[a]).
conc([s],[t],[s,t]).
not(conc(A,[B],[C|D])).

!- conc([],[1,2],[1,2]).

...

Is conc([],[A|B],[A|B]) always true? **y.**

I know:

conc([],[A|B],[A|B]).
conc([s],[t],[s,t]).
not(conc(A,[B],[C|D])).

!- conc([1],[2,3],[1,2,3]).

...

OPERADORES

Truncamiento

Truncamiento

Superado el tiempo límite 100000 ms. para Absorción

Truncamiento

L_1=conc([],[a],[a])

L_2=conc([],[1,2],[1,2])

Truncamiento

L_1=conc([s],[t],[s,t])

L_2=conc([1],[2,3],[1,2,3])

Ejemplos con CIGOL (II)

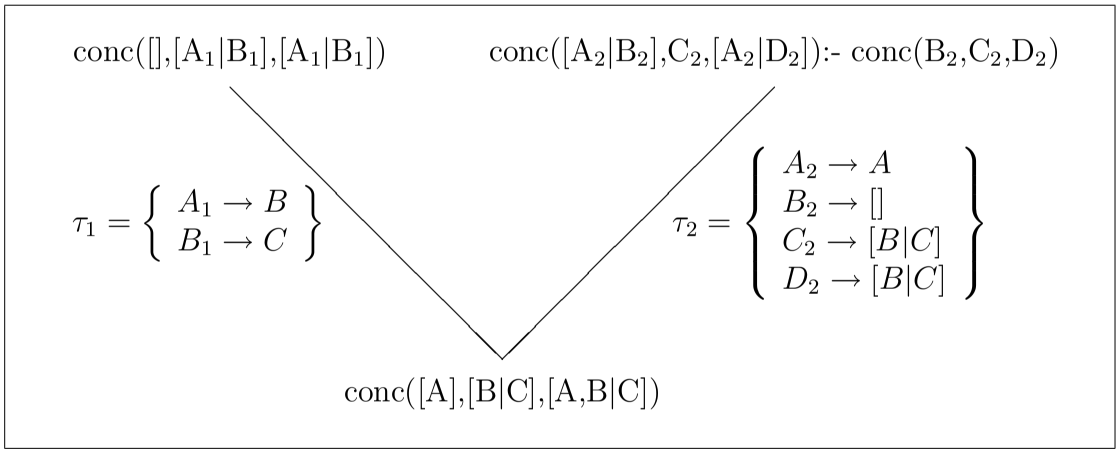
Is conc([A],[B|C],[A,B|C]) always true? **y.**

...

Absorción

conc([],[A|B],[A|B])

conc([A],[B|C],[A,B|C])



New clauses:[(conc([A|B],C,[A|D]):- conc(B,C,D)]

cover new facts: conc([A,B],[C|D],[A,B,C|D])] ...

Are new clauses always true: **y.**

...

I know:

conc([],[A|B],[A|B]).

conc([A|B],C,[A|D]):- conc(B,C,D).

not(conc(A,[B],[C|D])).

!- conc([],[],[]).

Truncamiento

...

Is conc([],A,A) always true? **y.**

...

I know:

conc([],A,A).

conc([A|B],C,[A|D]):- conc(B,C,D)

not(conc(A,[B],[C|D])).

Golem (S. Muggleton y C. Feng, 1990)

La menor generalización general

- Plotkin (1970) y Reynolds (1970) dotan al conjunto de términos de un lenguaje de estructura de retículo mediante la relación de θ–subsunción.
- Un concepto C *subsume* a otro D si $D \subset C$
- La cláusula C_1 θ–subsume a la cláusula C_2 si existe una sustitución θ tal que $C_1\theta \subset C_2$
- Forman *retículo*, i.e., dadas C_1 y C_2 existe un único $inf(C_1, C_2)$ y un único $sup(C_1, C_2)$ que llamaremos *menor generalización general* de C_1 y C_2.

Golem (mgg)

Menor generalización general (mgg):

Sea $\psi : TERM \times TERM \rightarrow VAR$

$$mgg(f(t_1,\ldots,t_n), g(s_1,\ldots,s_n)) = \begin{cases} f(mgg(t_1,s_1),\ldots,mgg(t_n,s_n)) & \text{Si } f = g \\ \psi(f(t_1,\ldots,t_n), g(s_1,\ldots,s_n)) & \text{e.o.c.} \end{cases}$$

$$mgg(p(t_1,\ldots,t_n), q(s_1,\ldots,s_n)) = \begin{cases} p(mgg(t_1,s_1),\ldots,mgg(t_n,s_n)) & \text{Si } p = q \\ \text{No definida} & \text{e.o.c.} \end{cases}$$

$$mgg(C_1, C_2) = \{mgg(l_1, l_2) : l_1 \in C_1,\, l_2 \in C_2\}$$

Ejemplos con Golem (I)

Lógica proposicional

- **Lenguaje:**
  - Dos símbolos proposicionales: p y q
  - Cinco conectivas: $\neg$, $\vee$, $\wedge$, $\rightarrow$ y $\leftrightarrow$.
- **Definición:**
  1. p es una fórmula.
  2. q es una fórmula.
  3. Si F es una fórmula, también $(\neg F)$.
  4. Si F y G son fórmulas, entonces también lo son $(F \vee G)$, $(F \wedge G)$, $(F \rightarrow G)$ y $(F \leftrightarrow G)$.

$\mathbf{O}^+$

$f([q, \rightarrow, [\neg, q]]).$
$f([p, \wedge, q]).$
$f([\neg, p]).$
$f([q, \wedge, p]).$
$f([[\neg, p], \wedge, [\neg, q]]).$
$f([p, \leftrightarrow, q]).$
$f(q).$
$f([q, \leftrightarrow, p]).$
$f([[\neg, p], \leftrightarrow, [p, \wedge, q]]).$
$f([p, \leftrightarrow, [\neg, p]]).$

$f([\neg, [\neg, p]]).$
$f([q, \vee, p]).$
$f([[\neg, q], \vee, [p, \vee, q]]).$
$f([p, \vee, [\neg, p]]).$
$f([p, \rightarrow, q]).$
$f(p).$
$f([\neg, q]).$
$f([p, \vee, q]).$
$f([q, \rightarrow, p]).$

$\mathbf{O}^-$

$f([\neg, \neg]).$
$f([q, \vee, \vee]).$
$f([\neg, \vee, p]).$
$f([\neg, \rightarrow, q]).$
$f([q, q, p]).$
$f([p, \rightarrow, \rightarrow]).$
$f([p, p, q]).$
$f([\wedge, p, p]).$
$f([p, \wedge, \vee]).$
$f([\neg, \wedge, q]).$

$f([p, \leftrightarrow, \wedge]).$
$f([\vee, \leftrightarrow, p]).$
$f([\leftrightarrow, \leftrightarrow, \leftrightarrow]).$
$f([p, p, [p, \leftrightarrow, q]]).$
$f([\neg, p, [\neg, p]]).$
$f([[\neg, q], \leftrightarrow, [p, \vee, \wedge]]).$

- **Respuesta:**

$f(q).$
$f(p).$
$f([\neg, A]) : -f(A).$
$f([A, \rightarrow, B]) : -f([\neg, A]), f(B).$
$f([A, \wedge, B]) : -f([\neg, A]), f(B).$
$f([A, \leftrightarrow, B]) : -f([\neg, A]), f(B).$
$f([A, \vee, B]) : -f(A), f(B).$

Ejemplos con Golem (II.a)

Lenguaje natural

- **Lenguaje**
  - Ocho determinantes **(d)**: *el, la, los, las, un, una, unos, unas.*
  - Ocho nombres comunes **(n)**: *hombre, hombres, mujer, mujeres, niño, niños, niña, niñas.*
  - Ocho adjetivos **(a)**: *moreno, morena, morenos, morenas, rubio, rubia, rubios, rubias.*
  - Ocho nombres propios **(np)**: *Pepe, Paco, Antonio, Eduardo, María, Ana, Rosa, Julia.*
- **Conocomiento base** *(Clasificación)***:**

  cl(paco,[np,m,s])
  cl(rubias,[a,f,p])

Ejemplos con Golem (II.b)

- **Ejemplos:** Estructura: *(np), (d)+(n), (d)+(n)+(a)*

  O$^+$: *sn([la,niña,morena])*
  O$^-$: *sn([una,niños,moreno])*

- **Respuesta:**

$$sn([la, mujer]).$$
$$sn([una, mujer]).$$
$$sn([A, niña]) : -cl(A, [d, f, s]).$$
$$sn([A]) : -cl(A, [np, B, s]).$$
$$sn([A, B]) : -cl(A, [d, m, s]), cl(B, [n, m, s]).$$
$$sn([A, B]) : -cl(A, [d, C, p]), cl(B, [n, C, p]).$$
$$sn([A, B, C]) : -cl(A, [d, D, E]), cl(B, [n, D, E]), cl(C, [a, D, E]).$$

Aplicación con Golem (I)

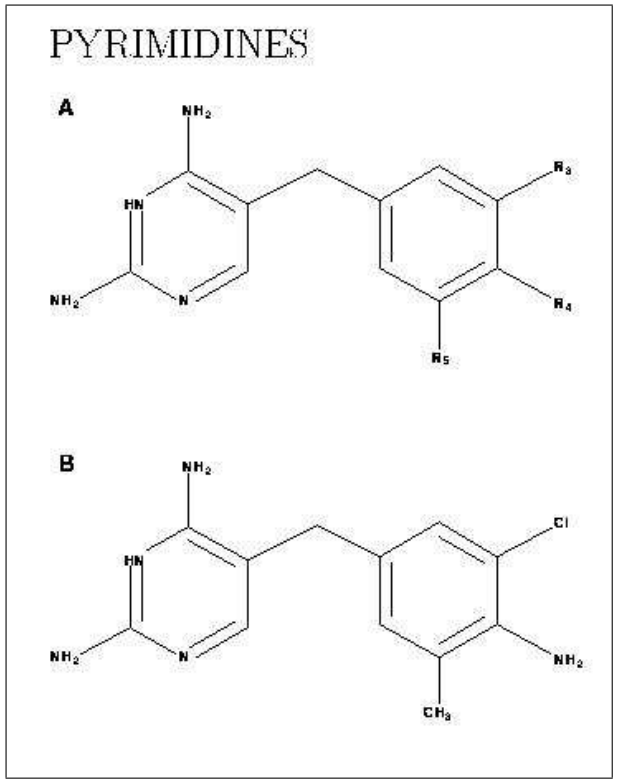
Predicción de la estructura secundaria de las proteinas

(OUCL en cooperación con Imperial Cancer Research Fund)

- Dada la estructura primaria de una proteina (secuencia de aminoácidos),
  - Encontrar la estructura secundaria
  - Predecir si los residuos individuales formaán una hélice levógira
- **Ejemplos:** 12 proteinas no homólogas (1612 residuos)
- **Conocimiento base:** Propiedades físicas y químicas de los residuos individuales y su posición relativa dentro de la proteina
- **Sistema:** GOLEM
- 21 cláusula producidas, cada una de unos 15 literales
- Su precisión sobre un test independiente fue del 82%, mientras que la precisión del mejor método convencional fue del 73%

Aplicación con Golem (II)

Predicción y Comparación de la acción de fármacos



- **Ejemplos:** 44 fármacos que se ajustan a la plantilla A
- **Conocimiento base:** Propiedades químicas de los sustituyentes
- **Sistema ILP:** GOLEM
- Las cláusulas inducidas fueron consideradas como una teoría novedosa por los químicos
- La correlación entre el resultado de la predicción y la acción real de los fármacos estudiados fue mejor que la alcanzada por métodos de regresión.

Aplicación con Golem (III)

Clasificación biológica de la calidad del agua de un río

- **Dada** una lista de indicadores biológicos tomados en distintas muestras de agua y sus niveles de abundancia, **clasificarlos** en una de las cinco clases B1a, B1b, B2, B3, B4.
- **Ejemplos:** 300 muestras de la cuenca superior de un río de Gran Bretaña, clasificados por expertos.
- **Conocimiento base:** Relaciones entre los niveles de abundancia.
- **Sistemas:** GOLEM, CLAUDIEN
- Reglas descubiertas interesantes (según evaluación experta):

$$b1b(X) \leftarrow ancilidae(X, A), gammaridae(X, B), \ldots, greater\_than(D, B).$$

Progol

- Método ascendente
- El usuario especifica qué expresiones de la lógica de primer orden pueden usarse como espacio de hipótesis H (declaraciones de modo)
- Para cada ejemplo $(x_i, f(x_i))$ no cubierto por por la hipótesis actual, Progol busca la hipótesis más específica h_i en H tal que

$$(B \wedge h_i \wedge x_i) \vdash f(x_i)$$

Ejemplo con Progol (I)

- **Declaraciones de modo:**

$$:-modeb(1, tiene\_agallas(+animal))?$$

  i.e., el predicado $tiene\_agallas/1$ puede aparecer en el cuerpo de las cláusulas de salida, que recibe como entrada un argumento de tipo *animal* y que este literal sólo puede tener éxito una vez.

- **Tipos:** $animal(perro)$, $animal(delfin)$, $animal(tortuga)$, ...
- **Conocimiento base:** $numero\_de\_patas(perro, 4)$, $tiene\_agallas(tiburon)$, ...
- **Ejemplos:** $clase(aguila, ave)$, $clase(perro, mamifero)$, $clase(tiburon, pez)$ ...

Ejemplo con Progol (II)

- **Salida:**

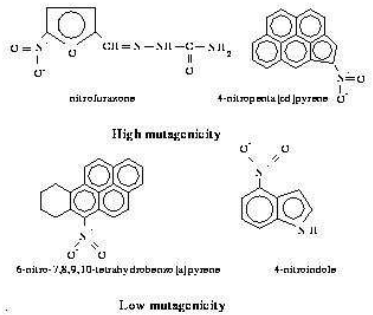
```
clase(A,mamifero) :- tiene_leche(A).
clase(A,reptil) :- en_la_piel_tiene(A,escamas),
                   habitat(A,tierra).
clase(A,ave) :- en_la_piel_tiene(A,plumas).
clase(A,pez) :- tiene_agallas(A).
clase(A,reptil) :- en_la_piel_tiene(A,escamas),
                   numero_de_patas(A,4).
[Total number of clauses = 5]

[Time taken 6.380s]
```

Aplicación con Progol

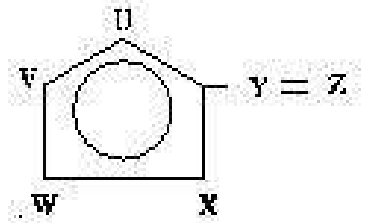
PREDICCION DE *MUTAGENICITY*

(OUCL en cooperación con el Imperial Cancer Research Fund)

Componentes nitro-aromaticos



Nueva estructura



- **Ejemplos** 188 componentes "positivos" y 42 componentes "negativos"
- **Conocimiento base:** Propiedades de átomos y enlaces
- **Sistema:** PROGOL
- Descubierta una nueva clave estructural para *mutagenicity* alta
- Mejor que la regresión sobre conjuntos "negativos" (88% vs 69%), y comparable en el caso de conjuntos "positivos" (88 % vs 89%)

Invención de predicados (I)

Derivadas de potencias de una variable

- **Conocimiento base:** *Para toda potencia de una variable x, x^m, existe un único monomio en esa variable, ax^b, que representa la derivada de dicha potencia respecto de la variable.*
- **Ejemplos:**

$$\frac{d\,x^2}{dx} = 2x \qquad\qquad \frac{d\,x^5}{dx} = 5x^4$$

- **Hipótesis**:

$$\frac{d\,x^m}{dx} = mx^{m-1}$$

| ?- **cigol.**
!- **deriv([1,X,2],[2,X,1]).**
...
I know:
deriv([1,A,2],[2,A,1]).

!- **deriv([1,X,5],[5,X,4]).**
...

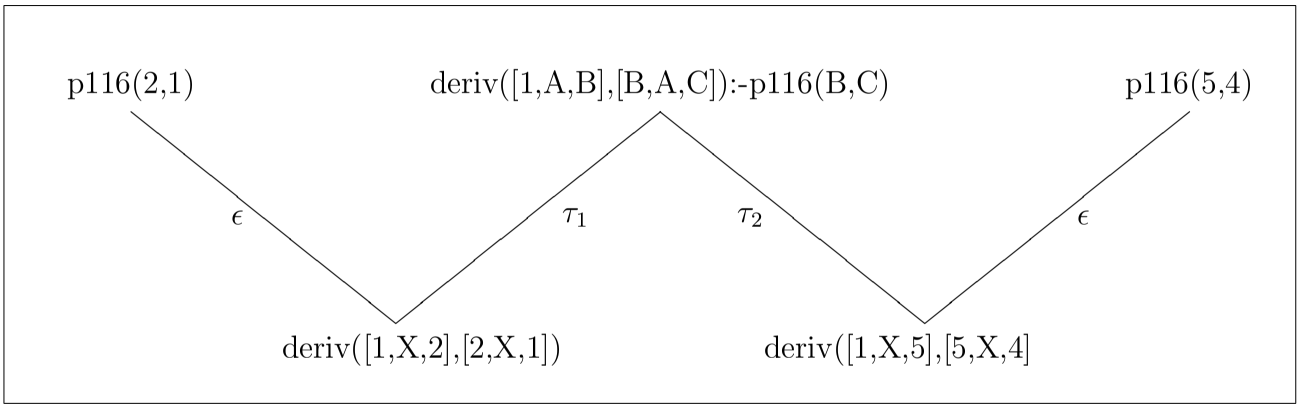
Is deriv([1,A,B],[B,A,C]) always true? **n.**

OPERADORES
Truncamiento

Truncamiento
L_1=deriv([1,X,2],[2,X,1])
L_2=deriv([1,X,5],[5,X,4])

Intra–construcción
deriv([1,X,2],[2,X,1])
deriv([1,X,5],[5,X,4])

Invención de predicados (II)



deriv([1,A,B],[B,A,C]:- p116(B,C).
p116(2,1).
p116(5,4).

What shall I call p116? **'menos_1'**

...

I know:

deriv([1,A,B],[B,A,C]:- menos_1(B,C).
menos_1(2,1).
menos_1(5,4).
not(deriv([1,A,B],[B,A,C])).

Artículos

- Lavrač, N. y De Raedt, L. *Inductive Logic Programming: A survey of European Research* AICOM Vol. 8,1 pp.: 3–19 Marzo 1995
- Muggleton, S. *Inductive Logic Programming* First Conference on Algorithmic Learning Theory, Tokio, Ohmsha, 1990
- Muggleton, S. y Buntine, W. *Machine invention of first order predicates by inverting resolution* Proc. 5th International Conference on Machine Learning pp.: 339–352. Morgan–Kaufmann, 1988
- Muggleton, S. y De Raedt, L. *Inductive Logic Programming: Theory and Methods* Journal of Logic Programming 19,20 pp.: 629–679, 1994
- Muggleton, S. y Feng, C. *Efficient induction of logic programs* pp.:281–298 en *Inductive Logic Programming* S. Muggleton (Ed.) Academic Press, 1992
- Muggleton, S. *Inverse Entailment and Progol* New Generation Computing Journal, May 1995

Internet

- Curso de la Universidad de Oxford

  `http://www.comlab.ox.ac.uk/oucl/courses/msc-comp/ilp/index.html`

- ILPNET

  `http://www-ai.ijs.si/ilpnet.html`

- Universidad de York

  `http://www.cs.york.ac.uk/mlg/`

- The Online School on Inductive Logic Programing and Knowledge Discovery in Databases

  `http://www-ai.ijs.si/~ilpnet/ilpkdd/`

- Base de datos del GMD

  `http://www.gmd.de/ml-archive/`