

Tema 2: Listas, operadores y aritmética

Grupo de Programación Declarativa

{José A. Alonso, Andrés Cordon y Miguel A. Gutiérrez}

Dpto. de Ciencias de la Computación e Inteligencia Artificial

UNIVERSIDAD DE SEVILLA

Listas

- Definición de listas

- La lista vacía $[]$ es una lista.
- Si L es una lista, entonces $.(a,L)$ es una lista.

- Ejemplos

?- $.(a, []) = [a]$

Yes

?- $.(a, .(b, [])) = [a,b]$

Yes

?- $.(X,Y) = [a] .$

$X = a$

$Y = []$

?- $.(X,Y) = [a,b] .$

$X = a$

$Y = [b]$

?- $.(X, .(Y,Z)) = [a,b] .$

$X = a$

$Y = b$

$Z = []$

Listas

- Escritura abreviada

$$[X|Y] = .(X,Y)$$

- Ejemplos con escritura abreviada

$$?- [X|Y] = [a,b] .$$

$$X = a$$

$$Y = [b]$$

$$?- [X|Y] = [a,b,c,d] .$$

$$X = a$$

$$Y = [b, c, d]$$

$$?- [X,Y|Z] = [a,b,c,d] .$$

$$X = a$$

$$Y = b$$

$$Z = [c, d]$$

Listas

- Concatenación de listas (append)

- *Especificación:* $\text{conc}(A,B,C)$ se verifica si C es la lista obtenida escribiendo los elementos de la lista B a continuación de los elementos de la lista A . Por ejemplo,

$?- \text{conc}([a,b], [b,d], C).$
 $C = [a,b,b,d]$

- *Descripción:*

1. Si A es la lista vacía, entonces la concatenación de A y B es B .
2. Si A es una lista cuyo primer elemento es X y cuyo resto es D , entonces la concatenación de A y B es una lista cuyo primer elemento es X y cuyo resto es la concatenación de D y B .

- *Definición 1:*

$\text{conc}(A,B,C) :- A=[], C=B.$
 $\text{conc}(A,B,C) :- A=[X|D], \text{conc}(D,B,E), C=[X|E].$

- *Definición 2:*

$\text{conc}([],B,B).$
 $\text{conc}([X|D],B,[X|E]) :- \text{conc}(D,B,E).$

Listas

- *Nota:* Analogía entre la definición de `conc` y la de suma,
- *Consulta 1:* ¿Cuál es el resultado de concatenar las listas `[a,b]` y `[c,d,e]`?

```
?- conc([a,b],[c,d,e],L).  
L = [a, b, c, d, e]
```

- *Consulta 2:* ¿Qué lista hay que añadirle al lista `[a,b]` para obtener `[a,b,c,d]`?

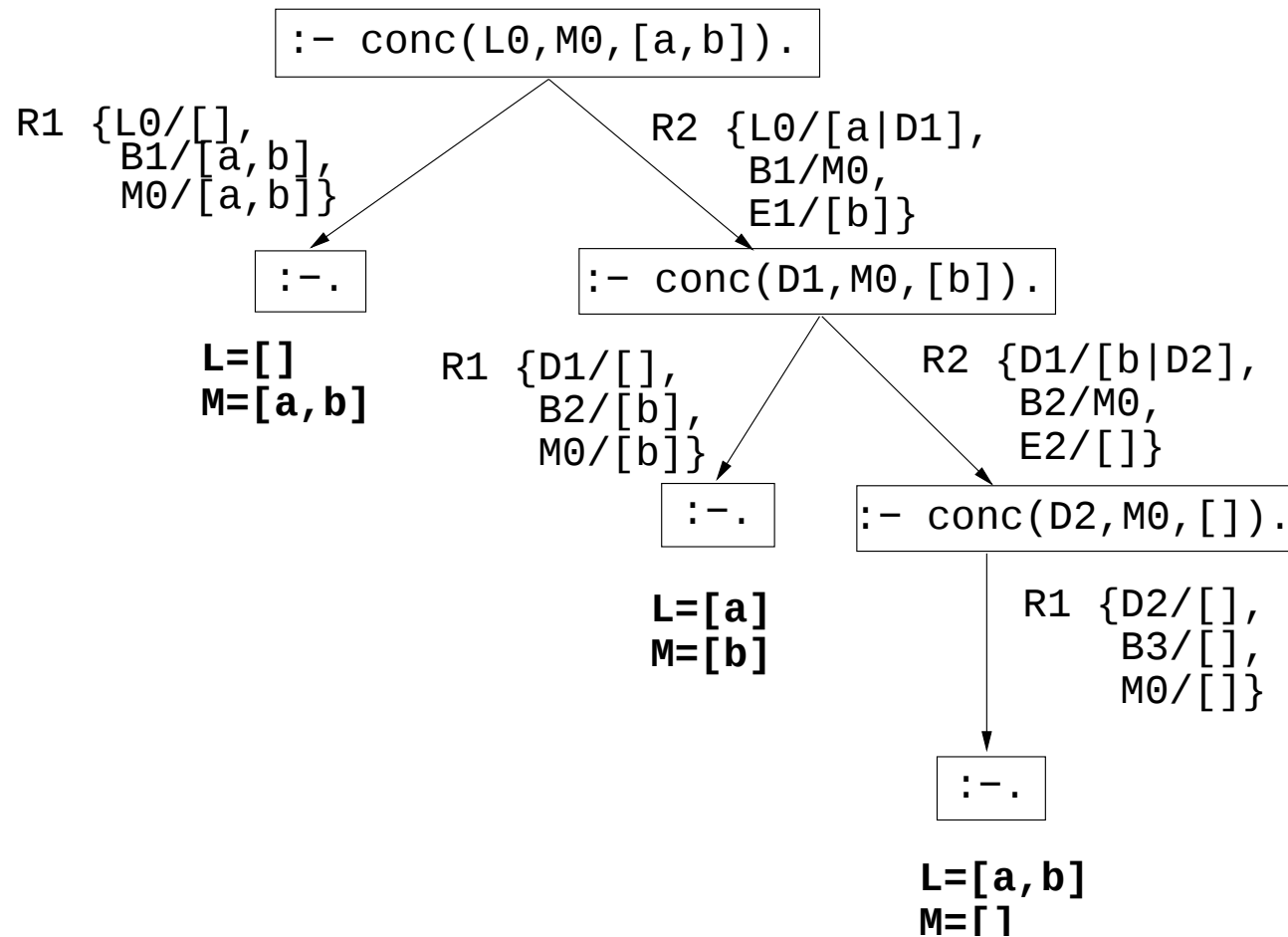
```
?- conc([a,b],L,[a,b,c,d]).  
L = [c, d]
```

- *Consulta 3:* ¿Qué dos listas hay que concatenar para obtener `[a,b]`?

```
?- conc(L,M,[a,b]).  
L = []  
M = [a, b] ;  
L = [a]  
M = [b] ;  
L = [a, b]  
M = [] ;  
No
```

Listas

- Árbol de deducción correspondiente a $?- \text{conc}(L, M, [a, b])$.



Listas

- La relación de pertenencia (member)
 - *Especificación:* pertenece(X,L) se verifica si X es un elemento de la lista L.
 - *Definición 1:*

```
pertenece(X, [X|L]).  
pertenece(X, [Y|L]) :- pertenece(X,L).
```

- *Definición 2:*

```
pertenece(X, [X|_]).  
pertenece(X, [_|L]) :- pertenece(X,L).
```

Listas

- *Consultas:*

?- pertenece(b,[a,b,c]).

Yes

?- pertenece(d,[a,b,c]).

No

?- pertenece(X,[a,b,a]).

X = a ;

X = b ;

X = a ;

No

?- pertenece(a,L).

L = [a|_G233] ;

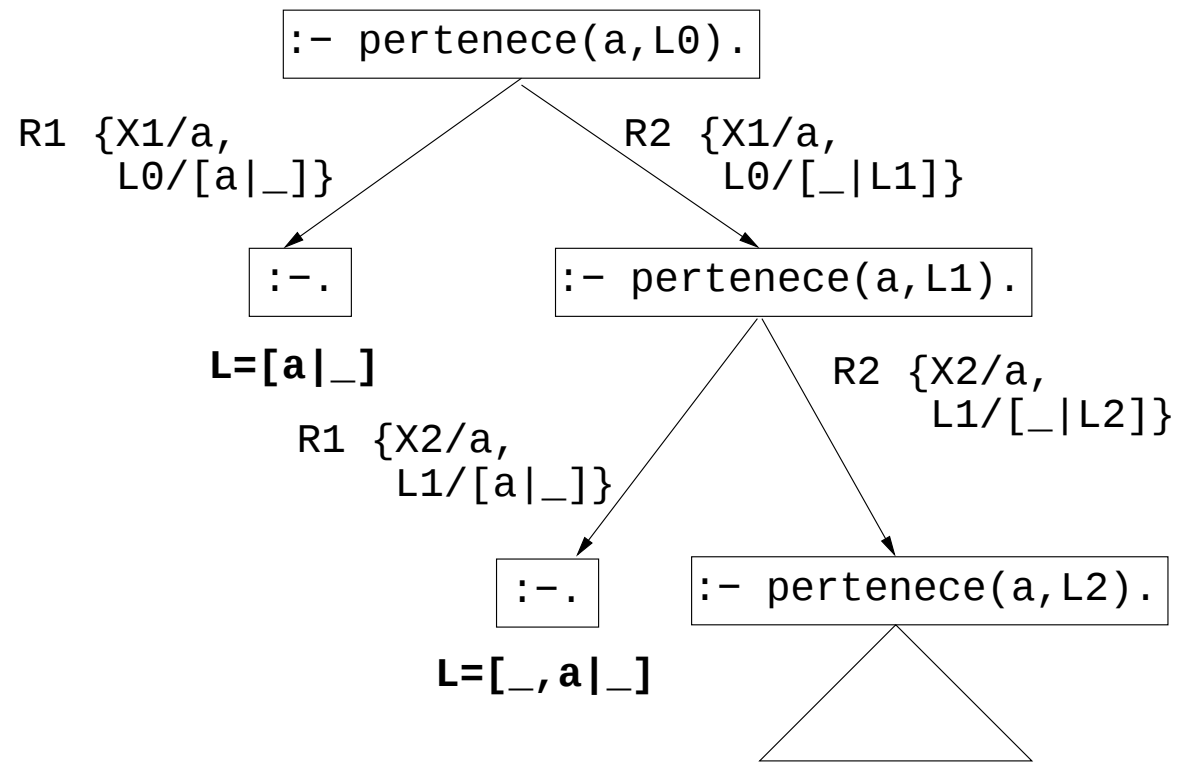
L = [_G232, a|_G236] ;

L = [_G232, _G235, a|_G239]

Yes

Listas

- Árbol de deducción de $?- \text{pertenece}(a, L)$.



Disyunciones

- Definición de pertenece con disyunción

`pertenece(X,[Y|L]) :- X=Y ; pertenece(X,L).`

- Definición equivalente sin disyunción

`pertenece(X,[Y|L]) :- X=Y.`

`pertenece(X,[Y|L]) :- pertenece(X,L).`

Operadores

- Ejemplos de notación infija y prefija en expresiones aritméticas

?- +(X,Y) = a+b.

X = a

Y = b

?- +(X,Y) = a+b+c.

X = a+b

Y = c

?- +(X,Y) = a+(b+c).

X = a

Y = b+c

?- a+b+c = (a+b)+c.

Yes

?- a+b+c = a+(b+c).

No

Operadores

- Operadores aritméticos predefinidos:

Precedencia	Tipo	Operadores	
500	yfx	+, -	Infijo asocia por la izquierda
500	fx	-	Prefijo no asocia
400	yfx	*, /	Infijo asocia por la izquierda
200	xfy	^	Infijo asocia por la derecha

- Ejemplos de asociatividad

?- $X^Y = a^{b^c}$.

$X = a$

$Y = b^c$

?- $a^{b^c} = (a^b)^c$.

No

?- $a^{b^c} = a^{(b^c)}$.

Yes

Operadores

- Ejemplo de precedencia

?- $X+Y = a+b*c.$

$X = a$

$Y = b*c$

?- $X*Y = a+b*c.$

No

?- $X*Y = (a+b)*c.$

$X = a+b$

$Y = c$

?- $a+b*c = a+(b*c).$

Yes

?- $a+b*c = (a+b)*c.$

No

Operadores

- Definición de operadores

- Definición (ejemplo_operadores.pl)

```
:-op(800,xfx,estudian).  
:-op(400,xfx,y).
```

```
juan y ana estudian lógica.
```

- Consultas

```
?- [ejemplo_operadores].  
Yes
```

```
?- Quienes estudian lógica.  
Quienes = juan y ana
```

```
?- juan y Otro estudian Algo.  
Otro = ana  
Algo = lógica
```

Aritmética

- Evaluación de expresiones aritmética con is

```
?- X is 2+3^3.
```

```
X = 29
```

```
?- X is 2+3, Y is 2*X.
```

```
X = 5
```

```
Y = 10
```

- Relaciones aritméticas: <, =<, >, >=, := y /=

```
?- 3 =< 5.
```

```
Yes
```

```
?- 3 > X.
```

```
[WARNING: Arguments are not sufficiently instantiated]
```

```
?- 2+5 = 10-3.
```

```
No
```

```
?- 2+5 := 10-3.
```

```
Yes
```

```
?- 2+5 /= 10-3.
```

```
No
```

Aritmética

- Definición de procedimientos aritméticos

- `factorial(X,Y)` se verifica si Y es el factorial de X. Por ejemplo,

```
?- factorial(3,Y).
```

```
Y = 6 ;
```

```
No
```

- Definición

```
factorial(1,1).
```

```
factorial(X,Y) :-
```

```
    X > 1,
```

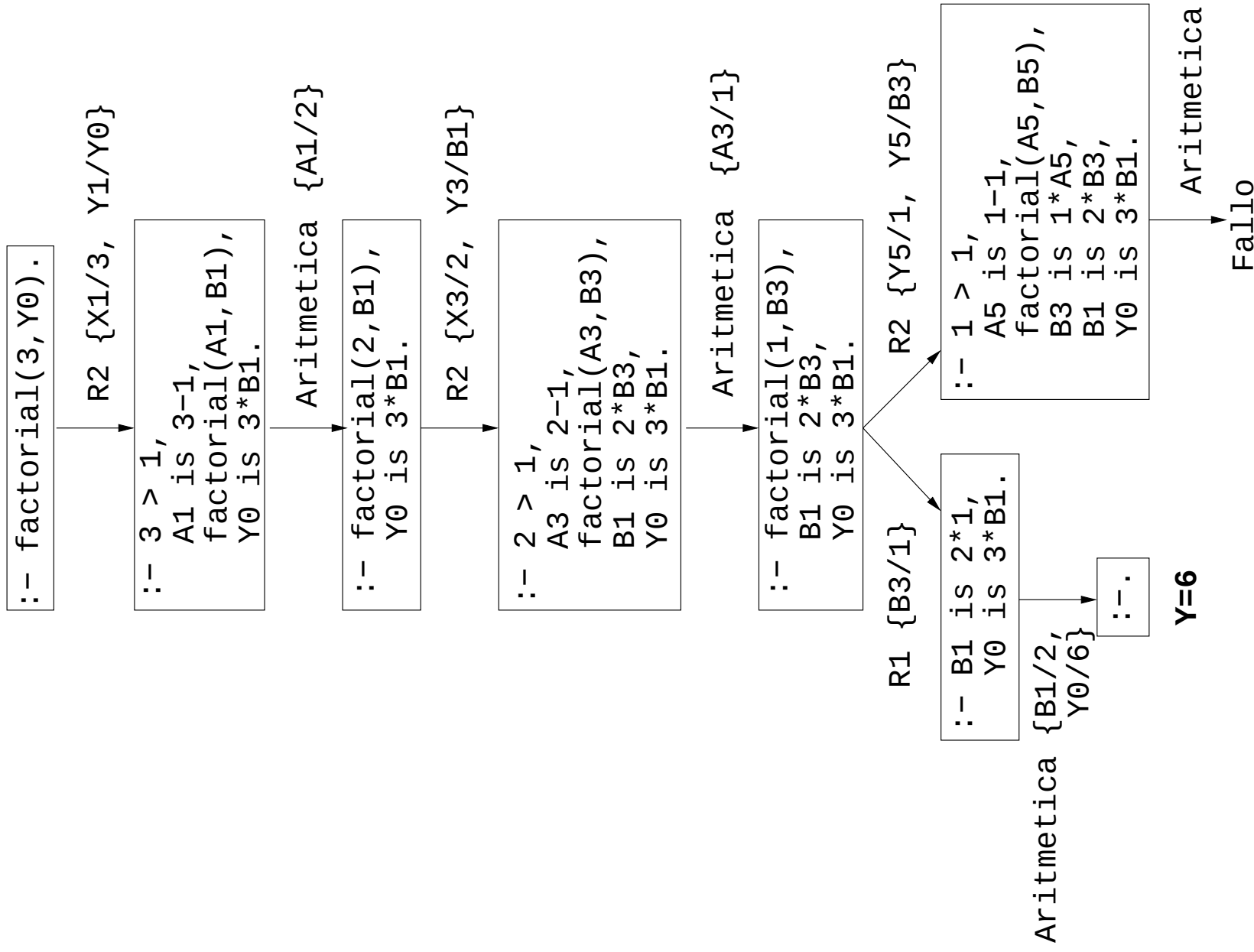
```
    A is X - 1,
```

```
    factorial(A,B),
```

```
    Y is X * B.
```

Aritmética

- Árbol de deducción de $?- \text{factorial}(3, Y)$.



Bibliografía

- Alonso, J.A. y Borrego, J. *Deducción automática (Vol. 1: Construcción lógica de sistemas lógicos)* (Ed. Kronos, 2002)
- Bratko, I. *Prolog Programming for Artificial Intelligence (2nd ed.)* (Addison–Wesley, 1990)
- Clocksin, W.F. y Mellish, C.S. *Programming in Prolog (Fourth Edition)* (Springer Verlag, 1994)
- Covington, M.A.; Nute, D. y Vellino, A. *Prolog Programming in Depth* (Prentice Hall, 1997)
- Sterling, L. y Shapiro, E. *L'art de Prolog* (Masson, 1990)
- Van Le, T. *Techniques of Prolog Programming* (John Wiley, 1993)